

Big Data Analytics for Large-Scale Multimedia Search

Big Data Analytics for Large-Scale Multimedia Search

Edited by

Stefanos Vrochidis

Information Technologies Institute, Centre for Research and Technology Hellas
Thessaloniki, Greece

Benoit Huet

EURECOM
Sophia-Antipolis
France

Edward Y. Chang

HTC Research & Healthcare
San Francisco, USA

Ioannis Kompatsiaris

Information Technologies Institute, Centre for Research and Technology Hellas
Thessaloniki, Greece

WILEY

This edition first published 2019
© 2019 John Wiley & Sons Ltd.

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording or otherwise, except as permitted by law. Advice on how to obtain permission to reuse material from this title is available at <http://www.wiley.com/go/permissions>.

The right of Stefanos Vrochidis, Benoit Huet, Edward Y. Chang and Ioannis Kompatsiaris to be identified as the authors of the editorial material in this work asserted in accordance with law.

Registered Offices

John Wiley & Sons, Inc., 111 River Street, Hoboken, NJ 07030, USA

John Wiley & Sons Ltd, The Atrium, Southern Gate, Chichester, West Sussex, PO19 8SQ, UK

Editorial Office

The Atrium, Southern Gate, Chichester, West Sussex, PO19 8SQ, UK

For details of our global editorial offices, customer services, and more information about Wiley products visit us at www.wiley.com.

Wiley also publishes its books in a variety of electronic formats and by print-on-demand. Some content that appears in standard print versions of this book may not be available in other formats.

Limit of Liability/Disclaimer of Warranty

While the publisher and authors have used their best efforts in preparing this work, they make no representations or warranties with respect to the accuracy or completeness of the contents of this work and specifically disclaim all warranties, including without limitation any implied warranties of merchantability or fitness for a particular purpose. No warranty may be created or extended by sales representatives, written sales materials or promotional statements for this work. The fact that an organization, website, or product is referred to in this work as a citation and/or potential source of further information does not mean that the publisher and authors endorse the information or services the organization, website, or product may provide or recommendations it may make. This work is sold with the understanding that the publisher is not engaged in rendering professional services. The advice and strategies contained herein may not be suitable for your situation. You should consult with a specialist where appropriate. Further, readers should be aware that websites listed in this work may have changed or disappeared between when this work was written and when it is read. Neither the publisher nor authors shall be liable for any loss of profit or any other commercial damages, including but not limited to special, incidental, consequential, or other damages.

Library of Congress Cataloging-in-Publication Data

Names: Vrochidis, Stefanos, 1975- editor. | Huet, Benoit, editor. | Chang, Edward Y., editor. | Kompatsiaris, Ioannis, editor.

Title: Big Data Analytics for Large-Scale Multimedia Search / Stefanos Vrochidis, Information Technologies Institute, Centre for Research and Technology Hellas, Thessaloniki, Greece; Benoit Huet, EURECOM, Sophia-Antipolis, France; Edward Y. Chang, HTC Research & Healthcare, San Francisco, USA; Ioannis Kompatsiaris, Information Technologies Institute, Centre for Research and Technology Hellas, Thessaloniki, Greece.

Description: Hoboken, NJ, USA : Wiley, [2018] | Includes bibliographical references and index. |

Identifiers: LCCN 2018035613 (print) | LCCN 2018037546 (ebook) | ISBN 9781119376989 (Adobe PDF) | ISBN 9781119377009 (ePub) | ISBN 9781119376972 (hardcover)

Subjects: LCSH: Multimedia data mining. | Big data.

Classification: LCC QA76.9.D343 (ebook) | LCC QA76.9.D343 V76 2018 (print) | DDC 005.7 – dc23

LC record available at <https://lcn.loc.gov/2018035613>

Cover design: Wiley

Cover image: © spainter_vfx/iStock.com

Set in 10/12pt WarnockPro by SPi Global, Chennai, India

Printed and bound by CPI Group (UK) Ltd, Croydon, CR0 4YY

10 9 8 7 6 5 4 3 2 1

Contents

Introduction	<i>xv</i>
List of Contributors	<i>xix</i>
About the Companion Website	<i>xxiii</i>

Part I Feature Extraction from Big Multimedia Data 1

1	Representation Learning on Large and Small Data	3
	<i>Chun-Nan Chou, Chuen-Kai Shie, Fu-Chieh Chang, Jocelyn Chang and Edward Y. Chang</i>	
1.1	Introduction	3
1.2	Representative Deep CNNs	5
1.2.1	AlexNet	6
1.2.1.1	ReLU Nonlinearity	6
1.2.1.2	Data Augmentation	7
1.2.1.3	Dropout	8
1.2.2	Network in Network	8
1.2.2.1	MLP Convolutional Layer	9
1.2.2.2	Global Average Pooling	9
1.2.3	VGG	10
1.2.3.1	Very Small Convolutional Filters	10
1.2.3.2	Multi-scale Training	11
1.2.4	GoogLeNet	11
1.2.4.1	Inception Modules	11
1.2.4.2	Dimension Reduction	12
1.2.5	ResNet	13
1.2.5.1	Residual Learning	13
1.2.5.2	Identity Mapping by Shortcuts	14
1.2.6	Observations and Remarks	15
1.3	Transfer Representation Learning	15
1.3.1	Method Specifications	17
1.3.2	Experimental Results and Discussion	18
1.3.2.1	Results of Transfer Representation Learning for OM	19
1.3.2.2	Results of Transfer Representation Learning for Melanoma	20
1.3.2.3	Qualitative Evaluation: Visualization	21

1.3.3	Observations and Remarks	23
1.4	Conclusions	24
	References	25

2 Concept-Based and Event-Based Video Search in Large Video Collections 31

Foteini Markatopoulou, Damianos Galanopoulos, Christos Tzelepis, Vasileios Mezaris and Ioannis Patras

2.1	Introduction	32
2.2	Video preprocessing and Machine Learning Essentials	33
2.2.1	Video Representation	33
2.2.2	Dimensionality Reduction	34
2.3	Methodology for Concept Detection and Concept-Based Video Search	35
2.3.1	Related Work	35
2.3.2	Cascades for Combining Different Video Representations	37
2.3.2.1	Problem Definition and Search Space	37
2.3.2.2	Problem Solution	38
2.3.3	Multi-Task Learning for Concept Detection and Concept-Based Video Search	40
2.3.4	Exploiting Label Relations	41
2.3.5	Experimental Study	42
2.3.5.1	Dataset and Experimental Setup	42
2.3.5.2	Experimental Results	43
2.3.5.3	Computational Complexity	47
2.4	Methods for Event Detection and Event-Based Video Search	48
2.4.1	Related Work	48
2.4.2	Learning from Positive Examples	49
2.4.3	Learning Solely from Textual Descriptors: Zero-Example Learning	50
2.4.4	Experimental Study	52
2.4.4.1	Dataset and Experimental Setup	52
2.4.4.2	Experimental Results: Learning from Positive Examples	53
2.4.4.3	Experimental Results: Zero-Example Learning	53
2.5	Conclusions	54
2.6	Acknowledgments	55
	References	55

3 Big Data Multimedia Mining: Feature Extraction Facing Volume, Velocity, and Variety 61

Vedhas Pandit, Shahin Amiriparian, Maximilian Schmitt, Amr Mousa and Björn Schuller

3.1	Introduction	61
3.2	Scalability through Parallelization	64
3.2.1	Process Parallelization	64
3.2.2	Data Parallelization	64
3.3	Scalability through Feature Engineering	65
3.3.1	Feature Reduction through Spatial Transformations	66
3.3.2	Laplacian Matrix Representation	66

3.3.3	Parallel latent Dirichlet allocation and bag of words	68
3.4	Deep Learning-Based Feature Learning	68
3.4.1	Adaptability that Conquers both Volume and Velocity	70
3.4.2	Convolutional Neural Networks	72
3.4.3	Recurrent Neural Networks	73
3.4.4	Modular Approach to Scalability	74
3.5	Benchmark Studies	76
3.5.1	Dataset	76
3.5.2	Spectrogram Creation	77
3.5.3	CNN-Based Feature Extraction	77
3.5.4	Structure of the CNNs	78
3.5.5	Process Parallelization	79
3.5.6	Results	80
3.6	Closing Remarks	81
3.7	Acknowledgements	82
	References	82

Part II Learning Algorithms for Large-Scale Multimedia 89

4	Large-Scale Video Understanding with Limited Training Labels	91
	<i>Jingkuan Song, Xu Zhao, Lianli Gao and Liangliang Cao</i>	
4.1	Introduction	91
4.2	Video Retrieval with Hashing	91
4.2.1	Overview	91
4.2.2	Unsupervised Multiple Feature Hashing	93
4.2.2.1	Framework	93
4.2.2.2	The Objective Function of MFH	93
4.2.2.3	Solution of MFH	95
4.2.2.3.1	Complexity Analysis	96
4.2.3	Submodular Video Hashing	97
4.2.3.1	Framework	97
4.2.3.2	Video Pooling	97
4.2.3.3	Submodular Video Hashing	98
4.2.4	Experiments	99
4.2.4.1	Experiment Settings	99
4.2.4.1.1	Video Datasets	99
4.2.4.1.2	Visual Features	99
4.2.4.1.3	Algorithms for Comparison	100
4.2.4.2	Results	100
4.2.4.2.1	CC_WEB_VIDEO	100
4.2.4.2.2	Combined Dataset	100
4.2.4.3	Evaluation of SVH	101
4.2.4.3.1	Results	102
4.3	Graph-Based Model for Video Understanding	103
4.3.1	Overview	103
4.3.2	Optimized Graph Learning for Video Annotation	104

4.3.2.1	Framework	104
4.3.2.2	OGL	104
4.3.2.2.1	Terms and Notations	104
4.3.2.2.2	Optimal Graph-Based SSL	105
4.3.2.2.3	Iterative Optimization	106
4.3.3	Context Association Model for Action Recognition	107
4.3.3.1	Context Memory	108
4.3.4	Graph-based Event Video Summarization	109
4.3.4.1	Framework	109
4.3.4.2	Temporal Alignment	110
4.3.5	TGIF: A New Dataset and Benchmark on Animated GIF Description	111
4.3.5.1	Data Collection	111
4.3.5.2	Data Annotation	112
4.3.6	Experiments	114
4.3.6.1	Experimental Settings	114
4.3.6.1.1	Datasets	114
4.3.6.1.2	Features	114
4.3.6.1.3	Baseline Methods and Evaluation Metrics	114
4.3.6.2	Results	115
4.4	Conclusions and Future Work	116
	References	116
5	Multimodal Fusion of Big Multimedia Data	121
	<i>Ilias Gialampoukidis, Elisavet Chatzilari, Spiros Nikolopoulos, Stefanos Vrochidis and Ioannis Kompatsiaris</i>	
5.1	Multimodal Fusion in Multimedia Retrieval	122
5.1.1	Unsupervised Fusion in Multimedia Retrieval	123
5.1.1.1	Linear and Non-linear Similarity Fusion	123
5.1.1.2	Cross-modal Fusion of Similarities	124
5.1.1.3	Random Walks and Graph-based Fusion	124
5.1.1.4	A Unifying Graph-based Model	126
5.1.2	Partial Least Squares Regression	127
5.1.3	Experimental Comparison	128
5.1.3.1	Dataset Description	128
5.1.3.2	Settings	129
5.1.3.3	Results	129
5.1.4	Late Fusion of Multiple Multimedia Rankings	130
5.1.4.1	Score Fusion	131
5.1.4.2	Rank Fusion	132
5.1.4.2.1	Borda Count Fusion	132
5.1.4.2.2	Reciprocal Rank Fusion	132
5.1.4.2.3	Condorcet Fusion	132
5.2	Multimodal Fusion in Multimedia Classification	132
5.2.1	Related Literature	134
5.2.2	Problem Formulation	136
5.2.3	Probabilistic Fusion in Active Learning	137
5.2.3.1	If $P(S=0 V,T) \neq 0$:	138

5.2.3.2	If $P(S=0 V,T) \neq 0$:	138
5.2.3.3	Incorporating Informativeness in the Selection ($P(S V)$)	139
5.2.3.4	Measuring Oracle's Confidence ($P(S T)$)	139
5.2.3.5	Re-training	140
5.2.4	Experimental Comparison	141
5.2.4.1	Datasets	141
5.2.4.2	Settings	142
5.2.4.3	Results	143
5.2.4.3.1	Expanding with Positive, Negative or Both	143
5.2.4.3.2	Comparing with Sample Selection Approaches	145
5.2.4.3.3	Comparing with Fusion Approaches	147
5.2.4.3.4	Parameter Sensitivity Investigation	147
5.2.4.3.5	Comparing with Existing Methods	148
5.3	Conclusions	151
	References	152
6	Large-Scale Social Multimedia Analysis	157
	<i>Benjamin Bischke, Damian Borth and Andreas Dengel</i>	
6.1	Social Multimedia in Social Media Streams	157
6.1.1	Social Multimedia	157
6.1.2	Social Multimedia Streams	158
6.1.3	Analysis of the Twitter Firehose	160
6.1.3.1	Dataset: Overview	160
6.1.3.2	Linked Resource Analysis	160
6.1.3.3	Image Content Analysis	162
6.1.3.4	Geographic Analysis	164
6.1.3.5	Textual Analysis	166
6.2	Large-Scale Analysis of Social Multimedia	167
6.2.1	Large-Scale Processing of Social Multimedia Analysis	167
6.2.1.1	Batch-Processing Frameworks	167
6.2.1.2	Stream-Processing Frameworks	168
6.2.1.3	Distributed Processing Frameworks	168
6.2.2	Analysis of Social Multimedia	169
6.2.2.1	Analysis of Visual Content	169
6.2.2.2	Analysis of Textual Content	169
6.2.2.3	Analysis of Geographical Content	170
6.2.2.4	Analysis of User Content	170
6.3	Large-Scale Multimedia Opinion Mining System	170
6.3.1	System Overview	171
6.3.2	Implementation Details	171
6.3.2.1	Social Media Data Crawler	171
6.3.2.2	Social Multimedia Analysis	173
6.3.2.3	Analysis of Visual Content	174
6.3.3	Evaluations: Analysis of Visual Content	175
6.3.3.1	Filtering of Synthetic Images	175
6.3.3.2	Near-Duplicate Detection	177
6.4	Conclusion	178
	References	179

7	Privacy and Audiovisual Content: Protecting Users as Big Multimedia Data Grows Bigger	183
	<i>Martha Larson, Jaeyoung Choi, Manel Slokom, Zekeriya Erkin, Gerald Friedland and Arjen P. de Vries</i>	
7.1	Introduction	183
7.1.1	The Dark Side of Big Multimedia Data	184
7.1.2	Defining Multimedia Privacy	184
7.2	Protecting User Privacy	188
7.2.1	What to Protect	188
7.2.2	How to Protect	189
7.2.3	Threat Models	191
7.3	Multimedia Privacy	192
7.3.1	Privacy and Multimedia Big Data	192
7.3.2	Privacy Threats of Multimedia Data	194
7.3.2.1	Audio Data	194
7.3.2.2	Visual Data	195
7.3.2.3	Multimodal Threats	195
7.4	Privacy-Related Multimedia Analysis Research	196
7.4.1	Multimedia Analysis Filters	196
7.4.2	Multimedia Content Masking	198
7.5	The Larger Research Picture	199
7.5.1	Multimedia Security and Trust	199
7.5.2	Data Privacy	200
7.6	Outlook on Multimedia Privacy Challenges	202
7.6.1	Research Challenges	202
7.6.1.1	Multimedia Analysis	202
7.6.1.2	Data	202
7.6.1.3	Users	203
7.6.2	Research Reorientation	204
7.6.2.1	Professional Paranoia	204
7.6.2.2	Privacy as a Priority	204
7.6.2.3	Privacy in Parallel	205
	References	205

Part III Scalability in Multimedia Access 209

8	Data Storage and Management for Big Multimedia	211
	<i>Björn Þór Jónsson, Gylfi Þór Guðmundsson, Laurent Amsaleg and Philippe Bonnet</i>	
8.1	Introduction	211
8.1.1	Multimedia Applications and Scale	212
8.1.2	Big Data Management	213
8.1.3	System Architecture Outline	213
8.1.4	Metadata Storage Architecture	214
8.1.4.1	Lambda Architecture	214
8.1.4.2	Storage Layer	215
8.1.4.3	Processing Layer	216

8.1.4.4	Serving Layer	216
8.1.4.5	Dynamic Data	216
8.1.5	Summary and Chapter Outline	217
8.2	Media Storage	217
8.2.1	Storage Hierarchy	217
8.2.1.1	Secondary Storage	218
8.2.1.2	The Five-Minute Rule	218
8.2.1.3	Emerging Trends for Local Storage	219
8.2.2	Distributed Storage	220
8.2.2.1	Distributed Hash Tables	221
8.2.2.2	The CAP Theorem and the PACELC Formulation	221
8.2.2.3	The Hadoop Distributed File System	221
8.2.2.4	Ceph	222
8.2.3	Discussion	222
8.3	Processing Media	222
8.3.1	Metadata Extraction	223
8.3.2	Batch Processing	223
8.3.2.1	Map-Reduce and Hadoop	224
8.3.2.2	Spark	225
8.3.2.3	Comparison	226
8.3.3	Stream Processing	226
8.4	Multimedia Delivery	226
8.4.1	Distributed In-Memory Buffering	227
8.4.1.1	Memcached and Redis	227
8.4.1.2	Alluxio	227
8.4.1.3	Content Distribution Networks	228
8.4.2	Metadata Retrieval and NoSQL Systems	228
8.4.2.1	Key-Value Stores	229
8.4.2.2	Document Stores	229
8.4.2.3	Wide Column Stores	229
8.4.2.4	Graph Stores	229
8.4.3	Discussion	229
8.5	Case Studies: Facebook	230
8.5.1	Data Popularity: Hot, Warm or Cold	230
8.5.2	Mentions Live	231
8.6	Conclusions and Future Work	231
8.6.1	Acknowledgments	232
	References	232
9	Perceptual Hashing for Large-Scale Multimedia Search	239
	<i>Li Weng, I-Hong Jhuo and Wen-Huang Cheng</i>	
9.1	Introduction	240
9.1.1	Related work	240
9.1.2	Definitions and Properties of Perceptual Hashing	241
9.1.3	Multimedia Search using Perceptual Hashing	243
9.1.4	Applications of Perceptual Hashing	243
9.1.5	Evaluating Perceptual Hash Algorithms	244

9.2	Unsupervised Perceptual Hash Algorithms	245
9.2.1	Spectral Hashing	245
9.2.2	Iterative Quantization	246
9.2.3	K -Means Hashing	247
9.2.4	Kernelized Locality Sensitive Hashing	249
9.3	Supervised Perceptual Hash Algorithms	250
9.3.1	Semi-Supervised Hashing	250
9.3.2	Kernel-Based Supervised Hashing	252
9.3.3	Restricted Boltzmann Machine-Based Hashing	253
9.3.4	Supervised Semantic-Preserving Deep Hashing	255
9.4	Constructing Perceptual Hash Algorithms	257
9.4.1	Two-Step Hashing	257
9.4.2	Hash Bit Selection	258
9.5	Conclusion and Discussion	260
	References	261

Part IV Applications of Large-Scale Multimedia Search 267

10	Image Tagging with Deep Learning: Fine-Grained Visual Analysis	269
	<i>Jianlong Fu and Tao Mei</i>	
10.1	Introduction	269
10.2	Basic Deep Learning Models	270
10.3	Deep Image Tagging for Fine-Grained Image Recognition	272
10.3.1	Attention Proposal Network	274
10.3.2	Classification and Ranking	275
10.3.3	Multi-Scale Joint Representation	276
10.3.4	Implementation Details	276
10.3.5	Experiments on CUB-200-2011	277
10.3.6	Experiments on Stanford Dogs	280
10.4	Deep Image Tagging for Fine-Grained Sentiment Analysis	281
10.4.1	Learning Deep Sentiment Representation	282
10.4.2	Sentiment Analysis	283
10.4.3	Experiments on SentiBank	283
10.5	Conclusion	284
	References	285
11	Visually Exploring Millions of Images using Image Maps and Graphs	289
	<i>Kai Uwe Barthel and Nico Hezel</i>	
11.1	Introduction and Related Work	290
11.2	Algorithms for Image Sorting	293
11.2.1	Self-Organizing Maps	293
11.2.2	Self-Sorting Maps	294
11.2.3	Evolutionary Algorithms	295
11.3	Improving SOMs for Image Sorting	295

11.3.1	Reducing SOM Sorting Complexity	295
11.3.2	Improving SOM Projection Quality	297
11.3.3	Combining SOMs and SSMs	297
11.4	Quality Evaluation of Image Sorting Algorithms	298
11.4.1	Analysis of SOMs	298
11.4.2	Normalized Cross-Correlation	299
11.4.3	A New Image Sorting Quality Evaluation Scheme	299
11.5	2D Sorting Results	301
11.5.1	Image Test Sets	301
11.5.2	Experiments	302
11.6	Demo System for Navigating 2D Image Maps	304
11.7	Graph-Based Image Browsing	306
11.7.1	Generating Semantic Image Features	306
11.7.2	Building the Image Graph	307
11.7.3	Visualizing and Navigating the Graph	310
11.7.4	Prototype for Image Graph Navigation	312
11.8	Conclusion and Future Work	313
	References	313
12	Medical Decision Support Using Increasingly Large Multimodal Data Sets	317
	<i>Henning Müller and Devrim Ünay</i>	
12.1	Introduction	317
12.2	Methodology for Reviewing the Literature in this chapter	320
12.3	Data, Ground Truth, and Scientific Challenges	321
12.3.1	Data Annotation and Ground Truthing	321
12.3.2	Scientific Challenges and Evaluation as a Service	321
12.3.3	Other Medical Data Resources Available	322
12.4	Techniques used for Multimodal Medical Decision Support	323
12.4.1	Visual and Non-Visual Features Describing the Image Content	323
12.4.2	General Machine Learning and Deep Learning	323
12.5	Application Types of Image-Based Decision Support	326
12.5.1	Localization	326
12.5.2	Segmentation	326
12.5.3	Classification	327
12.5.4	Prediction	327
12.5.5	Retrieval	327
12.5.6	Automatic Image Annotation	328
12.5.7	Other Application Types	328
12.6	Discussion on Multimodal Medical Decision Support	328
12.7	Outlook or the Next Steps of Multimodal Medical Decision Support	329
	References	330

Conclusions and Future Trends 337

Index 339

Introduction

In recent years, the rapid development of digital technologies, including the low cost of recording, processing, and storing media, and the growth of high-speed communication networks enabling large-scale content sharing, has led to a rapid increase in the availability of multimedia content worldwide. The availability of such content, as well as the increasing user need of analysing and searching into large multimedia collections, increases the demand for the development of advanced search and analytics techniques for big multimedia data. Although multimedia is defined as a combination of different media (e.g., audio, text, video, images etc.) this book mainly focuses on textual, visual, and audiovisual content, which are considered the most characteristic types of multimedia.

In this context, the big multimedia data era brings a plethora of challenges to the fields of multimedia mining, analysis, searching, and presentation. These are best described by the Vs of big data: volume, variety, velocity, veracity, variability, value, and visualization. A modern multimedia search and analytics algorithm and/or system has to be able to handle large databases with varying formats at extreme speed, while having to cope with unreliable “ground truth” information and “noisy” conditions. In addition, multimedia analysis and content understanding algorithms based on machine learning and artificial intelligence have to be employed. Further, the interpretation of the content over time may change, leading to a “drifting target” with multimedia content being perceived differently in different times with often low value of data points. Finally, the assessed information needs to be presented in comprehensive and transparent ways to human users.

The main challenges for big multimedia data analytics and search are identified in the areas of:

- multimedia representation by extracting low- and high-level conceptual features
- application of machine learning and artificial intelligence for large-scale multimedia
- scalability in multimedia access and retrieval.

Feature extraction is an essential step in any computer vision and multimedia data analysis task. Though progress has been made in past decades, it is still quite difficult for computers to accurately recognize an object or comprehend the semantics of an image or a video. Thus, feature extraction is expected to remain an active research area in advancing computer vision and multimedia data analysis for the foreseeable

future. The traditional approach of feature extraction is model-based in that researchers engineer useful features based on heuristics, and then conduct validations via empirical studies. A major shortcoming of the model-based approach is that exceptional circumstances such as different lighting conditions and unexpected environmental factors can render the engineered features ineffective. The data-driven approach complements the model-based approach. Instead of human-engineered features, the data-driven approach learns representation from data. In principle, the greater the quantity and diversity of data, the better the representation can be learned.

An additional layer of analysis and automatic annotation of big multimedia data involves the extraction of high-level concepts and events. Concept-based multimedia data indexing refers to the automatic annotation of multimedia fragments with specific simple labels, e.g., “car”, “sky”, “running” etc., from large-scale collections. In this book we mainly deal with video as a characteristic multimedia example for concept-based indexing. To deal with this task, concept detection methods have been developed that automatically annotate images and videos with semantic labels referred to as concepts. A recent trend in video concept detection is to learn features directly from the raw keyframe pixels using deep convolutional neural networks (DCNNs). On the other hand, event-based video indexing aims to represent video fragments with high-level events in a given set of videos. Typically, events are more complex than concepts, i.e., they may include complex activities, occurring at specific places and times, and involving people interacting with other people and/or object(s), such as “opening a door”, “making a cake”, etc. The event detection problem in images and videos can be addressed either with a typical video event detection framework, including feature extraction and classification, and/or by effectively combining textual and visual analysis techniques.

When it comes to multimedia analysis, machine learning is considered to be one of the most popular techniques that can be applied. These include CNN for representation learning such as imagery and acoustic data, as well as recurrent neural networks for series data, e.g., speech and video. The challenge of video understanding lies in the gap between large-scale video data and the limited resource we can afford in both label collection and online computing stages.

An additional step in the analysis and retrieval of large-scale multimedia is the fusion of heterogeneous content. Due to the diverse modalities that form a multimedia item (e.g., visual, textual modality), multiple features are available to represent each modality. The fusion of multiple modalities may take place at the feature level (early fusion) or the decision level (late fusion). Early fusion techniques usually rely on the linear (weighted) combination of multimodal features, while lately non-linear fusion approaches have prevailed. Another fusion strategy relies on graph-based techniques, allowing the construction of random walks, generalized diffusion processes, and cross-media transitions on the formulated graph of multimedia items. In the case of late fusion, the fusion takes place at the decision level and can be based on (i) linear/non-linear combinations of the decisions from each modality, (ii) voting schemes, and (iii) rank diffusion processes. Scalability issues in multimedia processing systems typically occur for two reasons: (i) the lack of labelled data, which limits the scalability with respect to the number of supported concepts, and (ii) the high computational overload in terms of both processing time and memory complexity. For the first problem, methods that learn primarily on weakly labelled data (weakly supervised learning, semi-supervised learning)

have been proposed. For the second problem, methodologies typically rely on reducing the data space they work on by using smartly-selected subsets of the data so that the computational requirements of the systems are optimized.

Another important aspect of multimedia nowadays is the social dimension and the user interaction that is associated with the data. The internet is abundant with opinions, sentiments, and reflections of the society about products, brands, and institutions hidden under large amounts of heterogeneous and unstructured data. Such analysis includes the contextual augmentation of events in social media streams in order to fully leverage the knowledge present in social media, taking into account temporal, visual, textual, geographical, and user-specific dimensions. In addition, the social dimension includes an important privacy aspect. As big multimedia data continues to grow, it is essential to understand the risks for users during online multimedia sharing and multimedia privacy. Specifically, as multimedia data gets bigger, automatic privacy attacks can become increasingly dangerous. Two classes of algorithms for privacy protection in a large-scale online multimedia sharing environment are involved. The first class is based on multimedia analysis, and includes classification approaches that are used as filters, while the second class is based on obfuscation techniques.

The challenge of data storage is also very important for big multimedia data. At this scale, data storage, management, and processing become very challenging. At the same time, there has been a proliferation of big data management techniques and tools, which have been developed mostly in the context of much simpler business and logging data. These tools and techniques include a variety of noSQL and newSQL data management systems, as well as automatically distributed computing frameworks (e.g., Hadoop and Spark). The question is which of these big data techniques apply to today's big multimedia collections. The answer is not trivial since the big data repository has to store a variety of multimedia data, including raw data (images, video or audio), meta-data (including social interaction data) associated with the multimedia items, derived data, such as low-level concepts and semantic features extracted from the raw data, and supplementary data structures, such as high-dimensional indices or inverted indices. In addition, the big data repository must serve a variety of parallel requests with different workloads, ranging from simple queries to detailed data-mining processes, and with a variety of performance requirements, ranging from response-time driven online applications to throughput-driven offline services. Although several different techniques have been developed there is no single technology that can cover all the requirements of big multimedia applications.

Finally, the book discusses the two main challenges of large-scale multimedia search: accuracy and scalability. Conventional techniques typically focus on the former. However, recently attention has mainly been paid to the latter, since the amount of multimedia data is rapidly increasing. Due to the curse of dimensionality, conventional feature representations of high dimensionality are not in favour of fast search. The big data era requires new solutions for multimedia indexing and retrieval based on efficient hashing. One of the robust solutions is perceptual hash algorithms, which are used for generating hash values from multimedia objects in big data collections, such as images, audio, and video. A content-based multimedia search can be achieved by comparing hash values. The main advantages of using hash values instead of other content representations is that hash values are compact and facilitate fast in-memory indexing and search, which is very important for large-scale multimedia search.

Given the aforementioned challenges, the book is organized in the following chapters. Chapters 1, 2, and 3 deal with feature extraction from big multimedia data, while Chapters 4, 5, 6, and 7 discuss techniques relevant to machine learning for multimedia analysis and fusion. Chapters 8, and 9 deal with scalability in multimedia access and retrieval, while Chapters 10, 11, and 12 present applications of large-scale multimedia retrieval. Finally, we conclude the book by summarizing and presenting future trends and challenges.

List of Contributors

Laurent Amsaleg

Univ Rennes, Inria, CNRS
IRISA
France

Shahin Amiriparian

ZD.B Chair of Embedded Intelligence for
Health Care and Wellbeing
University of Augsburg
Germany

Kai Uwe Barthel

Visual Computing Group
HTW Berlin
University of Applied Sciences
Berlin
Germany

Benjamin Bischke

German Research Center for Artificial
Intelligence and TU Kaiserslautern
Germany

Philippe Bonnet

IT University of Copenhagen
Copenhagen
Denmark

Damian Borth

University of St. Gallen
Switzerland

Edward Y. Chang

HTC Research & Healthcare
San Francisco, USA

Elisavet Chatzilari

Information Technologies Institute
Centre for Research and Technology
Hellas
Thessaloniki
Greece

Liangliang Cao

College of Information and Computer
Sciences
University of Massachusetts Amherst
USA

Chun-Nan Chou

HTC Research & Healthcare
San Francisco, USA

Jaeyoung Choi

Delft University of Technology
Netherlands
and
International Computer Science Institute
USA

Fu-Chieh Chang

HTC Research & Healthcare
San Francisco, USA

Jocelyn Chang

Johns Hopkins University
Baltimore
USA

Wen-Huang Cheng

Department of Electronics Engineering
and Institute of Electronics
National Chiao Tung University
Taiwan

Andreas Dengel

German Research Center for Artificial
Intelligence and TU Kaiserslautern
Germany

Arjen P. de Vries

Radboud University
Nijmegen
The Netherlands

Zekeriya Erkin

Delft University of Technology and
Radboud University
The Netherlands

Gerald Friedland

University of California
Berkeley
USA

Jianlong Fu

Multimedia Search and Mining Group
Microsoft Research Asia
Beijing
China

Damianos Galanopoulos

Information Technologies Institute
Centre for Research and Technology
Hellas
Thessaloniki
Greece

Lianli Gao

School of Computer Science and Center
for Future Media
University of Electronic Science and
Technology of China
Sichuan
China

Ilias Gialampoukidis

Information Technologies Institute
Centre for Research and Technology
Hellas
Thessaloniki
Greece

Gylfi Þór Guðmundsson

Reykjavik University
Iceland

Nico Hezel

Visual Computing Group
HTW Berlin
University of Applied Sciences
Berlin
Germany

I-Hong Jhuo

Center for Open-Source Data & AI
Technologies
San Francisco
California

Björn Þór Jónsson

IT University of Copenhagen
Denmark

and

Reykjavik University
Iceland

Ioannis Kompatsiaris

Information Technologies Institute
Centre for Research and Technology
Hellas
Thessaloniki
Greece

Martha Larson

Radboud University and
Delft University of Technology
The Netherlands

Amr Mousa

Chair of Complex and Intelligent Systems
University of Passau
Germany

Foteini Markatopoulou

Information Technologies Institute
Centre for Research and Technology
Hellas
Thessaloniki
Greece

and

School of Electronic Engineering and
Computer Science
Queen Mary University of London
United Kingdom

Henning Müller

University of Applied Sciences Western
Switzerland (HES-SO)
Sierre
Switzerland

Tao Mei

JD AI Research
China

Vasileios Mezaris

Information Technologies Institute
Centre for Research and Technology
Hellas
Thessaloniki
Greece

Spiros Nikolopoulos

Information Technologies Institute
Centre for Research and Technology
Hellas
Thessaloniki
Greece

Ioannis Patras

School of Electronic Engineering and
Computer Science
Queen Mary University of London
United Kingdom

Vedhas Pandit

ZD.B Chair of Embedded Intelligence for
Health Care and Wellbeing
University of Augsburg
Germany

Maximilian Schmitt

ZD.B Chair of Embedded Intelligence for
Health Care and Wellbeing
University of Augsburg
Germany

Björn Schuller

ZD.B Chair of Embedded Intelligence for
Health Care and Wellbeing
University of Augsburg
Germany

and

GLAM - Group on Language, Audio and
Music
Imperial College London
United Kingdom

Chuen-Kai Shie

HTC Research & Healthcare
San Francisco, USA

Manel Slokom

Delft University of Technology
The Netherlands

Jingkuan Song

School of Computer Science and Center
for Future Media
University of Electronic Science and
Technology of China
Sichuan
China

Christos Tzelepis

Information Technologies Institute
Centre for Research and Technology
Hellas
Thessaloniki
Greece
and

School of Electronic Engineering and
Computer Science
QMUL, UK

Devrim Ünay

Department of Biomedical Engineering
Izmir University of Economics
Izmir
Turkey

Stefanos Vrochidis

Information Technologies Institute
Centre for Research and Technology
Hellas
Thessaloniki
Greece

Li Weng

Hangzhou Dianzi University
China
and
French Mapping Agency (IGN)
Saint-Mande
France

Xu Zhao

Department of Automation
Shanghai Jiao Tong University
China

About the Companion Website

This book is accompanied by a companion website:

www.wiley.com/go/vrochidis/bigdata



The website includes:

- Open source algorithms
- Data sets
- Tools materials for demonstration purpose

Scan this QR code to visit the companion website.



Part I

Feature Extraction from Big Multimedia Data

1

Representation Learning on Large and Small Data

Chun-Nan Chou, Chuen-Kai Shie, Fu-Chieh Chang, Jocelyn Chang and Edward Y. Chang

1.1 Introduction

Extracting useful features from a scene is an essential step in any computer vision and multimedia data analysis task. Though progress has been made in past decades, it is still quite difficult for computers to comprehensively and accurately recognize an object or pinpoint the more complicated semantics of an image or a video. Thus, feature extraction is expected to remain an active research area in advancing computer vision and multimedia data analysis for the foreseeable future.

The approaches in feature extraction can be divided into two categories: *model-centric* and *data-driven*. The model-centric approach relies on human heuristics to develop a computer model (or algorithm) to extract features from an image. (We use imagery data as our example throughout this chapter.) Some widely used models are Gabor filter, wavelets, and scale-invariant feature transform (SIFT) [1]. These models were engineered by scientists and then validated via empirical studies. A major shortcoming of the model-centric approach is that unusual circumstances that a model does not take into consideration during its design, such as different lighting conditions and unexpected environmental factors, can render the engineered features less effective. In contrast to the model-centric approach, which dictates representations independent of data, the data-driven approach learns representations from data [2]. Examples of data-driven algorithms are multilayer perceptron (MLP) and convolutional neural networks (CNNs), which belong to the general category of neural networks and deep learning [3, 4].

Both model-centric and data-driven approaches employ a model (algorithm or machine). The differences between model-centric and data-driven can be described in two related aspects:

- Can data affect model parameters? With model-centric, training data does not affect the model. With data-driven, such as MLP or CNN, their internal parameters are changed/learned based on the discovered structure in large data sets [5].
- Can more data help improve representations? Whereas more data can help a data-driven approach to improve representations, more data cannot change the

features extracted by a model-centric approach. For example, the features of an image can be affected by the other images in the CNN (because the structure parameters modified through back-propagation are affected by all training images), but the feature set of an image is invariant of the other images in a model-centric pipeline such as SIFT.

The greater the quantity and diversity of data, the better the representations can be learned by a data-driven pipeline. In other words, if a learning algorithm has seen enough training instances of an object under various conditions, e.g., in different postures, and has been partially occluded, then the features learned from the training data will be more comprehensive.

The focus of this chapter is on how *neural networks*, specifically CNNs, achieve effective representation learning. Neural networks, a kind of neuroscience-motivated models, were based on Hubel and Wiesel's research on cats' visual cortex [6], and subsequently formulated into computation models by scientists in the early 1980s. Pioneer neural network models include Neocognitron [7] and the shift-invariant neural network [8]. Widely cited enhanced models include LeNet-5 [9] and Boltzmann machines [10]. However, the popularity of neural networks surged only in 2012 after large training data sets became available. In 2012, Krizhevsky [11] applied deep convolutional networks on the ImageNet dataset¹, and their AlexNet achieved breakthrough accuracy in the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) 2012 competition.² This work convinced the research community and related industries that representation learning with big data is promising. Subsequently, several efforts have aimed to further improve the learning capability of neural networks. Today, the top-5 error rate³ for the ILSVRC competition has dropped to 3.57%, a remarkable achievement considering the error rate was 26.2% before AlexNet [11] was proposed.

We divide the remainder of this chapter into two parts before suggesting related reading in the concluding remarks. The first part reviews representative CNN models proposed since 2012. These key representatives are discussed in terms of three aspects addressed in He's tutorial presentation [14] at ICML 2016: (i) representation ability, (ii) optimization ability, and (iii) generalization ability. The representation ability is the ability of a CNN to learn/capture representations from training data assuming the optimum could be found. Here, the optimum refers to attaining the best solution of the underlying learning algorithm, modeled as an optimization problem. This leads to the second aspect that He's tutorial addresses: the optimization ability. The optimization ability is the feasibility of finding an optimum. Specifically on CNNs, the optimization problem is to find the optimal solution of the stochastic gradient descent. Finally, the generalization ability is the quality of the test performance once model parameters have been learned from training data.

The second part of this chapter deals with the small data problem. We present how features learned from one source domain with big data can be transferred to a different target domain with small data. This transfer representation learning approach is critical

1 ImageNet is a dataset of over 15 million labeled images belonging to roughly 22,000 categories [12].

2 The ILSVRC [13] evaluates algorithms for object detection and image classification on a subset of ImageNet, 1.2 million images over 1000 categories. Throughout this chapter, we focus on discussing image classification challenges.

3 The top-5 error used to evaluate the performance of image classification is the proportion of images such that the ground-truth category is outside the top-5 predicted categories.

for remedying the small data challenge often encountered in the medical domain. We use the Otitis Media detector, designed and developed for our XPRIZE Tricorder [15] device (code name DeepQ), to demonstrate how learning on a small dataset can be bolstered by transferring over learned representations from ImageNet, a dataset that is entirely irrelevant to otitis media.

1.2 Representative Deep CNNs

Deep learning has its roots in neuroscience. Strongly driven by the fact that the human visual system can effortlessly recognize objects, neuroscientists have been developing vision models based on physiological evidence that can be applied to computers. Though such research may still be in its infancy and several hypotheses remain to be validated, some widely accepted theories have been established. Building on the pioneering neuroscience work of Hubel [6], all recent models are founded on the theory that visual information is transmitted from the primary visual cortex (V1) over extrastriate visual areas (V2 and V4) to the inferotemporal cortex (IT). The IT in turn is a major source of input to the prefrontal cortex (PFC), which is involved in linking perception to memory and action [16].

The pathway from V1 to the IT, called the *ventral visual pathway* [17], consists of a number of simple and complex layers. The lower layers detect simple features (e.g., oriented lines) at the pixel level. The higher layers aggregate the responses of these simple features to detect complex features at the object-part level. Pattern reading at the lower layers is unsupervised, whereas recognition at the higher layers involves supervised learning. Pioneer computational models developed based on the scientific evidence include Neocognitron [7] and the shift-invariant neural network [8]. Widely cited enhanced models include LeNet-5 [9] and Boltzmann machines [10]. The remainder of this chapter uses representative CNN models, which stem from LeNet-5 [9], to present three design aspects: representation, optimization, and generalization.

CNNs are composed of two major components: *feature extraction* and *classification*. For feature extraction, a standard structure consists of stacked convolutional layers, which are followed by optional layers of contrast normalization or pooling. For classification, there are two widely used structures. One structure employs one or more fully connected layers. The other structure uses a global average pooling layer, which is illustrated in section 1.2.2.2.

The accuracy of several computer vision tasks, such as house number recognition [18], traffic sign recognition [19], and face recognition [20], has been substantially improved recently, thanks to advances in CNNs. For many similar object-recognition tasks, the advantage of CNNs over other methods is that CNNs join classification with feature extraction. Several works, such as [21], show that CNNs can learn superior representations to boost the performance of classification. Table 1.1 presents four top-performing CNN models proposed over the past four years and their performance statistics in the top-5 error rate. These representative models mainly differ in their number of layers or parameters. (Parameters refer to the learnable variables by supervised training including weight and bias parameters of the CNN models.) Besides the four CNN models depicted in Table 1.1, Lin et al. [22] proposed network in network (NIN), which has considerably influenced subsequent models such as GoogLeNet, Visual Geometry Group (VGG), and

Table 1.1 Image classification performance on the ImageNet subset designated for ILSVRC [13].

Model	Year	Rank	Error (top-5)	Number of parameter layers	Number of parameters in a single model
AlexNet [11]	2012	1st	16.42%	8	60m
VGG [23]	2014	2nd	7.32%	19	144m
GoogLeNet [24]	2014	1st	6.67%	22	5m
ResNet [25]	2015	1st	3.57%	152	60m

ResNet. In the following sections, we present these five models' novel ideas and key techniques, which have had significant impacts on designing subsequent CNN models.

1.2.1 AlexNet

Krizhevsky [11] proposed AlexNet, which was the winner of the ILSVRC-2012 competition and outperformed the runner-up significantly (top-5 error rate of 16% in comparison with 26%). The outstanding performance of AlexNet led to increased prevalence of CNNs in the computer vision field. AlexNet achieved this breakthrough performance by combining several novel ideas and effective techniques. Based on He's three aspects of deep learning models [14], these novel ideas and effective techniques can be categorized as follows:

- 1) *Representation ability*. In contrast to prior CNN models such as LetNet-5 [9], AlexNet was deeper and wider in the sense that both the number of parameter layers and the number of parameters are larger than those of its predecessors.
- 2) *Optimization ability*. AlexNet utilized a non-saturating activation function, the rectified linear unit (ReLU) function, to make training faster.
- 3) *Generalization ability*. AlexNet employed two effective techniques, data augmentation and dropout, to alleviate overfitting.

AlexNet's three key ingredients according to the description in [11] are ReLU nonlinearity, data augmentation, and dropout.

1.2.1.1 ReLU Nonlinearity

In order to model nonlinearity, the neural network introduces the activation function during the evaluation of neuron outputs. The traditional way to evaluate a neuron output f as a function g of its input z is with $f = g(z)$ where g can be a sigmoid function $g(z) = (1 + e^{-z})^{-1}$ or a hyperbolic tangent function $g(z) = \tanh(z)$. Both of these functions are saturating nonlinear. That is, the ranges of these two functions are fixed between a minimum value and a maximum value.

Instead of using saturating activation functions, however, AlexNet adopted the nonsaturating activation function ReLU proposed in [26]. ReLU computes the function $g(z) = \max(0, z)$, which has a threshold at zero. Using ReLU enjoys two benefits. First, ReLU requires less computation in comparison with sigmoid and hyperbolic

tangent functions, which involve expensive exponential operations. The other benefit is that ReLU, in comparison to sigmoid and hyperbolic tangent functions, is found to accelerate the convergence of stochastic gradient descent (SGD). As demonstrated in the first figure of [11], a CNN with ReLU is six times faster to train than that with a hyperbolic tangent function. Due to these two advantages, recent CNN models have adopted ReLU as their activation functions.

1.2.1.2 Data Augmentation

As shown in Table 1.1, the AlexNet architecture has 60 million parameters. This huge number of parameters makes overfitting highly possible if training data is not sufficient. To combat overfitting, AlexNet incorporates two schemes: data augmentation and dropout.

Thanks to ImageNet, AlexNet is the first model that enjoys *big data* and takes advantage of benefits from the data-driven feature learning approach advocated by [2]. However, even the 1.2 million ImageNet labeled instances are still considered insufficient given that the number of parameters is 60 million. (From simple algebra, 1.2 million equations are insufficient for solving 60 million variables.) Conventionally, when the training dataset is limited, the common practice in image data is to artificially enlarge the dataset by using label-preserving transformations [27–29]. In order to enlarge the training data, AlexNet employs two distinct forms of data augmentation, both of which can produce the transformed images from the original images with very little computation [11, 30].

The first scheme of data augmentation includes a random cropping function and horizontal reflection function. Data augmentation can be applied to both the training and testing stages. For the training stage, AlexNet randomly extracts smaller image patches (224×224) and their horizontal reflections from the original images (256×256). The AlexNet model is trained on these extracted patches instead of the original images in the ImageNet dataset. In theory, this scheme is capable of increasing the training data by a factor of $(256 - 224) \times (256 - 224) \times 2 = 2048$. Although the resultant training examples are highly interdependent, Krizhevsky [11] claimed that without this data augmentation scheme the AlexNet model would suffer from substantial overfitting. (This is evident from our algebra example.) For the testing stage, AlexNet generated ten patches, including four corner patches, one center patch, and each of the five patches' horizontal reflections from test images. Based on the generated ten patches, AlexNet first derived temporary results from the network's softmax layer and then made a prediction by averaging the ten results.

The second scheme of data augmentation alters the intensities of the RGB channels in training images by using principal component analysis (PCA). This scheme is used to capture an important property of natural images: the invariance of object identity to changes in the intensity and color of the illumination. The detailed implementation is as follows. First, the principal components of RGB pixel values are acquired by performing PCA on a set of RGB pixel values throughout the ImageNet training set. When a particular training image is chosen to train the network, each RGB pixel $I_{xy} = [I_{xy}^R, I_{xy}^G, I_{xy}^B]^T$ of this chosen training image is refined by adding the following quantity:

$$[\beta_1, \beta_2, \beta_3][\alpha_1 \lambda_1, \alpha_2 \lambda_2, \alpha_3 \lambda_3]^T,$$

where β_i and λ_i represent the i th eigenvector and the eigenvalue of the 3×3 covariance matrix of RGB pixel values, respectively, and α_i is a random variable drawn from a Gaussian model with mean zero and standard deviation 0.1. Note that each time one training image is chosen to train the network, each α_i is redrawn. Thus, during the training, α_i of data augmentation varies with different times for the same training image. Once α_i is drawn, α_i is applied to all the pixels of this chosen training image.

1.2.1.3 Dropout

Model ensembles such as *bagging* [31], *boosting* [32], and *random forest* [33] have long been shown to effectively reduce class-prediction variance and hence testing error. Model ensembles rely on combining the predictions from several different models. However, this method is impractical for large-scale CNNs such as AlexNet, since training even one CNN can take several days or even weeks.

Rather than training multiple large CNNs, Krizhevsky [11] employed the “dropout” technique introduced in [34] to efficiently perform model combination. This technique simply sets the output of each hidden neuron to zero with a probability p (e.g., 0.5 in [11]). Afterwards, the dropped-out neurons neither contribute to the forward pass nor participate in the subsequent back-propagation pass. In this manner, different network architectures are sampled when each training instance is presented, but all these sampled architectures share the same parameters. In addition to combining models efficiently, the dropout technique has the effect of reducing the complex co-adaptations of neurons, since a neuron cannot depend on the presence of other neurons. In this way, more robust features are forcibly learned. At the time of testing, all neurons are used, but their outputs are multiplied by p , which is a reasonable approximation of the geometric mean of the predictive distributions produced by the exponential quantity of dropout networks [34].

In [11], dropout was only applied to the first two fully connected layers of AlexNet and roughly doubled the number of iterations required for convergence. Krizhevsky [11] also claimed that AlexNet suffered from substantial overfitting without dropout.

1.2.2 Network in Network

Although NIN, presented in [22], has not ranked among the best of ILSVRC competitions in recent years, its novel designs have significantly influenced subsequent CNN models, especially its 1×1 convolutional filters. The 1×1 convolutional filters are widely used by current CNN models and have been incorporated into VGG, GoogLeNet, and ResNet. Based on He’s three aspects of learning deep models, the novel designs proposed in NIN can be categorized as follows:

- 1) *Representation ability*. In order to enhance the model’s discriminability, NIN adopted MLP convolutional layers with more complex structures to abstract the data within the receptive field.
- 2) *Optimization ability*. Optimization in NIN remained typical compared to that of the other models.
- 3) *Generalization ability*. NIN utilized global average pooling over feature maps in the classification layer because global average pooling is less prone to overfitting than traditional fully connected layers.

1.2.2.1 MLP Convolutional Layer

The work of Lin et al. [22] argued that the conventional CNNs [9] implicitly make the assumption that the samples of the latent concepts are linearly separable. Thus, typical convolutional layers generate feature maps with linear convolutional filters followed by nonlinear activation functions. This kind of feature map can be calculated as follows:

$$f_{i,j,k} = g(z_{i,j,k}), \text{ where } z_{i,j,k} = w_k^T x_{i,j} + b_k. \quad (1.1)$$

Here, (i, j) is the pixel index, and k is the filter index. Parameter $x_{i,j}$ stands for the input patch centered at location (i, j) . Parameters w_k and b_k represent the weight and bias parameters of the k th filter, respectively. Parameter z denotes the result of the convolutional layer and the input to the activation function, while g denotes the activation function, which can be a sigmoid $g(z) = (1 + e^{-z})^{-1}$, hyperbolic tangent $g(z) = \tanh(z)$, or ReLU $g(z) = \max(z, 0)$.

However, instances of the same concept often live on a nonlinear manifold. Hence, the representations that capture these concepts are generally highly nonlinear functions of the input. In NIN, the linear convolutional filter is replaced with an MLP. This new type of layer is called `mlpconv` in [22], where MLP convolves over the input. There are two reasons for choosing an MLP. First, an MLP is a general nonlinear function approximator. Second, an MLP can be trained by using back-propagation, and is therefore compatible with conventional CNN models. The first figure in [22] depicts the difference between a linear convolutional layer and an `mlpconv` layer. The calculation for an `mlpconv` layer is performed as follows:

$$\begin{aligned} f_{i,j,k_1}^1 &= g(z_{i,j,k_1}^1), \text{ where } z_{i,j,k_1}^1 = w_{k_1}^1 T x_{i,j} + b_{k_1}^1 \\ f_{i,j,k_2}^2 &= g(z_{i,j,k_2}^2), \text{ where } z_{i,j,k_2}^2 = w_{k_2}^2 T f_{i,j}^1 + b_{k_2}^2 \\ &\vdots \\ f_{i,j,k_n}^n &= g(z_{i,j,k_n}^n), \text{ where } z_{i,j,k_n}^n = w_{k_n}^n T f_{i,j}^{n-1} + b_{k_n}^n. \end{aligned} \quad (1.2)$$

Here, n is the number of layers in the MLP, and k_i is the filter index of the i th layer. Lin et al. [22] used ReLU as the activation function in the MLP.

From a pooling point of view, Eq. 1.2 is equivalent to performing cross-channel parametric pooling on a typical convolutional layer. Traditionally, there is no learnable parameter involved in the pooling operation. Besides, conventional pooling is performed within one particular feature map, and is thus not cross-channel. However, Eq. 1.2 performs a weighted linear recombination on the input feature maps, which then goes through a nonlinear activation function, therefore Lin et al. [22] interpreted Eq. 1.2 as a cross-channel parametric pooling operation. They also suggested that we can view Eq. 1.2 as a convolutional layer with a 1×1 filter.

1.2.2.2 Global Average Pooling

Lin [22] made the following remarks. The traditional CNN adopts the fully connected layers for classification. Specifically, the feature maps of the last convolutional layer are flattened into a vector, and this vector is fed into some fully connected layers followed by a softmax layer [11, 35, 36]. In this fashion, convolutional layers are treated as feature extractors, using traditional neural networks to classify the resulting features.

However, the traditional neural networks are prone to overfitting, thereby degrading the generalization ability of the overall network.

Instead of using the fully connected layers with regularization methods such as dropout, Lin [22] proposed global average pooling to replace the traditional fully connected layers in CNNs. Their idea was to derive one feature map from the last *mlpconv* layer for each corresponding category of the classification task. The values of each derived feature map would be averaged spatially, and all the average values would be flattened into a vector which would then be fed directly into the softmax layer. The second figure in [22] delineates the design of global average pooling. One advantage of global average pooling over fully connected layers is that there is no parameter to optimize in global average pooling, preventing overfitting at this layer. Another advantage is that the linkage between feature maps of the last convolutional layer and categories of classification can be easily interpreted, which allows for better understanding. Finally, global average pooling aggregates spatial information and thus offers more robust spatial translations of the input.

1.2.3 VGG

VGG, proposed by Simonyan and Zisserman [23], ranked first and second in the localization and classification tracks of the ImageNet Challenge 2014, respectively. VGG reduced the top-5 error rate of AlexNet from 16.42% to 7.32%, which is an improvement of more than 50%. Using very small (3×3) convolutional filters makes a substantial contribution to this improvement. Consequently, very small (3×3) convolutional filters have been very popular in recent CNN models. Here, the convolutional filter is small or large, depending on the size of its receptive field. According to He's three aspects of learning deep models, the essential ideas in VGG can be depicted as follows:

- 1) *Representation ability*. VGG used very small (3×3) convolutional filters, which makes the decision function more discriminative. Additionally, the depth of VGG was increased steadily to 19 parameter layers by adding more convolutional layers, an increase that is feasible due to the use of very small (3×3) convolutional filters in all layers.
- 2) *Optimization ability*. VGG used very small (3×3) convolutional filters, thereby decreasing the number of parameters.
- 3) *Generalization ability*. VGG employed training to recognize objects over a wide range of scales.

1.2.3.1 Very Small Convolutional Filters

According to [23], instead of using relatively large convolutional filters in the first convolutional layers (e.g., 11×11 with stride 4 in [11] or 7×7 with stride 2 in [21, 37]), VGG used very small 3×3 convolutional filters with stride 1 throughout the network. The output dimension of a stack of two 3×3 convolutional filters (without spatial pooling operation in between) is equal to the output dimension of one 5×5 convolutional filter. Thus, [23] claimed that a stack of two 3×3 convolutional filters has an effective receptive field of 5×5 . By following the same rule, we can conclude that three such filters construct a 7×7 effective receptive field.

The reasons for using smaller convolutional filters are twofold. First, the decision function is more discriminative. For example, using a stack of three 3×3 convolutional filters

instead of a single 7×7 convolutional filter can incorporate three nonlinear activation functions instead of using just one. Second, the number of parameters can be decreased. Assuming that the input as well as output feature maps have C channels, we can use our prior example as an illustration of decreased parameter number. The stack of three 3×3 convolutional filters is parametrized by $3(3^2C^2) = 27C^2$ weight parameters. On the other hand, a single 7×7 convolutional filter requires $7^2C^2 = 49C^2$ weight parameters, which is 81% more than that of three 3×3 filters. Simonyan and Zisserman [23] argued that we can view the usage of very small convolutional filters as imposing a regularization on the 7×7 convolutional filters and forcing them to have a decomposition through 3×3 filters (with nonlinearity injected in between).

1.2.3.2 Multi-scale Training

Simonyan and Zisserman [23] considered two approaches for setting the training scale to S . The first approach is to fix S , which corresponds to single-scale training. Single-scale training has been widely used in prior art [11, 21, 37]. However, objects in images can be of different sizes, and it is beneficial to take objects of different sizes into account during the training phase. Thus, the second approach proposed in VGG for setting to S is multi-scale training. In multi-scale training, each training image is individually rescaled by randomly sampling S from a certain range $[S_{min}, S_{max}]$. In VGG, S_{min} and S_{max} were set to 256 and 512, respectively. Simonyan and Zisserman [23] also interpreted this multi-scale training as a sort of data augmentation of the training set with scale jittering, where a single model is trained to recognize objects over a wide range of scales.

1.2.4 GoogLeNet

GoogLeNet, devised by Szegedy [24], held the record for classification and detection of ILSVRC 2014. GoogLeNet reached a top-5 error rate of 6.67%, which is better than that of VGG with 7.32% in the same year. This improvement is mainly attributed to the proposed “Inception module.” The essential ideas of GoogLeNet can be categorized as follows:

- 1) *Representation ability.* GoogLeNet increased the depth and width of the network while keeping the computational budget constant. Here, the depth and width of the network represent the number of network layers and the number of neurons at each layer, respectively.
- 2) *Optimization ability.* GoogLeNet improved utilization of computing resources inside the network through dimension reduction, thereby easing the training of networks.
- 3) *Generalization ability.* Given the number of labeled examples in the training set is the same, GoogLeNet utilized dimension reduction to decrease the number of parameters dramatically and was hence less prone to overfitting.

1.2.4.1 Inception Modules

The main idea of GoogLeNet is to consider how an optimal local sparse structure of a CNN can be approximated and covered by readily available dense components. After this structure is acquired, all we need to do is to repeat it spatially. Szegedy [24] crafted the Inception module for the optimal local sparse structure.

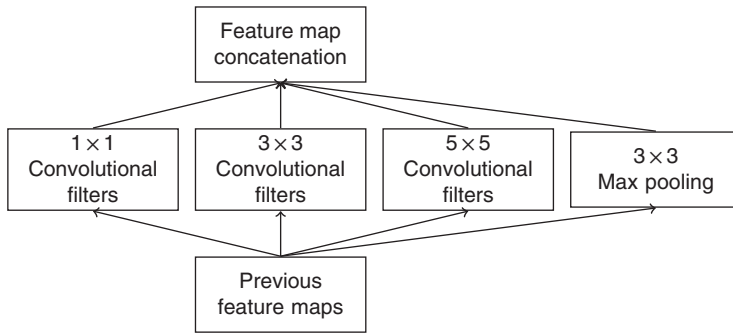


Figure 1.1 Naive version of the Inception module, refined from [24].

Szegedy [24] explains the design principle of the Inception module as follows. Each neuron from a layer corresponds to some region of the input image, and these neurons are grouped into feature maps according to their common properties. In the lower layers (the layers closer to the input), the correlated neurons would concentrate on the same local region. Thus, we would end up with a lot of groups concentrated in a single region, and these groups can be covered by using 1×1 convolutional filters, as suggested in [22], justifying the use of 1×1 convolutional filters in the Inception module.

However, there may be a small number of groups that are more spatially spread out and thus require larger convolutional filters for coverage over the larger patches. Consequently, the size of the convolutional filter used depends on the size of its receptive field. In general, there will be a decreasing number of groups over larger and larger regions. In order to avoid patch-alignment issues, the larger convolutional filters of the Inception module are restricted to 3×3 and 5×5 , a decision based more on convenience than on necessity.

Additionally, since max pooling operations have been essential for the success of current CNNs, Szegedy [24] suggested that adding an alternative parallel pooling path in the Inception module could have additional beneficial effects. The Inception module is a combination of all aforementioned components, including 1×1 , 3×3 , and 5×5 convolutional filters as well as 3×3 max pooling. Finally, their output feature maps are concatenated into a single output vector, forming the input for the next stage. Figure 1.1 shows the overall architecture of the devised Inception module.

1.2.4.2 Dimension Reduction

As illustrated in [24], the devised Inception module introduces one big problem: even a modest number of 5×5 convolutional filters can be prohibitively expensive on top of a convolutional layer with a large number of feature maps. This problem becomes even more pronounced once max pooling operations get involved since the number of output feature maps equals the number of feature maps in the previous layer. The merging of outputs of the pooling operation with outputs of convolutional filters would lead to an inevitable increase in the number of feature maps from layer to layer. Although the devised Inception module might cover the optimal sparse structure, it would do so very inefficiently, leading possibly to a computational blow-up within a few layers [24].

This dilemma inspired the second idea of the Inception module: to reduce dimensions judiciously only when the computational requirements would otherwise increase too

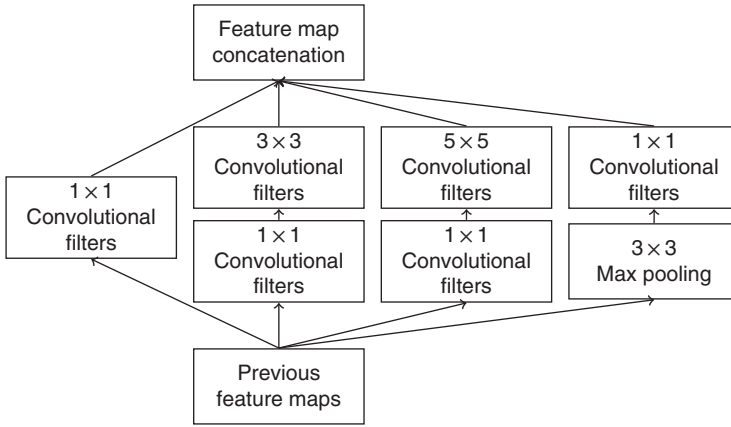


Figure 1.2 Inception module with dimension reduction, refined from [24].

much. For example, 1×1 convolutional filters are used to compute reductions before the more expensive 3×3 and 5×5 convolutional filters are used. In such a way, the number of neurons at each layer can be increased significantly without an uncontrolled blow-up in computational complexity at later layers. In addition to reductions, the Inception module also includes the use of ReLU activation functions for increased discriminative qualities. The final design is depicted in Figure 1.2.

1.2.5 ResNet

ResNet, proposed by [25], created a sensation in 2015 as the winner of several vision competitions in ILSVRC and COCO 2015, including ImageNet classification, ImageNet detection, ImageNet localization, COCO detection, and COCO segmentation. ResNet achieved a 3.57% top-5 error rate on the ImageNet test set, which was an almost 50% improvement from the 2014 winner, GoogLeNet, with a 6.67% top-5 error rate. Residual learning plays a critical role in ResNet since it eases the training of networks, and the networks can gain accuracy from considerably increased depth. As reported in He's tutorial presentation [14] at ICML 2016, ResNet addresses the three aspects of deep learning models as follows:

- 1) *Representation ability*. Although ResNet presents no explicit advantage on representation, it allowed models to go substantially deeper by re-parameterizing the learning between layers.
- 2) *Optimization ability*. ResNet enables very smooth forward and backward-propagation and hence greatly eases optimizing deeper models.
- 3) *Generalization ability*. Generalization is not explicitly addressed in ResNet, but [14] argued that deeper and thinner models have good generalization.

1.2.5.1 Residual Learning

Motivated by the degradation of training accuracy in deeper networks [38, 39], the idea of residual learning was proposed in [25] and was employed in ResNet. In accordance with [25], we will illustrate the idea of residual learning in the following. Residual

learning reformulates the few stacked layers as learning residual mappings with reference to the layer inputs, instead of learning unreferenced mappings.

Formally, let us denote $M_d(f)$ as a desired underlying mapping to be fit to a few stacked layers, with f denoting the inputs to the first of these layers. If one hypothesizes that multiple nonlinear layers can asymptotically approximate any complicated mapping, then it follows naturally to hypothesize that such layers can asymptotically approximate a residual mapping, i.e., $M_d(f) - f$ (assuming that the input and output are of the same dimensions). Thus, rather than expecting stacked layers to approximate $M_d(f)$, residual learning explicitly makes these layers approximate residual mapping $M_r(f) := M_d(f) - f$. The original mapping becomes $M_r(f) + f$. Although learning both residual mappings and unreferenced mappings should enable asymptotic approximates of desired mappings (as hypothesized), the ease of learning might be different.

With residual learning reformulation, if identity mappings are optimal, the solvers may simply drive the parameters of the multiple nonlinear layers towards zero to approach identity mappings. Though identity mappings are unlikely to be optimal in real cases, the residual learning reformulation may help precondition the learning problem. If the optimal mapping is closer to an identity mapping than to a zero mapping, it should be easier for the solvers to find the perturbations with reference to an identity mapping than to learn the function as a new one.

1.2.5.2 Identity Mapping by Shortcuts

In the following, we will further explain the implementation of residual learning in ResNet based on [25]. The formulation of $M_r(f) + f$ can be realized by devising neural networks with “shortcut connections”, which are those connections skipping one or more layers [40–42]. In ResNet, the shortcut connections simply perform identity mappings, and their outputs are added to the outputs of the stacked layers, as shown in Figure 1.3. These identity shortcut connections add neither an extra parameter nor computational complexity. The entire network can still be trained end-to-end with SGD and can be easily implemented by using common deep learning frameworks (e.g., Caffe [43], MXNet [44], Tensorflow [45], and SPeeDO [46]) without modifying the solvers. Formally, a building block of ResNet is defined as:

$$z = M_r(f, \{W_i\}) + f. \quad (1.3)$$

Here, f and z are the input and output vectors of the stacked layers, respectively. The function $M_r(f, \{W_i\})$ is the residual mapping to be learned. For example, there are two

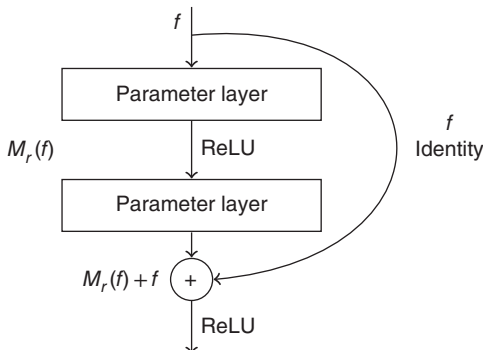


Figure 1.3 Residual learning: a building block, refined from [25].

layers in Figure 1.3, and hence the residual mapping for this devised building block is $M_r = W_2 g(W_1 f)$, where ResNet chooses ReLU as the activation function g and the biases are omitted for simplicity. The operation $M_r + f$ is performed by using a shortcut connection and then an element-wise addition. The second nonlinearity is employed after the addition (i.e., $g(z)$, see Figure 1.3). In [25], ResNet only employs a residual mapping M_r that has two or three layers, while more layers are permitted. However, Eq. 1.3 assumes that the dimensions of f and M_r must be the same. If the dimensions of the input and output channels differ, a linear projection W_s should be performed to match the dimensions. For this scenario, a building block turns out to be:

$$z = M_r(f, \{W_i\}) + W_s f. \quad (1.4)$$

For easy exposition, the above notations concern fully connected layers, but they are also applicable to convolutional layers. The mapping $M_r(f, \{W_i\})$ can represent multiple convolutional layers. The element-wise addition is performed on two feature maps, channel by channel.

1.2.6 Observations and Remarks

Despite the practical success of deep learning, theoretical understanding of the dynamics of learning in deep neural networks remains sparse. Progress has been made in interpreting deep neural networks via visualization (e.g., [47, 48]) and theoretical analysis (e.g., [49, 50]). However, why certain representations are learned or neurons activated to identify a particular object or semantic still cannot be well explained.

Specifically on the activation function selection, ReLU was shown to be effective for training ImageNet because of its computation efficiency and fast convergence in comparison with saturating nonlinear functions sigmoid and hyperbolic tangent. However, a theoretical study conducted by [51] using Riemannian geometry with the mean field theory of high dimensional chaos shows that when a network is very deep, sigmoidal networks with orthogonal weights can preserve both the norms of gradients and angles between pairs of gradients, whereas ReLUs cannot. At a recent conference talk, Ganguli [52] argued that there may be a potential path to resurrecting saturating nonlinearities in deep learning.

In summary, although this section presents effective methods to train the ImageNet dataset to achieve high classification accuracy, by no means the best configurations and parameter settings for training ImageNet should be directly applied to training other datasets with different network structures. Much research remains to be conducted for understanding and advancing deep learning.

1.3 Transfer Representation Learning

The success of CNNs relies on not only a good model to capture representations, but also a substantial amount of training data to learn representations from. Unfortunately, in many application domains, such as medicine, training data can be scarce and approaches such as data augmentation are not applicable. Transfer representation learning is a plausible alternative, which can remedy the insufficient training data issue. The common practice of transfer representation learning is to pretrain a CNN on a very large dataset

(called the source domain) and then to use the pretrained CNN as either an initialization or a fixed feature extractor for the task of interest (called target domain) [53].

We use disease diagnosis as the target domain to illustrate the problems of and solutions to the challenges of small data training. Specifically, we use otitis media (OM) and melanoma⁴ as two example diseases. The training data available to us are (i) 1195 OM images collected by seven otolaryngologists at Cathay General Hospital⁵, Taiwan [54] and (ii) 200 melanoma images from the PH² dataset [55]. The source domain from which representations are transferred to our two target diseases is ImageNet [12], which we have extensively discussed in the previous sections.

What are symptoms or characteristics of OM and melanoma? OM is any inflammation or infection of the middle ear, and treatment consumes significant medical resources each year [56]. Several symptoms, such as redness, bulging, and tympanic membrane perforation, may suggest an OM condition. Color, geometric, and texture descriptors may help in recognizing these symptoms. However, specifying these kinds of features involves a hand-crafted process and therefore requires domain expertise. Often, human heuristics obtained from domain experts may not be able to capture the most discriminative characteristics, and hence the extracted features cannot achieve high classification accuracy. Similarly, melanoma, a deadly skin cancer, is diagnosed based on the widely used dermoscopic “ABCD” rule [57], where A means asymmetry, B border, C color, and D different structures. The precise identification of such visual cues relies on experienced dermatologists to articulate. Unfortunately, there are many congruent patterns shared by melanoma and nevus, with skin, hair, and wrinkles often preventing noise-free feature extraction.

Our transfer representation learning experiments consist of the following five steps:

- 1) Unsupervised codebook construction: We learned a codebook from ImageNet images, and this codebook construction is “unsupervised” with respect to OM and melanoma.
- 2) Encode OM and melanoma images using the codebook: Each image was encoded into a weighted combination of the pivots in the codebook. The weighting vector is the feature vector of the input image.
- 3) Supervised learning: Using the transfer-learned feature vectors, we employed supervised learning to learn two classifiers from the 1195 labeled OM instances or 200 labeled melanoma instances.
- 4) Feature fusion: We also combined some heuristic features of OM (published in [54]) and ABCD features of melanoma with features learned via transfer learning.
- 5) Fine-tuning: We further fine-tuned the weights of the CNN using labeled data to improve classification accuracy.

As we will show in the remainder of this section, step 4 does not yield benefit, whereas the other steps are effective in improving diagnosis accuracy. In other words, these two disease examples demonstrate that features modeled by domain experts or physicians (the model-centric approach) are ineffective. The data-driven approach of big data representation learning combined with small data adaptation is convincingly promising.

⁴ In our award-winning XPRIZE Tricorder [15] device (code name DeepQ), we effectively diagnose 12 conditions, and OM and melanoma are two of them.

⁵ The dataset was used under a strict IRB process. The dataset was deleted by April 2015 after our experiments had been completed.

1.3.1 Method Specifications

We started with unsupervised codebook construction. On the large ImageNet dataset, we learned the representation of these images using AlexNet [11], presented in section 1.2.1. AlexNet contains eight neural network layers. The first five are convolutional and the remaining three are fully-connected. Different hidden layers represent different levels of abstraction concepts. We utilized AlexNet in Caffe [43] as our foundation to build our encoder to capture generic visual features.

For each image input, we obtained a feature vector using the codebook. The information of the image moves from the input layer to the output layer through the inner hidden layers. Each layer is a weighted combination of the previous layer and stands for a feature representation of the input image. Since the computation is hierarchical, higher layers intuitively represent higher concepts. For images, the neurons from lower levels describe rudimentary perceptual elements like edges and corners, whereas neurons from higher layers represent aspects of objects such as their parts and categories. To capture high-level abstractions, we extracted transfer-learned features of OM and melanoma images from the fifth, sixth, and seventh layers, denoted as pool5 (P5), fc6, and fc7 in Figure 1.4.

Once we had transfer-learned feature vectors of the 1195 collected OM images and 200 melanoma images, we performed supervised learning by training a support vector machine (SVM) classifier [59]. We chose SVMs to be our model since it is an effective classifier widely used by prior works. Using the same SVM algorithm lets us perform comparisons with the other schemes solely based on feature representation. As usual, we scaled features to the same range and found parameters through cross-validation. For fair comparisons with previous OM works, we selected the radial basis function (RBF) kernel.

To further improve classification accuracy, we experimented with two feature fusion schemes, which combine OM features hand-crafted by human heuristics (or model-centric) in [54] and our melanoma heuristic features with features learned from our codebook. In the first scheme, we combined transfer-learned and hand-crafted features to form fusion feature vectors. We then deployed the supervised learning on the fused feature vectors to train an SVM classifier. In the second scheme, we used the two-layer classifier fusion structure proposed in [54]. In brief, in the first layer we trained different classifiers based on different feature sets separately. We then combined the outputs from the first layer to train the classifier in the second layer.

Figure 1.5 summarizes our transfer representation learning approaches using OM images as an example. The top of the figure depicts two feature-learning schemes:

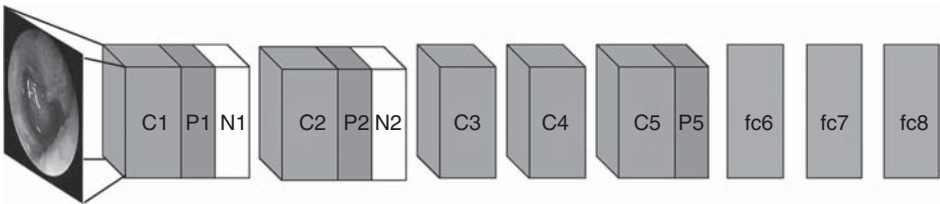


Figure 1.4 The flowchart of our transfer representation learning algorithm using OM images as an example [58].

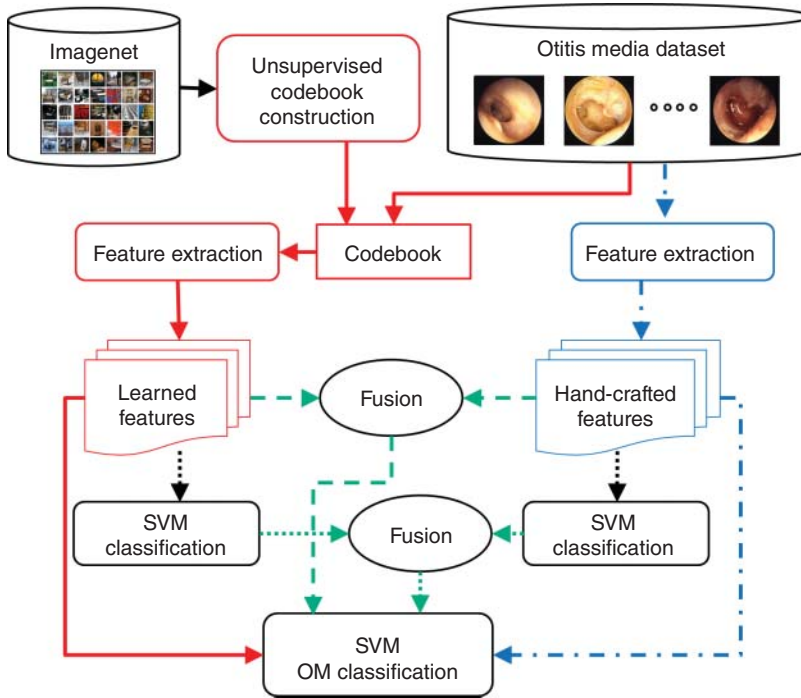


Figure 1.5 Four classification flows (the OM photos are from [58]).

the transfer-learned scheme on the left-hand side and the hand-crafted scheme on the right. The solid lines depict how OM or melanoma features are extracted via the transfer-learned codebook, whereas the dashed lines represent the flow of hand-crafted feature extraction. The bottom half of the figure describes two fusion schemes. Whereas the dashed lines illustrate the feature fusion by concatenating two feature sets, the dotted lines show the second fusion scheme at the classifier level. At the bottom of the figure, the four classification flows yield their respective OM-prediction decisions. In order from left to right in the figure are “transfer-learned features only”, “feature-level fusion”, “classifier-level fusion”, and “hand-crafted features only”.

1.3.2 Experimental Results and Discussion

Two sets of experiments were conducted to validate our idea. In this section, we first report OM classification performance by using our proposed transfer representation learning approach, followed by our melanoma classification performance. We then elaborate the correlations between images of ImageNet classes and images of disease classes by using a visualization tool to explain why transfer representation learning works.

For fine-tuning experiments, we performed a 10-fold cross-validation for OM and a 5-fold cross-validation for melanoma to train and test our models, so the test data are separated from the training dataset. We applied data augmentation, including random flip, mirroring, and translation, to all the images.

For the setting of training hyperparameters and network architectures, we used mini-batch gradient descent with a batch size of 64 examples, learning rate 0.001, momentum 0.9, and weight decay 0.0005. To fine-tune the AlexNet model, we replaced the fc6, fc7, and fc8 layers with three new layers initialized by using a Gaussian distribution with a mean of 0 and a standard deviation of 0.01. During the training process, the learning rates of those new layers were ten times greater than those of the other layers.

1.3.2.1 Results of Transfer Representation Learning for OM

Our 1195 OM image dataset encompasses almost all OM diagnostic categories: normal; acute otitis media (AOM): hyperemic stage, suppurative stage, ear drum perforation, subacute/resolution stage, bullous myringitis, barotrauma; otitis media with effusion (OME): with effusion, resolution stage (retracted); chronic otitis media (COM): simple perforation, active infection. Table 1.2 compares OM classification results for different feature representations. All experiments were conducted using 10-fold SVM classification. The measures of results reflect the discrimination capability of the features.

The first two rows in Table 1.2 show the results of human-heuristic methods (hand-crafted), followed by our proposed transfer-learned approach. The eardrum segmentation, denoted as “seg”, identifies the eardrum by removing OM-irrelevant information such as ear canal and earwax from the OM images [54]. The best accuracy achieved by using human-heuristic methods is 80.11%. With segmentation (first row), the accuracy improves 3% over that without segmentation (second row).

Rows 3 to 8 show the results of applying transfer representation learning. All results outperform the results shown in rows 1 and 2, suggesting that the features learned from transfer learning are superior to those of human-crafted ones.

Interestingly, segmentation does not help improve accuracy for learning representation via transfer learning. This indicates that the transfer-learned feature set is not only more discriminative but also more robust. Among three transfer-learning layer choices

Table 1.2 OM classification experimental results.

	Method	Accuracy (std)	Sensitivity	Specificity	F1 score
1	Heuristic w/ seg	80.11% (18.8)	83.33%	75.66%	0.822
2	Heuristic w/o seg	76.19% (17.8)	79.38%	71.74%	0.79
3	Transfer w/ seg (pool5)	87.86% (3.62)	89.72%	86.26%	0.89
4	Transfer w/o seg (pool5)	88.37% (3.41)	89.16%	87.08%	0.894
5	Transfer w/ seg (fc6)	87.58% (3.45)	89.33%	85.04%	0.887
6	Transfer w/o seg (fc6)	88.50% (3.45)	89.63%	86.90%	0.895
7	Transfer w/ seg (fc7)	85.60% (3.45)	87.50%	82.70%	0.869
8	Transfer w/o seg (fc7)	86.90% (3.45)	88.50%	84.90%	0.879
9	Feature fusion	89.22% (1.94)	90.08%	87.81%	0.90
10	Classifier fusion	89.87% (4.43)	89.54%	90.20%	0.898
11	Fine-tune	90.96% (0.65)	91.32%	90.20%	0.917

(layer 5 (pool5), layer 6 (fc6), and layer 7 (fc7)), fc6 yields slightly better prediction accuracy for OM. We believe that fc6 provides features that are more general or fundamental to transfer to a novel domain than pool5 and fc7 do. (Section 1.3.2.3 presents qualitative evaluation and explains why for OM fc6 is ideal.)

We also directly used the 1195 OM images to train a new AlexNet model. The resulting classification accuracy was only 71.8%, much lower than that achieved by applying transfer representation learning. This result confirms our hypothesis that even though CNN is a good model, with only 1195 OM images (without the ImageNet images to facilitate feature learning), it cannot learn discriminative features.

Two fusion methods, combining both hand-crafted and transfer learning features, achieved a slightly higher OM prediction F1 score (0.9 over 0.895) than using transfer-learned features only. This statistically insignificant improvement suggests that hand-crafted features do not provide much help.

Finally, we used OM data to fine-tune the AlexNet model, which achieves the highest accuracy. For fine-tuning, we replaced the original fc6, fc7, and fc8 layers with the new ones and used OM data to train the whole network without freezing any parameters. In this way, the learned features can be refined and are thus more aligned to the targeted task. This result attests that the ability to adapt representations to data is a critical characteristic that makes deep learning superior to the other learning algorithms.

1.3.2.2 Results of Transfer Representation Learning for Melanoma

We performed experiments on the PH² dataset whose dermoscopic images were obtained at the Dermatology Service of Hospital Pedro Hispano (Matosinhos, Portugal) under the same conditions through the Tuebinger Mole Analyzer system using a magnification of 20×. The assessment of each label was performed by an expert dermatologist.

Table 1.3 compares melanoma classification results for different feature representations. In Table 1.3, all the experiments except for the last two were conducted using 5-fold SVM classification. The last experiment involved fine-tuning, which was implemented and evaluated using Caffe. We also performed data augmentation to balance the PH² dataset (160 normal images and 40 melanoma images).

Unlike OM, we found the low-level features to be more effective in classifying melanoma. Among three transfer-learning layer choices, pool5 yields a more robust prediction accuracy than the other layers for melanoma. The deeper the layer is, the

Table 1.3 Melanoma classification experimental results.

	Method	Accuracy (std)	Sensitivity	Specificity	F1 score
1	ABCD rule w/ auto seg	84.38% (13.02)	85.63%	83.13%	0.8512
2	ABCD rule w/ manual seg	89.06% (9.87)	90.63%	87.50%	0.9052
3	Transfer w/o seg (pool5)	89.06% (10.23)	92.50%	85.63%	0.9082
4	Transfer w/o seg (fc6)	85.31% (11.43)	83.13%	87.50%	0.8686
5	Transfer w/o seg (fc7)	79.83% (14.27)	84.38%	74.38%	0.8379
6	Feature fusion	90.0% (9.68)	92.5%	87.5%	0.9157
7	Fine-tune	92.81% (4.69)	95.0%	90.63%	0.93

worse the accuracy becomes. We believe that pool5 provides low-level features that are suitable for delineating texture patterns that depict characteristics of melanoma.

Rows 3 and 7 show that the accuracy of transferred features is as good as that of the ABCD rule method with expert segmentation. These results reflect that deep transferred features are robust to noise such as hair or artifacts.

We used melanoma data to fine-tune the AlexNet model and obtained the best accuracy 92.81% since all network parameters are refined to fit the target task by employing back-propagation. We also compared our result with the cutting-edge method, which reported 98% sensitivity and 90% specificity on PH² [60]. This method requires pre-processing such as manual lesion segmentation to obtain “clean” data. In contrast, we utilized raw images without conducting any heuristic-based preprocessing. Thus, deep transfer learning can identify features in an unsupervised way to achieve as good classification accuracy as those features identified by domain experts.

1.3.2.3 Qualitative Evaluation: Visualization

In order to investigate what kinds of features are transferred or borrowed from the ImageNet dataset, we utilized a visualization tool to perform qualitative evaluation. Specifically, we used an attribute selection method, SVMAttributeEval [61] with Ranker search, to identify the most important features for recognizing OM and melanoma. Second, we mapped these important features back to their respective codebooks and used the visualization tool from [62] to find the top ImageNet classes causing the high value of these features. By observing the common visual appearances shared by the images of the disease classes and the retrieved top ImageNet classes, we were able to infer the transferred features.

Figure 1.6 demonstrates the qualitative analyses of four different cases: the normal eardrum, AOM, COM, and OME, which we will now proceed to explain in turn. First, the normal eardrum, nematode, and ticks are all similarly almost gray in color with a certain degree of transparency, features that are hard to capture with only hand-crafted methods. Second, AOM, purple-red cloth, and red wine have red colors as an obvious common attribute. Third, COM and seashells are both commonly identified by a calcified eardrum. Fourth, OME, oranges, and coffee all seem to share similar colors. Here, transfer learning works to recognize OM in an analogous fashion to how *explicit similes* are used in language to clarify meaning. The purpose of a simile is to provide information about one object by comparing it to something with which one is more familiar. For instance, if a doctor says that OM displays redness and certain textures, a patient may not be able to comprehend the doctor’s description exactly. However, if the doctor explains that OM presents with an appearance similar to that of a seashell, or with colors similar to that of red wine, oranges, or lattes, the patient is conceivably able to visualize the appearance of OM at a much more precise level. At level fc6, transfer representation learning works like finding similes that can help explain OM using the representations learned in the source domain (ImageNet).

To illustrate melanoma classification, Figure 1.7 presents visualization for three images, one benign nevus and two melanoma images. Columns (a) and (b) in Figure 1.7 show the same benign nevus image and its two most representative features. Columns (c) and (d) show the same melanoma image and its representative features. Column (e) displays a melanoma image with different visual characteristics.

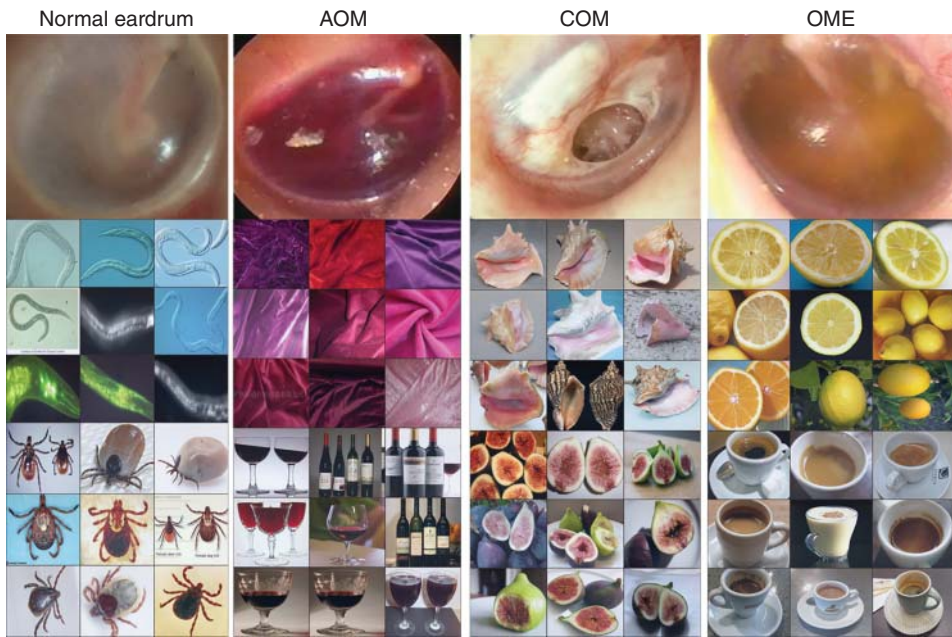


Figure 1.6 The visualization of helpful features from different classes corresponding to different OM symptoms (from left to right: normal eardrum, AOM, COM, and OME).

The top row of the figure shows three images inputted into the visualization tool. The second row reflects the activation maps, which we derived by employing C5 kernels on the input images. The larger the activation value used, the brighter the corresponding pixel returned. Using the respective kernels, the third row demonstrates the maximal activation map each image can reach [62]. We can thus conclude that each kernel has learned to detect specific features in order to capture the appearance of the images.

The classification of melanoma contrasts sharply with the classification of OM. We can exploit distinct visual features to classify different OM classes. However, melanoma and benign nevi share very similar textures, as melanoma evolves from benign nevi. Moreover, melanoma often has atypical textures and presents in various colors.

Among the three kinds of features in Figure 1.7, two of them are shared between benign nevi and melanoma. One is learned from the Cingulata shown in columns (a) and (c). According to columns (a) and (c), we can see that nevi, melanoma, and Cingulata share a pigment network and lattice pattern; the nevi in column (a) has a surrounding lattice pattern that causes a bright outline around the activation map in the second row. Other shared features are learned from the dirt images in columns (b) and (d); we can observe that nevi, melanoma, and dirt have similar colors and textures. For example, the nevi in column (b) is dirt-like in its interior, thus leading to the light circle in the second row. Column (e) shows one of the critical melanoma characteristics — a “blue-whitish veil” — which shares the same texture as that of the sparkle of sunlight on the lake (the third row of column (e)).

In the case of detecting melanoma versus benign nevus, effective representations of the diseases from higher-level visual characteristics cannot be found from the source

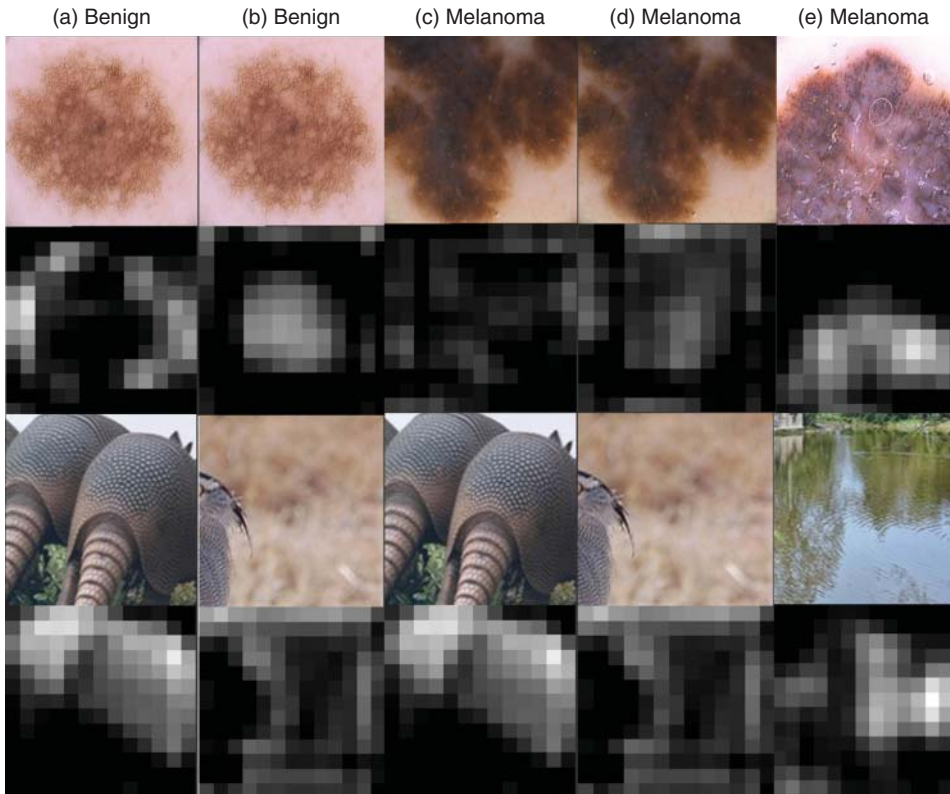


Figure 1.7 The visualization of helpful features for different patterns about classifying melanoma.

domain. Instead, the most effective representations are only transferrable at a lower level of the CNN. We believe that if the source domain can add substantial images of dirt and texture-rich objects, the effect of explicit similes may be utilized at a higher level of the CNN.

1.3.3 Observations and Remarks

In summary, this section demonstrates that transfer representation learning can potentially remedy two major challenges of medical image analysis: labeled data scarcity and medical domain knowledge shortage. Representations of OM and melanoma images can be effectively learned via transfer learning.

The transfer learned features achieve accuracy 90.96% (91.32% sensitivity and 90.20% specificity) for OM and 92.81% (95.0% sensitivity and 90.63% specificity) for melanoma, achieving an improvement in disease-classification accuracy over the feature extraction instructed by domain experts. Moreover, our algorithms do not require manual data cleaning beforehand, and the preliminary diagnosis of OM and melanoma can be derived without aid from doctors. Therefore, automatic disease diagnosis systems, which have the potential to help people lacking access to medical resources, are developmentally possible.

1.4 Conclusions

Deep learning owes its success to three key factors: scale of data, enhanced models to learn representations from data, and scale of computation. This chapter presents the importance of the data-driven approach to learn good representations from both big data and small data.

In terms of big data, it has been widely accepted in the research community that the more data the better for both representation and classification improvement. The question then is how to learn representations from big data and how to perform representation learning when data is scarce. We addressed the first question by presenting CNN model enhancements in the aspects of representation, optimization, and generalization. To address the small data challenge, we showed transfer representation learning to be effective. Transfer representation learning transfers the learned representation from a source domain where abundant training data is available to a target domain where training data is scarce. Transfer representation learning gave the OM and melanoma diagnosis modules of our XPRIZE Tricorder device (which finished second out of 310 competing teams [15]) a significant boost in diagnosis accuracy.

Our experiments on transfer learning provide three important insights on representation learning:

- 1) Low-level representations can be shared. Low-level perceptual features such as edges, corners, colors, and textures can be borrowed from some source domains where training data are abundant. After all, low-level representations are similar despite different high-level semantics.
- 2) Middle-level representations can be correlated. Analogous to explicit similes used in language, an object in the target domain can be “represented” or “explained” by some source domain features. In our OM visualization, we observed that a positive OM may be similar in color to seashells, red wine, oranges, or coffee — features learned and transferred from the ImageNet source domain.
- 3) Representations can adapt to a target domain. Even though, in the small data training situations, the amount of data is insufficient to learn effective representations by itself. Given representations learned from some big data source domains, the small data of the target domain can be used to align (e.g., re-weight) the representations learned from the source domains to adapt to the target domain.

Finally, we suggest further reading that provides details on improving the scale of the data and taking advantage of the scale of computation to speed up representation learning as follows.

For scale of data, refer to [2], which was originally submitted to *Transactions of IEEE* as an invited paper in 2010 but was rejected because a CNN pioneer did not consider scale of data to be a critical factor. (Two US patents on feature extraction [63] and objection recognition [64] using a hybrid approach of deep learning (model-centric) and big data (data-driven) were submitted in 2011 and granted in 2017 and 2014, respectively.) Two years later, AlexNet spoke volumes in support of the importance of the scale of data. For recent representative works in increasing data scale via synthetic data or unlabeled data, consult [65–70].

For an introduction to the computation of neural network models, refer to [71]. For available deep learning frameworks, refer to Torch [72], Caffe [43], MXNet [44], Theano [73], and TensorFlow [45]. For scale of computation, consult pioneering work at Google [74], and subsequent efforts on accelerating deep learning such as DistBelief [75], Adam [76], and SpeedO [46].

References

- 1 Lowe, D.G. (2004) Distinctive Image features from scale-invariant keypoints *International Journal of Computer Vision*, 60 (2), 91–110, doi: <http://dx.doi.org/10.1023/B:VISI.0000029664.99615.94>.
- 2 Chang, E.Y. (2011) Perceptual feature extraction (chapter 2), in *Foundations of Large-scale Multimedia Information Management and Retrieval: Mathematics of perception*, Springer, pp. 13–35.
- 3 Hinton, G.E. (2007) Learning multiple layers of representation. *Trends in Cognitive Sciences*, 11 (10), 428–434.
- 4 Hinton, G.E., Osindero, S., and Teh, Y.W. (2006) A fast learning algorithm for deep belief nets. *Neural Computation*, 18 (7), 1527–1554.
- 5 LeCun, Y., Bengio, Y., and Hinton, G. (2015) Deep learning. *Nature*, 521, 436–444.
- 6 Hubel, D.H. and Wiesel, T.N. (1968) Receptive fields and functional architecture of monkey striate cortex. *Journal of Physiology*, 195 (1), 215–243.
- 7 Fukushima, K. and Miyake, S. (1982) Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition, in *Competition and Cooperation in Neural Nets*, Springer, pp. 267–285.
- 8 McDermott, E. and Katagiri, S. (1989) Shift-invariant, multi-category phoneme recognition using Kohonen’s LVQ2, in *1989 International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, IEEE, pp. 81–84.
- 9 LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998) Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86 (11), 2278–2324.
- 10 Hinton, G.E. and Sejnowski, T.J. (1986) Learning and relearning in boltzmann machines, in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*. Volume 1: Foundations. MIT, Cambridge, pp. 282–317.
- 11 Krizhevsky, A., Sutskever, I., and Hinton, G.E. (2012) Imagenet classification with deep convolutional neural networks, in *Advances in Neural Information Processing Systems*, pp. 1097–1105.
- 12 Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., and Fei-Fei, L. (2009) Imagenet: A large-scale hierarchical image database, in *IEEE Conference on Computer Vision and Pattern Recognition 2009 (CVPR 2009)*, IEEE, pp. 248–255.
- 13 Lab, U.V. (2017), ILSVRC, <http://www.image-net.org/challenges/LSVRC/2017/index.php>.
- 14 He, K. (2016), ICML 2016 tutorial on deep residual networks, <http://kaiminghe.com/icml16tutorial/index.html>.
- 15 Qualcomm (2017), Xprize Tricorder Winning Teams, <http://tricorder.xprize.org/teams>.

- 16 Miller, E. (2000) The prefrontal cortex and cognitive control. *Nature Reviews Neuroscience*, 1 (1), 59–66.
- 17 Kravitz, D.J., Saleem, K.S., Baker, C.I., Ungerleider, L.G., and Mishkin, M. (2013) The ventral visual pathway: an expanded neural framework for the processing of object quality. *Trends in Cognitive Sciences*, 17 (1), 26–49.
- 18 Sermanet, P., Chintala, S., and LeCun, Y. (2012) Convolutional neural networks applied to house numbers digit classification, in *2012 21st International Conference on Pattern Recognition (ICPR)*, IEEE, pp. 3288–3291.
- 19 Sermanet, P. and LeCun, Y. (2011) Traffic sign recognition with multi-scale convolutional networks, in *The 2011 International Joint Conference on Neural Networks (IJCNN)*, IEEE, pp. 2809–2813.
- 20 Li, H., Lin, Z., Shen, X., Brandt, J., and Hua, G. (2015) A convolutional neural network cascade for face detection, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, pp. 5325–5334.
- 21 Zeiler, M.D. and Fergus, R. (2014) Visualizing and understanding convolutional networks, in *Computer Vision – ECCV 2014*, Springer, pp. 818–833.
- 22 Lin, M., Chen, Q., and Yan, S. (2013) Network in network. *arXiv preprint arXiv:1312.4400*.
- 23 Simonyan, K. and Zisserman, A. (2014) Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.
- 24 Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2015) Going deeper with convolutions, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, pp. 1–9.
- 25 He, K., Zhang, X., Ren, S., and Sun, J. (2015) Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*.
- 26 Nair, V. and Hinton, G.E. (2010) Rectified linear units improve restricted Boltzmann machines, in *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pp. 807–814.
- 27 Ciregan, D., Meier, U., and Schmidhuber, J. (2012) Multi-column deep neural networks for image classification, in *2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, pp. 3642–3649.
- 28 Cireşan, D.C., Meier, U., Masci, J., Gambardella, L.M., and Schmidhuber, J. (2011) High-performance neural networks for visual object classification. *arXiv preprint arXiv:1102.0183*.
- 29 Simard, P.Y., Steinkraus, D., Platt, J.C. et al. (2003) Best practices for convolutional neural networks applied to visual document analysis, in *ICDAR*, vol. 3, Citeseer, pp. 958–962.
- 30 Li, B., Chang, E., and Wu, Y. (2003) Discovery of a perceptual distance function for measuring image similarity. *Multimedia Systems*, 8 (6), 512–522.
- 31 Breiman, L. (1996) Bagging predictors. *Machine Learning*, 24 (2), 123–140.
- 32 Freund, Y. and Schapire, R.E. (1995) A decision-theoretic generalization of on-line learning and an application to boosting, in *European Conference on Computational Learning Theory*, Springer, pp. 23–37.
- 33 Breiman, L. (2001) Random forests. *Machine Learning*, 45 (1), 5–32.

- 34 Hinton, G.E., Srivastava, N., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R.R. (2012) Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*.
- 35 Zeiler, M.D. and Fergus, R. (2013) Stochastic pooling for regularization of deep convolutional neural networks. *arXiv preprint arXiv:1301.3557*.
- 36 Goodfellow, I.J., Warde-Farley, D., Mirza, M., Courville, A.C., and Bengio, Y. (2013) Maxout networks. *Proceedings of the 30th International Conference on Machine Learning*, Atlanta, Georgia, USA, 17–19 Jun, PMLR 2, 1319–1327.
- 37 Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., and LeCun, Y. (2013) Overfeat: Integrated recognition, localization and detection using convolutional networks. *arXiv preprint arXiv:1312.6229*.
- 38 He, K. and Sun, J. (2015) Convolutional neural networks at constrained time cost, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, pp. 5353–5360.
- 39 Srivastava, R.K., Greff, K., and Schmidhuber, J. (2015) Highway networks. *arXiv preprint arXiv:1505.00387*.
- 40 Bishop, C.M. (1995) *Neural Networks for Pattern Recognition*, Oxford University Press.
- 41 Ripley, B.D. (2007) *Pattern Recognition and Neural Networks*, Cambridge University Press.
- 42 Venables, W.N. and Ripley, B.D. (2013) *Modern Applied Statistics with S-PLUS*, Springer Science & Business Media.
- 43 Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., and Darrell, T. (2014) Caffe: Convolutional architecture for fast feature embedding, in *Proceedings of the ACM International Conference on Multimedia*, ACM, pp. 675–678.
- 44 Chen, T., Li, M., Li, Y., Lin, M., Wang, N., Wang, M., Xiao, T., Xu, B., Zhang, C., and Zhang, Z. (2015) Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems. *arXiv preprint arXiv:1512.01274*.
- 45 Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M. et al. (2016) Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*.
- 46 Zheng, Z., Jiang, W., Wu, G., and Chang, E.Y. (2015) Speedo: Parallelizing stochastic gradient descent for deep convolutional neural network, in *NIPS Workshop on Machine Learning Systems (LearningSys)*.
- 47 Foerster, J.N., Gilmer, J., Sohl-Dickstein, J., Chorowski, J., and Sussillo, D. (2017) Input switched affine networks: An RNN architecture designed for interpretability, in *Proceedings of the 34th International Conference on Machine Learning, Proceedings of Machine Learning Research*, vol. 70 (eds D. Precup and Y.W. Teh), International Convention Centre, Sydney, Australia, pp. 1136–1145.
- 48 Nagamine, T. and Mesgarani, N. (2017) Understanding the representation and computation of multilayer perceptrons: A case study in speech recognition, in *Proceedings of the 34th International Conference on Machine Learning, Proceedings of Machine Learning Research*, vol. 70 (eds D. Precup and Y.W. Teh), International Convention Centre, Sydney, Australia, pp. 2564–2573.

- 49 Gao, P. and Ganguli, S. (2015) On simplicity and complexity in the brave new world of large-scale neuroscience. *Current Opinion in Neurobiology*, 32, 148–155.
- 50 Koh, P.W. and Liang, P. (2017) Understanding black-box predictions via influence functions, in *Proceedings of the 34th International Conference on Machine Learning, Proceedings of Machine Learning Research*, vol. 70 (eds D. Precup and Y.W. Teh), PMLR, International Convention Centre, Sydney, Australia, pp. 1885–1894.
- 51 Poole, B., Lahiri, S., Raghu, M., Sohl-Dickstein, J., and Ganguli, S. (2016) Exponential expressivity in deep neural networks through transient chaos, in *Advances in Neural Information Processing Systems* 29, pp. 3360–3368, Curran Associates, Inc.
- 52 Ganguli, S. (2017) On the beneficial role of dynamic criticality and chaos in deep learning. *Plenary Talk, NIPS Principled Approaches to Deep Learning Workshop*.
- 53 Karpathy, A. and Johnson, J. (2017), CS231n convolutional neural network for visual recognition: transfer learning, <http://cs231n.github.io/transfer-learning/>.
- 54 Shie, C.K., Chang, H.T., Fan, F.C., Chen, C.J., Fang, T.Y., and Wang, P.C. (2014) A hybrid feature-based segmentation and classification system for the computer aided self-diagnosis of otitis media, in *Engineering in Medicine and Biology Society (EMBC), 2014 36th Annual International Conference of the IEEE, IEEE*, pp. 4655–4658.
- 55 Mendonça, T., Ferreira, P.M., Marques, J.S., Marcal, A.R., and Rozeira, J. (2013) Ph 2—a dermoscopic image database for research and benchmarking, in *Engineering in Medicine and Biology Society (EMBC), 2013 35th Annual International Conference of the IEEE*, pp. 5437–5440.
- 56 American Academy of Pediatrics Subcommittee on Management of Acute Otitis Media and others (2004) Diagnosis and management of acute otitis media. *Pediatrics*, 113 (5), 1451.
- 57 Stolz, W., Riemann, A., Cognetta, A., Pillet, L., Abmayr, W., Holzel, D., Bilek, P., Nachbar, F., and Landthaler, M. (1994) ABCD rule of dermatoscopy: a new practical method for early recognition of malignant melanoma *European Journal of Dermatology*, pp. 521–527.
- 58 Wikipedia (2017), Otitis media., https://en.wikipedia.org/wiki/Otitis_media.
- 59 Chang, C.C. and Lin, C.J. (2011) Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2 (3), 27.
- 60 Barata, C., Emre Celebi, M., and Marques, J.S. (2015) Melanoma detection algorithm based on feature fusion, in *Engineering in Medicine and Biology Society (EMBC), 2015 37th Annual International Conference of the IEEE, IEEE*, pp. 2653–2656.
- 61 Guyon, I., Weston, J., Barnhill, S., and Vapnik, V. (2002) Gene selection for cancer classification using support vector machines. *Machine Learning*, 46 (1-3), 389–422.
- 62 Yosinski, J., Clune, J., Nguyen, A., Fuchs, T., and Lipson, H. (2015) Understanding neural networks through deep visualization. *arXiv preprint arXiv:1506.06579*.
- 63 Wang, Z. and Chang, E.Y. (2017), US Patent#9547914: Techniques for Feature Extraction, <https://www.google.com/patents/USUS9547914>.
- 64 Chang, E.Y., Wang, Z., and Xia, D. (2014), US Patent#8798375: Object Recognition in Images, <https://www.google.com/patents/US8798375>.

- 65 Arora, S., Ge, R., Liang, Y., Ma, T., and Zhang, Y. (2017) Generalization and equilibrium in generative adversarial nets (GANs), in *Proceedings of the 34th International Conference on Machine Learning, Proceedings of Machine Learning Research*, vol. 70 (eds D. Precup and Y.W. Teh), PMLR, International Convention Centre, Sydney, Australia, pp. 224–232.
- 66 Jaderberg, M., Simonyan, K., Vedaldi, A., and Zisserman, A. (2014) Synthetic data and artificial neural networks for natural scene text recognition. *arXiv preprint arXiv:1406.2227*.
- 67 Peng, X., Sun, B., Ali, K., and Saenko, K. (2015) Learning deep object detectors from 3D models, in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1278–1286.
- 68 Rogez, G. and Schmid, C. (2016) Mocap-guided data augmentation for 3D pose estimation in the wild, in *Advances in Neural Information Processing Systems*, pp. 3108–3116, Curran Associates, Inc.
- 69 Shrivastava, A., Pfister, T., Tuzel, O., Susskind, J., Wang, W., and Webb, R. (2016) Learning from simulated and unsupervised images through adversarial training. *arXiv preprint arXiv:1612.07828*.
- 70 Sixt, L., Wild, B., and Landgraf, T. (2016) Rendergan: Generating realistic labeled data. *arXiv preprint arXiv:1611.01331*.
- 71 Dally, W. (2015), NIPS 2015 tutorial on high-performance hardware for machine learning, <https://media.nips.cc/Conferences/2015/tutorialslides/Dally-NIPS-Tutorial-2015.pdf>.
- 72 Collobert, R., Kavukcuoglu, K., and Farabet, C. (2011) Torch7: A Matlab-like environment for machine learning, in *BigLearn, NIPS Workshop*, EPFL-CONF-192376.
- 73 Al-Rfou, R., Alain, G., Almahairi, A., Angermueller, C., Bahdanau, D., Ballas, N., Bastien, F., Bayer, J., Belikov, A., Belopolsky, A. et al. (2016) Theano: A python framework for fast computation of mathematical expressions. *arXiv preprint arXiv:1605.02688*.
- 74 Cafarella, M.J., Chang, E.Y., Fikes, A., Halevy, A.Y., Hsieh, W.C., Alberto Lerner, J.M., and Muthukrishnan, S. (2008) Data management projects at google. *SIGMOD Record*, 37 (1), 34–38.
- 75 Dean, J., Corrado, G., Monga, R., Chen, K., Devin, M., Mao, M., Senior, A., Tucker, P., Yang, K., Le, Q.V. et al. (2012) Large scale distributed deep networks, in *Proceedings of the 25th International Conference on Neural Information Processing Systems (NIPS12)*, Volume 1, Lake Tahoe, Nevada, pp. 1223–1231, Curran Associates Inc.
- 76 Chilimbi, T.M., Suzue, Y., Apacible, J., and Kalyanaraman, K. (2014) Project Adam: Building an efficient and scalable deep learning training system, *Proceedings of the 11th USENIX Conference on Operating Systems Design and Implementation (OSDI'14)*, Broomfield, CO, USENIX Association pp. 571–582.

2

Concept-Based and Event-Based Video Search in Large Video Collections

Foteini Markatopoulou, Damianos Galanopoulos, Christos Tzelepis, Vasileios Mezaris and Ioannis Patras

Video content can be annotated with semantic information such as simple concept labels that may refer to objects (e.g., “car” and “chair”), activities (e.g., “running” and “dancing”), scenes (e.g., “hills” and “beach”), etc., or more complex (or high-level) events that describe the main action that takes place in the complete video. An event may refer to complex activities, occurring at specific places and times, which involve people interacting with other people and/or object(s), such as “changing a vehicle tire”, “making a cake”, or “attempting a bike trick”, etc. Concept-based and event-based video search refers to the retrieval of videos/video fragments (e.g., keyframes) that present specific simple concept labels or more complex events from large-scale video collections, respectively. To deal with concept-based video search, video concept detection methods have been developed that automatically annotate video fragments with semantic labels (concepts). Then, given a specific concept, a ranking component retrieves the top related video fragments for this concept. While significant progress has been made during the last few years in video concept detection, it continues to be a difficult and challenging task. This is due to the diversity in form and appearance exhibited by the majority of semantic concepts and the difficulty of expressing them using a finite number of representations. A recent trend is to learn features directly from the raw keyframe pixels using deep convolutional neural networks (DCNNs). Other studies focus on combining many different video representations in order to capture different perspectives of the visual information. Finally, there are studies that focus on multi-task learning in order to exploit concept model sharing, and methods that look for existing semantic relations, e.g., concept correlations. In contrast to concept detection, where we most often can use annotated training data for learning the detectors, in the problem of video event detection we can distinguish two different but equally important cases: when a number of positive examples or no positive examples at all (“zero-example” case) are available for training. In the first case, a typical video event detection framework includes a feature extraction and a classification stage, where an event detector is learned by training one or more classifiers for each event class using available features (sometimes similarly to the learning of concept detectors), usually followed by a fusion approach in order to combine different modalities. In

the latter case, where only a textual description is available for each event class, the research community has directed its efforts towards effectively combining textual and visual analysis techniques, such as using text analysis techniques, exploiting large sets of DCNN-based concept detectors and using various re-ranking methods, such as pseudo-relevance feedback or self-paced re-ranking. In this chapter, we survey the literature and present our research efforts for improving concept- and event-based video search. For concept-based video search, we focus on (i) feature extraction using hand-crafted and DCNN-based descriptors, (ii) dimensionality reduction using accelerated generalised subclass discriminant analysis (AGSDA), (iii) cascades of hand-crafted and DCNN-based descriptors, (iv) multi-task learning (MTL) to exploit model sharing, and (v) stacking architectures to exploit concept relations. For video event detection, we focus on methods which exploit positive examples, when available, again using DCNN-based features and AGSDA, and we also develop a framework for zero-example event detection that associates the textual description of an event class with the available visual concepts in order to identify the most relevant concepts regarding the event class. Additionally, we present a pseudo-relevant feedback mechanism that relies on AGSDA.

2.1 Introduction

Video understanding is the overall problem that deals with automatically detecting what is depicted in a video sequence. This problem can be divided into two separate sub-problems that focus on different levels of video analysis. Concept-based video search, which works at video fragment level, i.e., assigning one or more semantic concepts to video fragments (e.g., video keyframes) based on a predefined concept list (e.g., “car”, “running”) [1], and event-based video search, which works at the complete video level, i.e., detecting the main event that is presented on the overall video sequence. These two sub-problems share some video preprocessing techniques. For example, video representation and dimensionality reduction are typically the same. However, they are overall treated as two separate tasks because the different video analysis levels that each of them focuses on require different detection and recognition mechanisms.

Video concept detection is an important video understanding problem that facilitates many applications, such as semantics-based video segmentation, video event detection and concept-based video search. The latter, which is the problem investigated in this section, refers to the retrieval of videos/video fragments (e.g., keyframes) that present specific simple concept labels. In a typical video concept detection process, the video is initially segmented into meaningful fragments called shots, and each shot may be represented by one or more characteristic keyframes that are subsequently annotated. This is a multi-label classification problem (one keyframe may be annotated with more than one semantic concepts) that can be treated as multiple independent binary classification problems, where for each concept a model can be learned to distinguish keyframes in which the concept appears from those in which the concept does not appear. Given feature-based keyframe representations that have been extracted from different keyframes and also the ground-truth annotations for each keyframe (i.e., the concepts presented) any supervised machine learning algorithm that solves classification problems can be used to learn the relations between the low-level image representations and the high-level semantic concepts. It has been shown that combining

many different keyframe representations (e.g., SIFT, RGB-SIFT, DCNN-based) for the same concept, instead of using a single feature (e.g., only SIFT), improves the concept detection accuracy. Multi-task learning, which refers to methods that learn many tasks together at the same time, is another category of methods that aim to improve concept detection accuracy. Finally, exploiting label relations, the relations between concepts within a video shot (e.g., the fact that *sun* and *sky* will appear in the same video shot in most cases) is a third category of concept detection methods or accuracy improvement. We will focus on three general learning areas for improved concept-based video search:

- combination of video representations using cascades (section 2.3.2)
- multi-task learning for concept-based video search (section 2.3.3)
- exploiting label relations using a two-layer stacking architecture (section 2.3.4).

Event detection, the second sub-problem we tackle, facilitates applications such as event-based video search, i.e., the retrieval of videos that present a specific event. As a video event we consider a complex activity involving people interacting with other people and/or objects, e.g., “Renovating a home”. A typical video event detection pipeline starts with a feature extraction stage, which usually generates a plethora of low-, intermediate-, and high-level features from the different available modalities (visual, audio, and/or textual). A classification stage follows, where an event detector has been learned by training one or more classifiers for each event class using some or all of the extracted video features and positive video examples (under the assumption that such examples are available). Finally, a fusion approach may be followed in order to combine different modalities. A slightly different and more challenging problem related to video event detection is zero-example event detection, which refers to the case where only a textual description of the event is available without any positive training examples. Typically, a zero-example system starts by analysing the textual event description to transform it to a meaningful set of keywords. At the same time, a predefined set of concepts is used, on the one hand to find which of these concepts are related to the extracted keywords and consequently to the event description, and on the other hand to train visual concept detectors (i.e., using concept detection approaches such as those presented in section 2.3) that will be used to annotate the videos with these concepts. Regarding event-based video search we will present:

- methods for video event detection when positive training examples are available (section 2.4.2)
- methods for video event detection when only a textual description of the event is given (section 2.4.3).

2.2 Video preprocessing and Machine Learning Essentials

2.2.1 Video Representation

A variety of visual, textual, and audio features can be extracted to represent each piece of visual information; a review of different types of features can be found in [1]. Despite the fact that audio features have also been shown to be useful in visual understanding problems [1], in this work we focus mostly on visual features. Visual features are typically extracted from representative keyframes or similar 2D image structures [2], or from sequences of frames (e.g., motion features). We can distinguish

two main categories of (static, i.e. non-motion) visual features: hand-crafted features and features based on deep convolutional networks (DCNN-based). With respect to hand-crafted features, binary (ORB [3]) and non-binary (SIFT [4], SURF [5]) local descriptors, as well as color extensions of them [6] have been examined for video concept detection. Local descriptors are aggregated into global image representations by employing feature encoding techniques such as Fisher vector (FV) [7] and VLAD [8]. With respect to DCNN-based features, one or more hidden layers of a pretrained DCNN are typically used as a global image representation [9]. Several DCNN software libraries are available in the literature, e.g., Caffe, 2014 and MatConvNet, and different DCNN architectures have been proposed, e.g., CaffeNet [10], GoogLeNet [11], and VGG ConvNet [9]. DCNN-based descriptors present high discriminative power and generally outperform the local descriptors [12, 13].

A DCNN can be trained on a large-scale dataset and then can be used in two different ways to annotate new test keyframes with semantic concepts.

- a) As standalone classifier: Each test keyframe is forward-propagated by the network and the network's output is used as the final class distribution assigned to the keyframe [9, 10], a process also known as direct classification.
- b) As feature generator: The training set is forward-propagated by the network and the features extracted from one or more layers of the network are used as feature vectors to subsequently train one supervised classifier (concept detector) per concept. Each test keyframe is first described by the DCNN-based features and subsequently serves as input to the trained classifiers.

DCNN training requires learning millions of parameters, which means that a small-sized training set, which is the case for video datasets, could easily over-fit the DCNN on the training data. It has been proven that the bottom layers of a DCNN learn rather generic features, useful for different domains, while the top layers are task-specific [14]. Transferring a pretrained network in a new dataset by fine-tuning its parameters is a common strategy that can take advantage of the bottom generic layers and adjust the top layers to the target dataset and the new target concepts. Fine-tuning is a process in which the weights of a pretrained DCNN are used as the starting point for a new target training set and are modified in order to adapt the pretrained DCNN to the new target dataset [15].

2.2.2 Dimensionality Reduction

Dimensionality reduction in multimedia understanding problems, such as those of concept- and event-based video search, has been shown to be very useful, since in such domains the data representations are typically high-dimensional. To this end, non-linear discriminant analysis (NDA) approaches [16], which aim to identify a discriminant subspace of the original high-dimensional input space, and GPU implementations of computationally intensive algorithms have recently received increasing attention in large-scale video analysis problems [17–19]. For instance, generalised subclass discriminant analysis (GSDA) combined with linear support vector machines (LSVMs) achieved excellent performance in multimedia event detection [18], while in [17, 20] GPU accelerated machine learning algorithms obtain substantial speedups.

In this work, to perform dimensionality reduction efficiently, we use a non-linear discriminant analysis technique called accelerated generalised subclass discriminant analysis (AGSDA) [16] along with its GPU implementation [21, 22]. This method identifies a discriminant subspace of the input space in three steps: (i) Gram matrix computation, (ii) eigenvalue decomposition of the between subclass factor matrix, and (iii) computation of the solution of a linear matrix system with a symmetric positive semidefinite (SPSD) matrix of coefficients. Based on the fact that the computationally intensive parts of AGSDA, i.e., Gram matrix computation and identification of the SPSP linear matrix system solution, are highly parallelisable, a GPU implementation of AGSDA is employed [22]. The experimental evaluation on large-scale datasets of TRECVID for concept and event detection shows that the combination of GPU-AGSDA and LSVM outperforms LSVM alone in training time, memory consumption, and detection accuracy.

2.3 Methodology for Concept Detection and Concept-Based Video Search

2.3.1 Related Work

Video concept detection is a multi-label classification problem (one keyframe may be annotated with more than one semantic concept) that is typically treated as multiple independent binary classification problems, one per concept. Given feature-based keyframe representations that have been extracted from different keyframes and also the ground-truth annotations for each keyframe (i.e., the concepts presented) any supervised machine learning algorithm that solves classification problems can be used in order to train a concept detector. It has been shown that combining many different keyframe representations (e.g., SIFT, RGB-SIFT, DCNN-based) for the same concept, instead of using a single feature (e.g., only SIFT), improves the concept detection accuracy. The typical way to combine multiple features is to train several supervised classifiers for the same concept, each trained separately on a different feature. When all the classifiers give their decisions, a fusion step computes the final confidence score (e.g., by averaging); this process is known as late fusion. Hierarchical late fusion [23] is a more elaborate approach; classifiers that have been trained on more similar features (e.g., SIFT and RGB-SIFT) are first fused together and then other dissimilar classifiers (e.g., DCNN-based) are sequentially fused with the previous groups. A second category of classifier combination approach performs ensemble pruning to select a subset of the classifiers prior to their fusion. For example, [24] uses a genetic algorithm to automatically select an optimal subset of classifiers separately for each concept. Finally, there is a third group of popular ensemble-based algorithms, namely cascade architectures, which have been used in various visual classification tasks for training and combining detectors [25, 26]. In a cascade architecture, e.g., [27], the classifiers are arranged in stages, from the less computationally demanding to the most demanding (or may be arranged according to other criteria, such as their accuracy). A keyframe is classified sequentially by each stage and the next stage is triggered only if the previous one returns a positive prediction (i.e., that the concept or object appears

in the keyframe). The rationale behind this is to rapidly reject keyframes that clearly do not match the classification criteria and focus on those keyframes that are more difficult and more likely to depict the sought concept. Cascades of classifiers have been mainly used in object detection tasks, however they have also been briefly examined for video/image concept detection [25, 27].

Independently training concept detectors is a single-task learning (STL) process, where each task involves recognizing one concept. However, video concept detection can be treated as an MTL problem where the different tasks (one per concept) can be learned jointly by allowing the sharing of knowledge across them. The main difference between MTL methods is the way they define task relatedness, i.e., the type of knowledge that should be shared. Some methods identify shared features between different tasks and use regularization to model task relatedness [28]. Others identify a shared subspace over the task parameters [29, 30]. These methods make the strong assumption that all tasks are related; some newer methods consider the fact that some tasks may be unrelated [31, 32].

Two main types of method that exploit label relations have been adopted in the literature: (i) stacking-based approaches that collect the scores produced by a baseline set of concept detectors (in a first layer) and introduce a second learning step in order to refine them (in a second layer) and (ii) inner-learning approaches that follow a single-step learning process that jointly considers low-level visual features and concept correlation information [1]. Assuming that we first train a set of SVM-based or logistic regression (LR) independent concept detectors using either a cascade architecture, or another typical fusion scheme (e.g., late fusion in terms of arithmetic mean), and then apply this set of concept detectors to new test keyframes, stacking approaches aim to refine the initial predictions on the test set by detecting dependencies among concepts in the last layer of the stack. For example, discriminative model fusion (DMF) [33] obtains concept score predictions from the individual concept detectors in the first layer in order to create a *model vector* for each shot. These vectors form a meta-level training set, which is used to train a second layer of independent concept detectors. Correlation-based pruning of stacked binary relevance models (BSBRM) [34] extends the previous approach by pruning the predictions of non-correlated concept detectors before the training of each individual classifier of the second-layer models. Similarly to DMF, the baseline CBCF (BCBCF) [35] forms model vectors, in this case using the ground-truth annotation, in order to train second-layer binary relevance (BR) models. Furthermore, the authors of [35] note that not all concepts can take advantage of CBCF, so their method refines only a subset of them. Another group of stacking approaches is the graph-based ones, which model label correlations explicitly [1]. The multi-cue fusion (MCF) method [36] uses the ground-truth annotation to build decision trees that describe the relations among concepts, separately for each concept. Initial scores are refined by approximating these graphs. Inner-learning approaches, on the other hand, make use of contextual information from the beginning of the concept learning process. For example, the authors of [37] propose methods that simultaneously learn the relation between visual features and concepts, and also the correlations among concepts. However, inner-learning approaches suffer from computational complexity. For example, [37] has complexity at least quadratic to the number of concepts, making it inapplicable to real problems where the number of concepts is large (e.g., hundreds or thousands).

2.3.2 Cascades for Combining Different Video Representations

A cascade architecture, for example the one we developed in [38], can be used to effectively combine many base classifiers that have been trained for the same concept (Figure 2.1). Each stage j of the cascade encapsulates a stage classifier D_j that either combines many base classifiers (B_1, B_2, \dots, B_{f_j} , Figure 2.1b) that have been trained on different types of features or contains only one base classifier (B_1) that has been trained on a single type of features (Figure 2.1c). In the first case, the output of f_j base classifiers is combined in order to return a single-stage output score $D_j(I) = \frac{1}{f_j} \sum_{i=1}^{f_j} B_i(I)$, $f_j \geq 1$ in the $[0,1]$ range. The second case is a special case where $f_j = 1$. Let I indicate an input keyframe; the classifier D_{j+1} of the cascade will be triggered for it only if the previous classifier does not reject the input keyframe I . Each stage j of the cascade is associated with a rejection threshold, while a stage classifier is said to reject an input keyframe if $D_j(I) < \theta_j$. A rejection indicates the classifier's belief that the concept does not appear in the keyframe. Let $D = \{D_1, D_2, \dots, D_n\}$ be a set of n independently trained classifiers for a specific concept.

The rest of this section presents an advanced algorithm that sets the ordering of cascade stages (i.e., the ordering of stage classifiers) and assigns thresholds to each stage in order to instantiate the proposed cascade of Figure 2.1. The ordering of stages and the threshold assignment is performed for the optimization of the complete cascade and not the optimization of each stage separately from the other stages.

2.3.2.1 Problem Definition and Search Space

Let $\mathbf{S} = [s_1, s_2, \dots, s_n]^T$ denote a vector of integer numbers in $[-1, 0] \cup [1, n]$. Each number represents the index of a classifier from D and appears at most once. The value -1 indicates that a classifier from D is omitted. Consequently, \mathbf{S} expresses the ordering of the pretrained classifiers D_1, \dots, D_n . For example, given a pretrained set of four classifiers $D = \{D_1, D_2, D_3, D_4\}$, the solution $\mathbf{S} = [2, 1, 3, -1]^T$ denotes the cascade $D_{2,1,3,-1} : D_2 \rightarrow D_1 \rightarrow D_3$, where stage classifier D_4 is not used at all. In addition, let $\theta = [\theta_1, \theta_2, \dots, \theta_n]^T$ denote a vector of rejection thresholds for the solution \mathbf{S} and let $T = \{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^M$, where

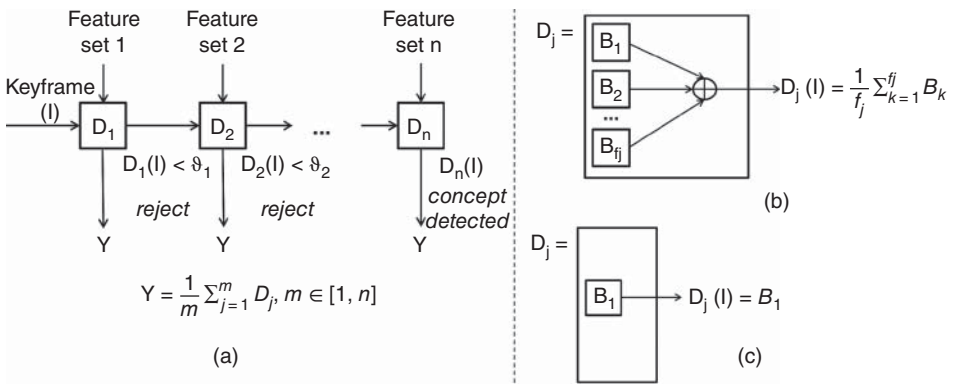


Figure 2.1 (a) Block diagram of the developed cascade architecture for one concept. (b) Stage combining many base classifiers trained on different features. (c) Stage with one base classifier trained on a single feature.

$y_i \in \{\pm 1\}$, is a set of annotated training samples for the given concept (\mathbf{x}_i being the feature vectors and y_i the ground-truth annotations). The problem we aim to solve is to find the pair of the index sequence \mathbf{S} (that leads to the cascade $D_{\mathbf{S}} : D_{s_1} \rightarrow D_{s_2} \rightarrow \dots \rightarrow D_{s_n}$) and the vector of thresholds $\boldsymbol{\theta} = [\theta_1^*, \theta_2^*, \dots, \theta_n^*]^\top$ that maximizes the expected ranking gain on the finite set T . The implied optimization problem is given by the following equation:

$$(\mathbf{S}^*, \boldsymbol{\theta}^*) = \underset{(\mathbf{S}, \boldsymbol{\theta})}{\operatorname{argmax}} \{F(D_{\mathbf{S}}, T, \boldsymbol{\theta})\}, \quad (2.1)$$

where the ranking function $F(D_{\mathbf{S}}, T, \boldsymbol{\theta})$ can be defined as the expected ranking gain of $D_{\mathbf{S}}$ on T , that is

$$F_{AP}(D_{\mathbf{S}}, T, \boldsymbol{\theta}) = AP@k(\operatorname{rank}(y), \operatorname{rank}(D_{\mathbf{S}}(T, \boldsymbol{\theta})),$$

where, $\operatorname{rank}(y)$ is the actual ranking of the samples in T (i.e., samples with $y_i = 1$ are ranked higher than samples with $y_i = -1$), and $\operatorname{rank}(D_{\mathbf{S}}(T, \boldsymbol{\theta}))$ is the predicted ranking of the samples of cascade $D_{\mathbf{S}}$, $\boldsymbol{\theta}$ on T . $AP@k$ is the average precision in the top k samples.

Let $l \leq n$ refer to the number of variables $s_j \in \mathbf{S}$ whose value is different from -1 (i.e., l is the number of cascade stages that solution \mathbf{S} implies). The size of the search space related to the ordering of cascade stages is $\sum_{l=1}^n \binom{n}{l} l!$ (i.e., all index sequences for $l = 1$, all permutations of index sequences for $l = 2$, and similarly for all higher values of l , up to $l = n$). Furthermore, $\Theta \subset \mathbb{R}^n$ is the search space that consists of all the possible rejection thresholds for each stage of the cascade. To collect candidate threshold values, we apply each stage classifier on the training set T . Each of the M returned probability output scores constitutes a candidate threshold. The size of the search space equals M^n . Considering that this is a large search space, an exhaustive search cannot be practically applied. To solve the problem we propose the greedy search algorithm described below.

2.3.2.2 Problem Solution

Our algorithm finds the final solution by sequentially replacing at each iteration a simple solution (consisting of a cascade with a certain number of stages) with a more complex one (consisting of a cascade with one additional stage). Algorithm 2.1 presents the proposed greedy search algorithm that instantiates the proposed cascade (Figure 2.1). Figure 2.2 presents graphically the algorithm's steps. Let $\mathbf{S} = [s_1, s_2, \dots, s_n]^\top$ and $\boldsymbol{\theta} = [\theta_{s_1}, \theta_{s_2}, \dots, \theta_{s_n}]^\top$ represent a solution. Each variable s_1, s_2, \dots, s_n can take n possible values, from 1 to n or the value -1 which indicates that a stage is omitted. Each variable $\theta_{s_1}, \theta_{s_2}, \dots, \theta_{s_n}$ can take M possible values. Initially we set $s_j = -1$ for $j = 1, \dots, n$ and $\boldsymbol{\theta} = [0, 0, \dots, 0]^\top$ where $|\boldsymbol{\theta}| = n$. In the first step the algorithm optimizes \mathbf{S} with respect to s_n (Alg. 2.1: States 1–3) in order to build the solution:

$$\mathbf{S}_0 = [-1, -1, \dots, s_n]^\top, \boldsymbol{\theta}_0 = [0, 0, \dots, 0]^\top,$$

where according to (2.1),

$$s_n^* = \underset{s_n}{\operatorname{argmax}} \{F_{AP}(D_{\mathbf{S}_0}, T, \boldsymbol{\theta}_0)\} \quad (2.2)$$

and $\boldsymbol{\theta}_0^* = [0, 0, \dots, \theta_{s_n}^*], \theta_{s_n}^* = 0$. This can be interpreted as the optimal solution of $l = 1$ that maximizes (2.1). Then the algorithm, in iteration j (Alg. 2.1: States 4–7), assumes

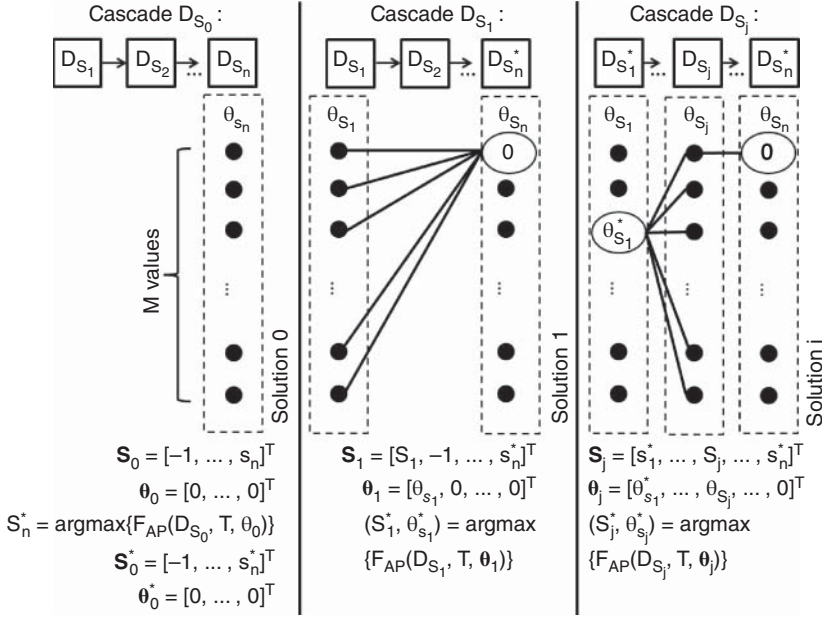


Figure 2.2 Threshold assignment and stage ordering of the proposed cascade architecture (Figure 2.1).

that it has solution with $l = j$, that is:

$$\begin{aligned} S_{j-1}^* &= [s_1^*, s_2^*, \dots, s_{j-1}^*, -1, -1, \dots, s_n^*]^T, \\ \theta_{j-1}^* &= [\theta_{s_1}^*, \theta_{s_2}^*, \dots, \theta_{s_{j-1}}^*, 0, 0, \dots, \theta_{s_n}^*]^T, \end{aligned}$$

and finds the pair of S_j and θ_j in one step as follows. It optimizes the pair of S_{j-1}^* and θ_{j-1}^* with respect to s_j and θ_j , respectively, in order to find the solution:

$$\begin{aligned} S_j &= [s_1^*, s_2^*, \dots, s_{j-1}^*, s_j, -1, -1, \dots, s_n^*]^T, \\ \theta_j &= [\theta_{s_1}^*, \theta_{s_2}^*, \dots, \theta_{s_{j-1}}^*, \theta_{s_j}, 0, 0, \dots, \theta_{s_n}^*]^T. \end{aligned}$$

According to (2.1):

$$(s_j^*, \theta_{s_j}^*) = \operatorname{argmax}_{(s_j, \theta_{s_j})} \{F_{AP}(D_{S_j}, T, \theta_j)\}. \quad (2.3)$$

The algorithm finds the pair of (s_j, θ_{s_j}) that optimizes (2.1). The complexity of this calculation equals $(n - j) \times M$. This corresponds to $n - j$ possible values that variable s_j can take in iteration j and M possible threshold rejection values that variable θ_{s_j} can take for every different instantiation of s_j . Finally, the optimal sequence S^* equals

$$S^* = \operatorname{argmax}_{S \in \{S_0^*, S_1^*, \dots, S_{n-1}^*\}} \{F_{AP}(D_S, T, \theta)\}, \quad (2.4)$$

which is the sequence that optimizes (2.1) within all the iterations of the algorithm (Algorithm 2.1, States 6–7). The optimal threshold vector θ^* is the vector connected to the optimal sequence S^* . Our algorithm focuses on the optimization of the complete cascade and not the optimization of each stage separately from the other stages. This is

Algorithm 2.1 Cascade stage ordering and threshold search

Input: Training set $T = \{x_i, y_i\}_{i=1}^M, y_i \in \{\pm 1\}$; n trained classifiers $D = \{D_1, D_2, \dots, D_n\}$

Output: (i) An index sequence S^* , of the ordering of cascade stages: $D_S^* : D_{s_1}^* \rightarrow D_{s_2}^* \rightarrow \dots \rightarrow D_{s_n}^*$. (ii) A vector of thresholds $\theta^* = [\theta_{s_1}^*, \theta_{s_2}^*, \dots, \theta_{s_n}^*]^\top$

Initialize: $S = [s_1, s_2, \dots, s_n]^\top, s_j = -1, j = 1, \dots, n, \theta = [0, 0, \dots, 0]^\top, |\theta| = n,$

1. $s_n^* = \operatorname{argmax}_{s_n} \{F_{AP}(D_{s_n}, T, \theta_0)\}$ (2.1),
- $S_0 = [-1, -1, \dots, s_n]^\top, \theta_0 = [0, 0, \dots, 0]^\top$
2. $\maxCost = F_{AP}(D_{S_0}, T, \theta_0^*),$
- $S_0^* = [-1, -1, \dots, s_n^*]^\top, \theta_0^* = [0, 0, \dots, \theta_{s_n}^*]^\top, \theta_{s_n}^* = 0$
3. $S^* = S_0^*, \theta^* = \theta_0^*$
- for** $j = 1$ to $n - 1$ **do**
 4. $(s_j^*, \theta_{s_j}^*) = \operatorname{argmax}_{(s_j, \theta_{s_j})} \{F_{AP}(D_{S_j}, T, \theta_j)\}$ (2.1),
 - $S_j = [\dots, s_j, -1, \dots, s_n^*]^\top, \theta_j = [\dots, \theta_{s_j}, 0, \dots, \theta_{s_n}^*]^\top$
 5. $cost = F_{AP}(D_{S_j}, T, \theta_j^*),$
 - $S_j^* = [\dots, s_j^*, -1, \dots, s_n^*]^\top, \theta_j^* = [\dots, \theta_{s_j}^*, 0, \dots, \theta_{s_n}^*]^\top$
 - if** $cost > \maxCost$ **then**
 6. $\maxCost = \max(cost, \maxCost)$
 7. $S^* = S_j^*, \theta^* = \theta_j^*$
 - end if**
- end for**

expected to give a better complete solution. Furthermore, the algorithm can be slightly modified to make the search more efficient. For example, at each iteration we can keep the p best solutions. However, this would increase the computational cost.

2.3.3 Multi-Task Learning for Concept Detection and Concept-Based Video Search

Having seen in the previous section the way that different video representations can be combined in a cascade architecture to improve the accuracy of the individual concept detectors, in this section we will focus on the way that more accurate individual concept detectors can be built by sharing knowledge across the concepts. MTL focuses exactly on the sharing of knowledge across different tasks. Assuming that some groups of concepts are expected to be related through some underlying structure, we can train an MTL classifier instead of using STL (e.g., SVMs). MTL is expected to improve the concept-based video search accuracy. Training MTL algorithms is similar to the training of the STL alternatives. Specifically, each type of feature serves as input to the MTL algorithm in order to train an MTL model. The scores returned from the MTL models for the same concept are fused using any existing fusion scheme, e.g, late fusion (averaging), cascades etc.

We chose the following MTL algorithms in order to examine different assumptions of task relatedness. The first method makes the strong assumption that all tasks are related. The next two methods consider the fact that some tasks may be unrelated [31, 32]:

- The L_1 -norm (or Lasso) regularized methods are widely used to introduce sparsity to the model and achieve the goal of reducing model complexity and feature learning [39]. An implementation that extends the L_1 -norm regularized STL to

MTL formulations is proposed in [40]. A common simplification of Lasso in MTL is that the parameter controlling the sparsity is shared among all tasks, assuming that different tasks share the same sparsity parameter. We will refer to this MTL method as Lasso-MTL in the following.

- The clustered MTL algorithm (CMTL) [31] uses a clustering approach to assign to the same cluster parameters of tasks that lie nearby in terms of their L2 distance.
- The adaptive MTL (AMTL) [32] decomposes the task parameters into a low-rank structure that captures task relations and a group-sparse structure that detects outlier tasks.

2.3.4 Exploiting Label Relations

To further improve the accuracy of a video concept detection system we can use a stacking architecture that exploits the concept relations in the last layer of the stack. Specifically, in this section we present a two-layer stacked model, initially proposed in [42], which uses appropriate multi-label classification methods able to capture concept relations. To build concept detectors the two-layer concept detection system presented in Figure 2.3 is employed. The first layer builds multiple independent concept detectors, using either STL or MTL approaches section 2.3.3. In the second layer of the stacking architecture, the fused scores from the first layer are aggregated in model vectors and refined using a multi-label learning algorithm that incorporates concept correlations. We choose the label powerset (LP) [41] transformation that models correlations among sets of more than two concepts. The typical stacking methods learn concept relations only by using the meta-level feature space, i.e., the learning of each concept in the second layer is still independent of the learning of the rest of the concepts. In contrast, our stacking architecture learns concept relations in the last layer of the stack both from the outputs of first-layer concept detectors and by modeling relations directly from the ground-truth annotation of the meta-level training set. This is achieved by instantiating our architecture with the LP [41] algorithm that searches for subsets of labels that appear together in the training set and consider each set as a

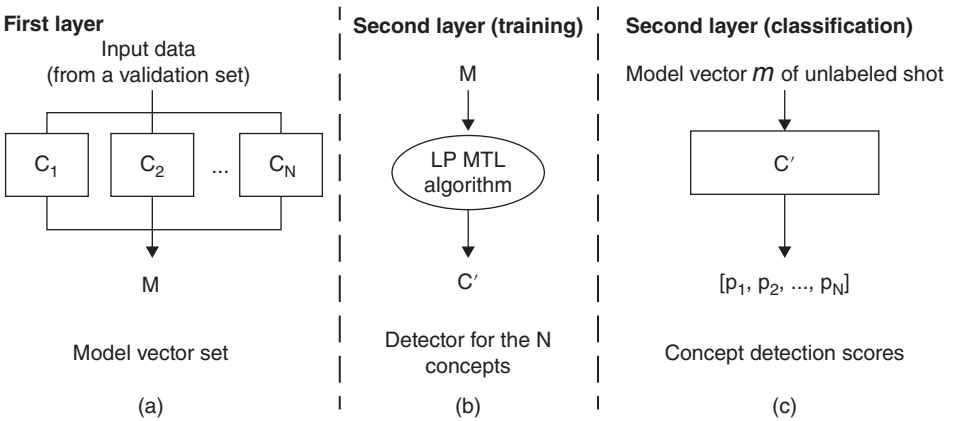


Figure 2.3 A two-layer stacking architecture instantiated with the LP [41] multi-label classification algorithm.

separate class in order to solve a multi-class problem. Of course, different multi-label classification algorithms that model different types of concept relations could also be used, such as the calibrated label ranking (CLR) algorithm that models relations between pairs of concepts and the ML- k NN algorithm [43] that models multiple relations in the neighbourhood of each testing instance.

2.3.5 Experimental Study

2.3.5.1 Dataset and Experimental Setup

The TRECVID semantic indexing (SIN) task [44] is a popular benchmarking activity that provides a large-scale dataset for video concept detection. The task is as follows. Given a set of shot boundaries for the SIN test dataset and a predefined list of concepts, participants are asked to return, for each concept, the top 2000 video shots from the test set, ranked according to the highest possibility of depicting the concept. The presence of each concept is assumed to be binary, i.e., it is either present or absent in the given standard video shot. If the concept was true for some frame within the shot, then it was true for the shot. This is a simplification adopted for the benefits it affords in pooling of results and approximating the basis for calculating recall. A list of 346 concepts is provided that has been collaboratively annotated by the participants and by Quaero annotators. In this study our experiments were performed on the TRECVID 2013 SIN dataset [44], which provides the following materials:

- a development set that contains roughly 800 hours of Internet archive videos comprising more than 500,000 shots
- a test set that contains roughly 200 hours of videos, comprising 112,677 shots
- shot boundaries (for both sets)
- a set of 346 concepts for training
- elements of ground-truth: some shots were collaboratively annotated.

The TRECVID 2013 SIN task was evaluated for 38 semantic concepts by calculating the mean extended inferred average precision (MXinfAP) at depth 2000 [45]. MXinfAP is an approximation of the mean average precision (MAP) that has been adopted by TRECVID [44] because it is suitable for the partial ground-truth that accompanies the TRECVID dataset [44].

In section 2.3.5.2 we evaluate the developed methods for each of the three general learning areas that aim to improve concept-based video search, i.e., cascades of classifiers (section 2.3.2), multi-task learning (section 2.3.3), and exploiting label relations (section 2.3.4). In our experiments we extracted the following features: three binary descriptors (ORB, RGB-ORB, and OpponentORB) and six non-binary descriptors (SIFT, RGB-SIFT, and OpponentSIFT; SURE, RGB SURE, and OpponentSURE). All of them were compacted using PCA and were subsequently aggregated using the VLAD encoding. In addition, we used features based on the pretrained CaffeNet-1k, GoogLeNet-1k, and 16-layer deep ConvNet [9] networks trained on 1000 ImageNet [46] categories. We applied each of these networks on the TRECVID keyframes and we used as a feature (i) the output of the second last fc layer of ConvNet (fc7), which resulted in a 4096-element vector, (ii) the output of the last fc layer of CaffeNet-1k (fc8), which resulted to a 1000-element, and (iii) the output of the last fc layer of GoogLeNet-1k (loss3/classifier). We refer to these features as CONV, CAFFE, and

GNET in the sequel, respectively. To train our base classifiers (i.e., LSVMs) for each concept, the TRECVID training set was used.

First, we compared the developed cascade (section 2.3.2) with five different ensemble combination approaches:

- i) Late-fusion with arithmetic mean [23].
- ii) The ensemble pruning method proposed by [24].
- iii) The simple cascade proposed by [27] with fixed ordering of the stages in terms of classifier accuracy. We refer to this method as cascade-thresholding.
- iv) A cascade with fixed ordering of the stages in terms of classifier accuracy, and the offline dynamic programming algorithm for threshold assignment proposed by [47]. In contrast to [47], which aims to improve the overall classification speed, we optimize the overall detection performance of the cascade in terms of AP. We refer to this method as cascade-dynamic in the sequel.
- v) A boosting-based approach (i.e., the multi-modal sequential SVM [48]). We refer to this method as AdaBoost.

For all the methods, except for late-fusion, which does not require this, the training set was also used as the validation set. With respect to the developed method, we calculated the AP for each candidate cascade at three different levels (i.e., for $k = 50, 100$ and equal to the number of training samples per concept) and averaged the results. Second, we assessed the usefulness of MTL, presented in section 2.3.3, instantiated with different MTL algorithms, and compared it with the typical STL approach. The MALSAR MTL library [40] was used as the source of the MTL algorithms. Third, we assessed the usefulness of the stacking architecture, presented in section 2.3.4. For this we further used the TRECVID 2012 test set (approx. 200 hours; 145,634 shots), which is a subset of the 2013 development set, as a validation set to train our multi-label classification algorithm for the second layer of the stack. Finally, we evaluate jointly all the proposed methods in a single concept detection system.

2.3.5.2 Experimental Results

Tables 2.1, 2.2, 2.3, 2.4 and 2.5 present the results of our experiments in terms of MXinfAP [45]. Table 2.1 presents the MXinfAP for the different types of features that were used by the algorithms of this section. Each line of this table was used as a cascade stage for the cascade-based methods (Table 2.2: M3, M4, M6). Specifically, stages that correspond to SIFT, SURF, and ORB consist of three base classifiers (i.e., for the grayscale descriptor and its two color variants), while the stages of DCNN features (CAFFE, CONV, GNET) consist of one base classifier each. For the late fusion methods (Table 2.2: M1, M2) and the boosting-based method (Table 2.2: M5), the corresponding base classifiers per line of Table 2.1 were first combined by averaging the classifier output scores and then the combined outputs of all lines were further fused together. We adopted this grouping of similar base classifiers as this was shown to improve the performance for all the methods in our experiments, increasing the MXinfAP by $\sim 2\%$. For M2 we replaced the genetic algorithm with an exhaustive search (i.e., to evaluate all $2^6 - 1$ possible classifier subsets) because this was more efficient for the examined number of classifiers.

Table 2.2 presents the performance of the developed cascade-based method (section 2.3.2) and compares it with other classifier combination methods. The second

Table 2.1 Performance (MXinfAP, %) for each of the stage classifiers used in the experiments (the MXinfAP for stage classifiers that are made of more than one base classifier are reported in parenthesis).

Stage classifier	MXinfAP	Base classifiers
ORBx3	17.91 (12.18,13.81,14.12)	ORB, RGB-ORB, OpponentORB
SURFx3	18.68 (14.71,15.49,15.89)	SURF, OpponentSURF, RGB-SURF
SIFTx3	20.23 (16.55,16.73,16.75)	SIFT, OpponentSIFT, RGB-SIFT
CAFFE	19.80	Last fully connected layer of CaffeNet
GNET	24.36	Last fully connected layer of GoogLeNet
CONV	25.26	Second last fully connected layer of ConvNet

Table 2.2 Performance (MXinfAP, %) for different classifier combination approaches.

	M1	M2	M3	M4	M5	M6
Stage classifiers	Late-fusion [23]	Ensemble pruning [24]	Cascade-thresholding [27]	Cascade-dynamic [47]	AdaBoost [48]	Cascade-proposed (section 2.3.2)
ORBx3, SURFx3, CAFFE, SIFTx3, GNET, CONV	29.84	29.74	29.79	29.84	29.70	29.96

Table 2.3 MXInfAP (%) for different STL and MTL methods, trained on the features of Table 2.1. Scores for the same concept were fused in terms of arithmetic mean using late-fusion.

	Single-task learning			Multi-task learning		
	LR	LSVM	KSVM	Lasso-MTL [39]	AMTL [32]	CMTL [31]
MXInfAP	27.56	29.84	29.29	31.15	30.3	29.51

Table 2.4 Performance (MXinfAP, %) for a typical single-layer concept detection pipeline and the developed two-layer stacking architecture instantiated with the LP algorithm. For the developed method we also report CPU times that refer to mean training time (in minutes) for all concepts, and application of the trained second-layer detectors on one shot of the test set (in milliseconds). Column (a) shows the results for detectors trained on raw features. Column (b) shows the results for detectors trained on AGSDA-reduced features. The relative improvement w.r.t. the typical approach is shown in parentheses.

	Dimensionality reduction		(c) Mean exec. time training/testing
	(a) N/A	(b) AGSDA	
Cascade	29.96	30.03	N/A
Proposed-LP	30.9 (+3.1%)	30.35 (+1.1%)	549.40/24.93

Table 2.5 MXinfAP for different configurations of our concept detection approach.

Dataset	Late-fusion STL	Cascade	Cascade+AGSDA	Cascade+AGSDA+MTL	Cascade+AGSDA+MTL+LP
SIN 2015	23.7	23.9	24.98	25.8*	26.03
SIN 2013	29.84	29.96	30.03	32.47*	32.57

column shows the stage classifiers that were considered, i.e., the evaluated system utilised six stage classifiers and all 12 types of features. The best results were reached by the proposed cascade, which outperforms all the other methods, reaching a MXinfAP of 29.96 %. Compared to the ensemble pruning method (M2) the results show that exploring the best ordering of visual descriptors on a cascade architecture (M6), instead of just combining subsets of them (M2), can improve the accuracy of video concept detection. In comparison to the other cascade-based methods (M3, M4) that utilize fixed stage orderings and different algorithms to assign the stage thresholds, the proposed cascade (M6) also shows small improvements in MXinfAP. These can be attributed to the fact that our method simultaneously searches for both optimal stage ordering and threshold assignment. These MXinfAP improvements of the proposed cascade, although small, are accompanied by considerable improvements in computational complexity, as discussed in section 2.3.5.3.

In Table 2.3 we evaluate the usefulness of using MTL, as presented in section 2.3.3, for concept-based video search. We performed comparisons across the following methods: (i) STL using LR, LSVM, and kernel SVM with radial kernel (KSVM) and (ii) MTL using Lasso-MTL [39], AMTL [32], and CMTL [31]. STL refers to the typical training of concept detectors, i.e., training independent classifiers per concept. All the features of Table 2.1 were used to train either STL or MTL detectors and late-fusion in terms of arithmetic mean was used to combine the scores from the different detectors for the same concept. According to Table 2.3 the best performance is achieved when the Lasso-MTL [39] algorithm is used. Lasso-MTL can define task relatedness on the parameters of the independently trained concept detectors, which is shown to perform better than all the STL approaches and also the compared MTL ones.

Table 2.4 shows the results of the two-layer stacking architecture presented in section 2.3.4 and compares it with the typical single-layer concept detection pipeline. To train our first layer classifiers we used all the features presented in Table 2.1 and combined them using the proposed cascade. We also experimented with dimensionality reduction of these features prior to using them as input to LSVMs. Specifically, the AGSDA method, presented in section 2.2.2, was used to derive a lower dimensional embedding of the original feature vectors. The features in the resulting subspace served as input to the LSVMs. Consequently, the first layer of the employed stacking consists of the independent detectors, either trained on the raw features (Table 2.4a) or trained on the AGSDA-reduced features (Table 2.4b), that are subsequently combined using the proposed cascade evaluated in Table 2.2. The same detectors were applied on a meta-learning validation set in order to construct model vectors that were introduced in the second layer. The second layer was instantiated with the LP [41] algorithm, and we refer to our method as P-LP.

P-LP outperforms the independent first-layer detectors, reaching an MXinfAP of 25.6%. LP considers each subset of labels (label sets) presented in the training set as a class of a multi-class problem, which seems to be helpful for the stacking architecture. In [42] we evaluated many more different multi-label learning algorithms for our two-layer stacking architecture and we compared these instantiations against BCBCF [35], DMF [33], BSBRM [34], and MCF [36], presented in section 2.3.4.

Table 2.5 evaluates the usefulness of sequentially introducing in the concept detection pipeline the three methods presented above (i.e., cascades of classifiers, multi-task learning, and exploitation of label relations), together with AGSDA, towards improving concept-based video search. The typical concept-detection system that uses STL and late-fusion to combine concept detectors trained on different features for the same concept is treated as our baseline. Then starting from a cascade architecture that more cleverly combines the different detectors, we continue by sequentially adding one more method to further improve concept detection accuracy. We present results for the TRECVID SIN 2013 dataset and also the TRECVID SIN 2015 dataset that consists of another test set and is evaluated on a different subset of the 346 available concepts. We observe that the proposed cascade performs slightly better in terms of MXinfAP compared to the late-fusion method, achieving 0.8% and 0.4% relative improvement for the SIN 2015 and SIN 2013 datasets, respectively. At the same time it is computationally less expensive during classification, as we will see in the next section. Dimensionality reduction with AGSDA leads to more discriminative features, which are indicated by an increase in the MXInfAP from 23.9% to 24.98% and 29.96% to 30.03% for the SIN 2015 and SIN 2013 datasets, respectively. Using the MTL-Lasso method in the concept-detection pipeline, instead of STL techniques, significantly outperforms all the previous methods. Finally, exploiting label relations results in further improvement. We conclude that the methods presented in this section, i.e., cascades of detectors, MTL, two-layer stacking architecture that exploits label relations, and dimensionality reduction using the AGSDA algorithm presented in section 2.2.2, are complementary and combining all of them in a concept detection system can boost concept-based video search, reaching the best overall MXInfAP of 26.03% and 32.57% for the SIN 2015 and SIN 2013 datasets, respectively. These numbers are among the best results for these specific datasets and, although seemingly low (since they are much lower than 100%), a qualitative analysis shows how good they are. For example, considering the concept “chair” that reaches an infAP of 32.84% on the SIN 2013 dataset, we observe in the resulting ranked list of concept-based retrieval results that among the top-20 retrieved keyframes all of them are positive, among the top-50 retrieved keyframes 46 are positive, and among the top-100 86 are positive. To investigate the statistical significance of the differences between the results of the various methods/combinations reported in Table 2.5 we used a paired t-test as suggested by [49]; in Table 2.5 the absence of * suggests statistical significance. We found that differences between the complete method (Cascade+AGSDA+MTL+LP) and all the other methods are significant (at 5% significance level), except for the run that combines Cascade+AGSDA+MTL. Investigating label correlations in a system that already combines cascades, AGSDA and MTL does not lead to significant improvement. However, LP is still a useful method that consistently improves concept-based video search in both of the considered datasets.

2.3.5.3 Computational Complexity

We continue the analysis of our results with the computational complexity of the different methods compared in Table 2.2 during the training and classification phase. Table 2.6 summarizes the computational complexity during the training phase. Let us assume that n stage classifiers need to be learned, M training examples are available for training the different methods, and Q is the quantization value, where $Q \leq M$. The late-fusion approach [23], which builds n models (one for each set of features), is the simplest one. Cascade-thresholding [27] follows, which evaluates n cascade stages in order to calculate the appropriate thresholds per stage. Cascade-dynamic [47] works in a similar fashion as cascade-thresholding, requiring a slightly higher number of evaluations. Cascade-proposed is the next least complex algorithm, requiring $Q(n(n+1)/2)$ classifier evaluations. Ensemble pruning [24] follows, requiring the evaluation of $2^n - 1$ classifier combinations. Finally, only AdaBoost requires the retraining of different classifiers, which depends on the complexity of the base classifier, in our case the SVM, making this method the computationally most expensive.

Table 2.7 presents the computational complexity of the proposed cascade-based method for the classification phase, and compares it with other classifier combination

Table 2.6 Training complexity: (a) the required number of classifier combinations during the training of different classifier combination approaches. and (b) the required number of classifiers to be retrained.

		Required classifier evaluations	Number of classifiers to be retrained
M1	Late-fusion [23]	–	–
M2	Ensemble pruning [24]	$(2^n - 1)M$	–
M3	Cascade-thresholding [27]	$\sum_{j=0}^n M_j, M_j \subseteq M_{j-1}$	–
M4	Cascade-dynamic [47]	$(n-2)Q^2$	–
M5	AdaBoost [48]	$M(n(n+1)/2)$	$n(n+1)/2$
M6	Cascade-proposed (section 2.3.2)	$Q(n(n+1)/2)$	–

Table 2.7 Relative amount of classifier evaluations (%) for different classifier combination approaches during the classification phase.

Stage classifiers	M1 Late-fusion [23]	M2 Ensemble pruning [24]	M3 Cascade-thresholding [27]	M4 Cascade-dynamic [47]	M5 AdaBoost [48]	M6 Cascade-proposed (section 2.3.2)
ORBx3, SURFx3, CAFFE SIFTx3, CONV, GNET	100	66.67	74.94	92.38	100	62.24

methods. We observe that the proposed algorithm reaches good accuracy while at the same time it is less computationally expensive than the other methods. Specifically, the best overall accuracy achieved 37.8% and 32.6% relative decrease in the number of classifier evaluations compared to the late-fusion alternative (Table 2.7: M1) and the cascade-dynamic alternative (Table 2.7: M4), respectively, which are the two most accurate methods after the proposed-cascade. Finally, we should note that the training for the proposed cascade is computationally more expensive than the training for the late-fusion and the cascade-dynamic methods. However, considering that training is only performed offline once, but classification will be repeated many times for any new input video, the latter is more important and this makes the reduction in the number of classifier evaluations that is observed in Table 2.7 for the proposed cascade very important.

With respect now to the two-layer stacking architecture, according to the last column of Table 2.4, one could argue that the proposed architecture requires a considerable amount of time. However, we should note here that extracting one model vector from one video shot, using the first-layer detectors for 346 concepts, requires approximately 3.2 minutes in our experiments, which is about three orders of magnitude slower than the proposed second-layer methods. As a result of the inevitable computational complexity of the first layer of the stack, the execution time that P-LP requires can be considered negligible. This is in sharp contrast to building a multi-label classifier directly from the low-level visual features of video shots, where the high requirements for memory space and computation time that the latter methods exhibit make their application to our dataset practically infeasible. Specifically, the computational complexity LP, when used in a single-layer architecture, depends on the complexity of the base classifier, in our case the LSVMs, and on the parameters of the learning problem (e.g., the number of training examples, and feature video dimensionality). LP would build a multi-class model, with the number of classes being equal to the number of distinct label sets in the training set; this is in order of N^2 in our dataset. Taking into consideration the dimensionality of the utilised feature vectors, using any such multi-label learning method in a single-layer architecture would require several orders of magnitude more computations compared to the BR alternative that we employ as the first layer in our presented stacking architecture. We conclude that the major obstacle to using multi-label classification algorithms in a one-layer architecture is the computation time requirement, and this finding further stresses the merit of using a multi-label stacking architecture. Finally, the training of the MTL algorithms, compared in Table 2.3, is computationally more expensive than the training of linear STL alternatives (e.g., LR, LSVM), but less expensive than the kernel SVM. Considering that training is performed offline only once, but classification will be repeated many times for any new input video, the latter is more important and MTL methods present the same classification time as the linear STL alternatives.

2.4 Methods for Event Detection and Event-Based Video Search

2.4.1 Related Work

There are several challenges associated with building an effective detector of video events. One of them is finding a video representation that reduces the gap between the

traditional low-level audio-visual features that can be extracted from the video and the semantic-level actors and elementary actions that are by definition the constituent parts of an event. In this direction, several works have shown the importance of using simpler visual concepts as a stepping stone for detecting complex events (e.g. [18, 50]). Another major challenge is to learn an association between the chosen video representation and the event or events of interest. For this, supervised machine learning methods are typically employed, together with suitably annotated training video corpora. While developing efficient and effective machine learning algorithms is a challenge in its own right, finding a sufficient number of videos that depict the event to use as positive training samples for training any machine learning method is not an easy feat. In fact, video event detection is even more challenging when the available positive training samples are limited, or even non-existent, that is, when one needs to train an event detector using only textual information that a human can provide about the event of interest.

Zero-example event detection is an active topic with many literature works proposing ways to build event detectors without any training samples and using solely the event’s textual description. Research towards this problem was triggered a few years ago when the TRECVID benchmark activity introduced the 0Ex task as a subtask of the Media Event Detection (MED) task [51]. A similar to zero-example event detection problem, known as zero-shot learning (ZSL), also appears in the image recognition task. A new unseen category, for which training data is not available, is asked to be detected in images [52–54]. It should be noted that although the two problems have many common properties, zero-example event detection is a more challenging problem as it focuses on more complex queries, where multiple actions, objects, and persons interact with each other compared to the simple object or animal classes that appear in ZSL [55]. The problem of zero-example event detection is typically addressed by transforming both the event textual description and the available videos into concept-based representations. Specifically, a large pool of concept detectors is used to annotate the videos with semantic concepts, the resulted vectors, also known as model vectors, contain the scores indicating the degree that each of the concepts is related to the video. The query description is analysed and the most related concepts from the pool are selected. Finally, the distance between the model vectors and the event concept vectors is calculated and the most related videos are retrieved [56–59].

In this work we present methods that solve the problem of event detection and event-based video search in two different cases. First when ground-truth annotated video examples are provided, we present how event detectors can be learned using the video positive examples that are available separately for each event (section 2.4.2). Second, when solely the textual description of event is given without any positive video examples, we show how event detectors can be learned using solely textual information for the examined events (section 2.4.3).

2.4.2 Learning from Positive Examples

The target of an event detection system is to learn a decision function $f : \mathcal{F} \rightarrow \{\pm 1\}$, where \mathcal{F} denotes the space where the video representations lie. f assigns a test video to the event class (labeled with the integer 1) or to the “rest of the world” class (labeled with the integer -1). For each event class, this is typically achieved using a training set

$\mathcal{X} = \{(\mathbf{x}_i, y_i) : \mathbf{x}_i \in \mathcal{F}, y_i \in \{0, \pm 1\}, i = 1, \dots, N\}$, where \mathbf{x}_i denotes the representation of the i th training video and y_i denotes the corresponding ground-truth label. Labels -1 and $+1$ correspond respectively to negative and positive training examples.

In order to train an event detector from positive video examples, we first utilized an extended and speeded-up version of our kernel subclass discriminant analysis [60, 61] for dimensionality reduction and then used a fast linear SVM (AGSDA+LSVM) in order to train an event detector.

Specifically, two types of visual information have been used for training the event detectors: motion features and DCNN-based features. We briefly describe the different visual modalities in the following:

- Each video is decoded into a set of keyframes at fixed temporal intervals (approximately 2 keyframes per second). We annotated the video frames based on 12988 ImageNet [46] concepts, 345 TRECVID SIN [62] concepts, 500 event-related concepts [63], 487 sport-related concepts [64], and 205 place-related concepts [65]. To obtain scores regarding the 12988 ImageNet concepts we used the pretrained GoogLeNet provided by [11]. We also experimented with a subset of the 12988 concepts; in order to do that we self-trained a GoogLeNet network [66] on 5055 ImageNet concepts (gnet5k). To obtain the scores regarding the 345 TRECVID SIN concepts and the 487 sport-related concepts we fine-tuned (FT) the gnet5k network on the TRECVID AVS development dataset and on the YouTube Sports-1M dataset [64], respectively. We also used the EventNet [63] that consists of 500 events and the Places205-GoogLeNet, which was trained on 205 scene categories of Places Database [65]. All the above networks were also used as feature generators, i.e., the output of one or more hidden layers was used as a global frame representation.
- For encoding motion information we use improved dense trajectories (DT) [67]. Specifically, we employ the following four low-level feature descriptors: histogram of oriented gradients (HOG), histogram of optical flow (HOF) and motion boundary histograms in both x (MBHx) and y (MBHy) directions. Hellinger kernel normalization is applied to the resulting feature vectors, followed by Fisher vector (FV) encoding with 256 Gaussian Mixture Model (GMM) codewords. Subsequently, the four feature vectors are concatenated to yield the final motion feature descriptor for each video in \mathbb{R}^{101376} .

The final feature vector representing a video is formed by concatenating the feature vectors derived for each visual modality (motion, model vectors), yielding a new feature vector in \mathbb{R}^{153781} .

2.4.3 Learning Solely from Textual Descriptors: Zero-Example Learning

In this section we present a method we developed in [68] that builds a fully automatic zero-example event detection system as presented in Figure 2.5. The developed system takes as input the event kit, i.e., a textual description of the event query, and retrieves the most related videos from the available event collection. We assume that the only knowledge available, with respect to each event class, is a textual description of it, which consists of a title, a free-form text, and a list of possible visual and audio cues, as in [69, 70]. Figure 2.4 shows an example of such a textual description for the event class *Attempting a bike trick*. To link this textual information with the visual content of the

Event Title: Attempting a bike trick.

Definition: One or more people attempt to do a trick on a bicycle, motorcycle, or other type of motorized bike. To count as a bike for purposes of this event, the vehicle must have two wheels (excludes unicycles, ATVs, etc).

Visual cues:

- **Scene:** outdoors, often in skate park, parking lot or street.
- **Objects/People:** person riding a bike, bike, ramps, helmet, concrete floor, audience.
- **Activities:** riding bike on one wheel, standing on top of bike, jumping with the bike (especially over or onto objects), spinning or flipping bike.

Audio cues:

- **Audio:** sounds of bike hitting surface during the trick, audience cheering.

Figure 2.4 The event kit text for the event class *Attempting a bike trick*.

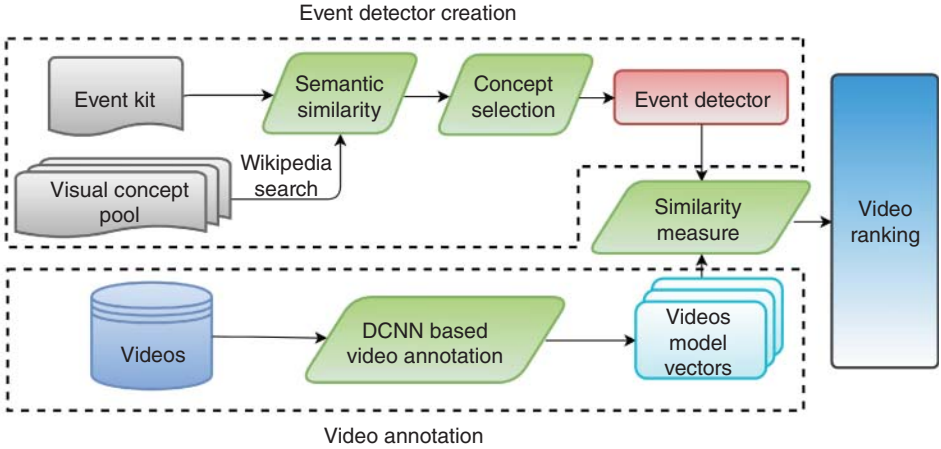


Figure 2.5 The proposed pipeline for zero-example event detection.

videos that we want to examine, similarly to [71, 72], we use (i) a pool of N_c concepts along with their titles and in some cases a limited number of subtitles (e.g., concept *bicycle-built-for-two* has the subtitles *tandem bicycle* and *tandem*) and (ii) a pretrained detector (based on DCNN output scores) for each concept.

Given the textual description of an event, our system first identifies N words or phrases that most closely relate to the event; this word-set is called the event language model (ELM). The ELM is based on the automatic extraction of word terms from the visual and audio cues of the event kit along with the title of the event. In parallel, for each of the X concepts of our concept pool, our framework similarly identifies M words or phrases: the concept language model (CLM) of the corresponding concept using the top-10 articles in Wikipedia and transforming this textual information into a bag of words (BoW) representation.

Subsequently, for each word in ELM and each word in each one of the CLMs we calculate the explicit semantic analysis (ESA) similarity [73] between them. For each CLM,

the resulting $N \times M$ distance matrix expresses the relation between the given event and the corresponding concept. In order to compute a single score expressing this relation, we apply to this matrix the Hausdorff distance. Consequently, a score is computed for each pair of ELM and CLM. The N_c considered concepts are ordered according to these scores (in descending order) and the K -top concepts along with their scores constitute our event detector.

In contrast to [59] and [58], where the number of selected concepts is fixed across the different events and motivated by statistical methods such as PCA [74], where a fraction of components are enough to efficiently or even better describe the data, we propose a statistical strategy that decides on the appropriate number of concepts k , where $k \leq k'$, that should be kept for an event query. First, we check if the event title is semantically close to any of the available concepts from the concept pool. If so, these concepts are used as the event detector. If this is not the case, our strategy orders the vector of concepts scores \mathbf{d}' in descending order, constructs an exponential curve, and then selects the first k concepts so that the corresponding area under the curve is $X\%$ of the total area under the curve. This procedure consequently returns a different number of selected concepts for different target events. For example for the event *Attempting a bike trick* the selected concepts are *ride a dirt bike*, *mountain biking*, *put on a bicycle chain*, and *ride a bicycle*, while for the event *Cleaning an appliance* only the concept *clean appliance* is selected. The final event detector is a k -element vector that contains the relatedness scores of the selected concepts.

In parallel, each video is decoded into as set of keyframes at fixed temporal intervals. Then, a set of pretrained concept-based DCNNs are applied to every keyframe and each keyframe is represented by the direct output of those networks. Finally, a video model vector is computed by averaging (in terms of arithmetic mean) the corresponding keyframe-level representations. Each element of a model vector indicates the degree that each of the predefined concepts appears in the video. The distance between an event detector and each of the video-level model vectors is calculated, and the h videos with the smallest distance are retrieved. As distance measure we choose the histogram intersection, which calculates the similarity of two discretized probability distributions and is defined as follows:

$$K_{\cap}(a, b) = \sum_{i=1}^k \min(a_i, b_i).$$

2.4.4 Experimental Study

2.4.4.1 Dataset and Experimental Setup

The multimedia event detection (MED) [75] task is part of TRECVID evaluation which is a popular benchmarking activity. The goal of the MED task is to support the creation of event detection and retrieval technologies that will permit users to define their own complex events and to quickly and accurately search large collections of multimedia clips. For this reason MED provides large-scale datasets for training and evaluation purposes as well as a set of events which consists of an event name, definition, explication (textual exposition of the terms and concepts), evidential descriptions, and illustrative video exemplars.

Table 2.8 MED 2016 Pre-specified events.

E021 - Attempting a bike trick	E031 - Beekeeping
E022 - Cleaning an appliance	E032 - Wedding shower
E023 - Dog show	E033 - Non-motorized vehicle repair
E024 - Giving directions to a location	E034 - Fixing musical instrument
E025 - Marriage proposal	E035 - Horse riding competition
E026 - Renovating a home	E036 - Felling a tree
E027 - Rock climbing	E037 - Parking a vehicle
E028 - Town hall meeting	E038 - Playing fetch
E029 - Winning a race without a vehicle	E039 - Tailgating
E030 - Working on a metal crafts project	E040 - Tuning a musical instrument

For learning from positive examples we used the PS-Training video set consisting of 2000 (80 hours) positive (or near-miss) videos as positive exemplars, and the Event-BG video set containing 5000 (200 hours) background videos as negative ones.

To evaluate both systems (i.e., learning from positive examples and zero-example learning) a common video dataset was used. We processed the MED16–EvalSub set consisting of 32,000 videos (960 hours). We evaluated all the methods on the 20 MED2016 [75] Pre-Specified events (E021-E040) for which event kits were provided (Table 2.8). We evaluated all the methods in terms of the mean average precision (MAP) and mean inferred average precision (mInfAP).

2.4.4.2 Experimental Results: Learning from Positive Examples

In this section, we validate the performance of the presented system for learning video event detectors from positive samples 2.4.2. We simulate two different case scenarios. In the first scenario only a few (10) positive video samples per event are available for training while in the second one, an abundance of training video samples are available (100 positive videos) for each individual event. Table 2.9 illustrates how the performance of our method is affected when different numbers of video samples are used.

Undoubtedly the plethora of positive video samples leads to significant performance improvement, i.e., the relative improvement is 45.3% in terms of MAP and 38% in terms of mInfAP@200%.

2.4.4.3 Experimental Results: Zero-Example Learning

In this section we evaluate the developed method for event-based video search in the case where only the event’s textual description is given (section 2.4.3. For our zero-example learning experiments we utilize two different concept pools as well as

Table 2.9 Learning from positive example results.

	MAP(%)	mInfAP@200(%)
10 positive videos	31.8	34.2
100 positive videos	46.2	47.5

Table 2.10 Zero-example learning results.

	MAP (%)	mInfAP@200 (%)
DCNN13K	14.6	12.2
DCNN14K	14.5	11.9
Train	16.2	14.2

an extension of the pipeline with an extra pseudo-relevance feedback step with online training in which the top retrieved videos are used as positive video samples and the AGSDA+LSVM method, as described in section 2.4.2, is used to train new event detectors.

- **DCNN13K:** In our first configuration we use the annotation from two different DCNNs: (i) the pretrained GoogLeNet provided by [66] trained on 12988 ImageNet concepts [46] and (ii) the EventNet [63] that consists of 500 events.
- **DCNN14K:** In this configuration, we use the annotation from five different DCNNs: (i) the pretrained GoogLeNet provided by [66] trained on 12988 ImageNet concepts [46], (ii) the GoogLeNet [11] self-trained on 5055 ImageNet concepts (gnet5k) and subsequently fine-tuned for 345 TRECVID SIN [62] concepts, (iii) the gnet5k network fine-tuned for 487 sport-related [64] concepts, (iv) the EventNet [63] that consists of 500 events, and (v) the Places205-GoogLeNet, trained on 205 scene categories [65].
- **Train:** In this configuration an online training stage is utilized using the top-10 retrieved videos from the first configuration as positive samples, and the learning procedure of section 2.4.2.

Table 2.10 compares the performance of these three different approaches for zero-example learning.

It is clear that adding the pseudo-relevance feedback step by using the top retrieved videos as positive samples has a significant impact on performance (the relative improvement is 10.96% in terms of MAP and 16.39% in terms of mInfAP@200).

However, the arbitrary addition of heterogeneous different concept pools leads to a small performance decrease due to noisy information that these concept pools add to the system (the percentage relative reduction in terms of mInfAP@200 is -2.5%).

Similarly to the concept detection case, a qualitative analysis shows that the event-based video search works satisfactorily. For example, considering the event *Horse-riding competition* that reaches an InfAP@200 of 13.9% and AP of 16.4% on the MED16-EvalSub dataset, we observe that among the top-20 retrieved videos 16 of them are positive, and similarly among the top-50 retrieved videos 32 are positive.

2.5 Conclusions

In this chapter we surveyed the literature and presented in more detail some of the methods that we have developed for concept-based and event-based video search. For concept-based video search we presented methods from three machine-learning areas (cascades of classifiers, multi-task learning, and exploitation of label relations) that

can improve the accuracy of concept-based video search and/or reduce computational complexity, in comparison to similar methods, thus enabling effective real-time video search. Our experiments on two large-scale datasets, i.e., TRECVID SIN 2013 and TRECVID SIN 2015, show the effectiveness and scalability of the presented methods, and especially how combining cascades of detectors, MTL, a two-layer stacking architecture, and dimensionality reduction with AGSDA in a concept detection system can boost concept-based video search. For the event-based video search, we presented methods for learning when ground-truth video data are available for a given event, as well as when only a textual description of events is given without any positive video examples. Our experiments on a large-scale video event dataset (MED16–EvalSub) showed that the exploitation of a larger number of video samples generally leads to better performance, but also, in the more realistic scenario of zero-example event detection problem, we presented a state-of-the-art method that achieves very promising results.

2.6 Acknowledgments

This work was supported by the EU's Horizon 2020 research and innovation programme under grant agreements H2020-693092 MOVING, H2020-687786 InVID, and H2020-732665 EMMA.

References

- 1 Snoek, C.G.M. and Worring, M. (2009) Concept-Based Video Retrieval. *Foundations and Trends in Information Retrieval*, 2 (4), 215–322.
- 2 Sidiropoulos, P., Mezaris, V., and Kompatsiaris, I. (2014) Video tomographs and a base detector selection strategy for improving large-scale video concept detection. *IEEE Transactions on Circuits and Systems for Video Technology*, 24 (7), 1251–1264.
- 3 Rublee, E., Rabaud, V., Konolige, K., and Bradski, G. (2011) ORB: An efficient alternative to SIFT or SURF, in *IEEE International Conference on Computer Vision*, IEEE pp. 2564–2571.
- 4 Lowe, D.G. (2004) Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60 (2), 91–110.
- 5 Bay, H., Tuytelaars, T., and Van Gool, L. (2006) Surf: Speeded up robust features, in *ECCV, LNCS*, vol. 3951, Springer, pp. 404–417.
- 6 Van de Sande, K.E.A., Gevers, T., and Snoek, C.G.M. (2010) Evaluating color descriptors for object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32 (9), 1582–1596.
- 7 Csurka, G. and Perronnin, F. (2011) Fisher vectors: Beyond bag-of-visual-words image representations, in *Computer Vision, Imaging and Computer Graphics. Theory and Applications, Communications in Computer and Information Science*, vol. 229 (eds P. Richard and J. Braz), Springer, Berlin, pp. 28–42.
- 8 Jegou, H. et al. (2010) Aggregating local descriptors into a compact image representation, in *IEEE on Computer Vision and Pattern Recognition (CVPR 2010)*, San Francisco, CA, pp. 3304–3311.

- 9 Simonyan, K. and Zisserman, A. (2014) Very deep convolutional networks for large-scale image recognition. arXiv technical report.
- 10 Krizhevsky, A., Ilya, S., and Hinton, G. (2012) Imagenet classification with deep convolutional neural networks, in *Advances in Neural Information Processing Systems (NIPS 2012)*, Curran Associates, Inc., pp. 1097–1105.
- 11 Szegedy, C. et al. (2015) Going deeper with convolutions, in *CVPR 2015*. URL <http://arxiv.org/abs/1409.4842>.
- 12 Safadi, B., Derbas, N., Hamadi, A., Budnik, M., Mulhem, P., and Qu, G. (2014) LIG at TRECVID 2014: Semantic Indexing, in *TRECVID 2014 Workshop*, Gaithersburg, MD, USA.
- 13 Snoek, C.G.M., Sande, K.E.A.V.D., Fontijne, D., Cappallo, S., Gemert, J.V., and Habibian, A. (2014) MediaMill at TRECVID 2014: Searching Concepts, Objects, Instances and Events in Video, in *TRECVID 2014 Workshop*, Gaithersburg, MD, USA.
- 14 Yosinski, J., Clune, J., Bengio, Y., and Lipson, H. (2014) How transferable are features in deep neural networks? *Proceedings of the 27th International Conference on Neural Information Processing Systems (NIPS'12)*, Volume 2, Montreal, Canada, pp. 3320–3328, MIT Press, Cambridge, MA.
- 15 Pittaras, N., Markatopoulou, F., Mezaris, V., et al. (2017) *Comparison of Fine-Tuning and Extension Strategies for Deep Convolutional Neural Networks*, Springer, Cham, pp. 102–114.
- 16 Gkalelis, N., Mezaris, V., Kompatsiaris, I., and Stathaki, T. (2013) Mixture subclass discriminant analysis link to restricted Gaussian model and other generalizations. *IEEE Transactions on Neural Networks and Learning Systems*, 24 (1), 8–21.
- 17 Agullo, E., Augonnet, C., Dongarra, J., et. al (2010) Faster, cheaper, better – a hybridization methodology to develop linear algebra software for GPUS, in *GPU Computing Gems*, Volume 2, Wen-mei W. Hwu (ed.), Morgan Kaufmann.
- 18 Gkalelis, N. and Mezaris, V. (2014) Video event detection using generalized subclass discriminant analysis and linear support vector machines, in *Proceedings of the International Conference on Multimedia Retrieval*, ACM, p. 25.
- 19 Schölkopf, B. and Smola, A.J. (2002) *Learning with kernels: support vector machines, regularization, optimization, and beyond*, MIT Press.
- 20 Athanasopoulos, A., Dimou, A., Mezaris, V., and Kompatsiaris, I. (2011) GPU acceleration for support vector machines, in *WIAMIS 2011: 12th International Workshop on Image Analysis for Multimedia Interactive Services*, Delft, The Netherlands, 13–15 April.
- 21 Arestis-Chartampilas, S., Gkalelis, N., and Mezaris, V. (2015) GPU accelerated generalised subclass discriminant analysis for event and concept detection in video, in *Proceedings of the 23rd ACM International Conference on Multimedia*, ACM, pp. 1219–1222.
- 22 Arestis-Chartampilas, S., Gkalelis, N., and Mezaris, V. (2016) AKSDA-MSVM: A GPU-accelerated multiclass learning framework for multimedia, in *Proceedings of the 2016 ACM on Multimedia Conference*, ACM, pp. 461–465.
- 23 Strat, S.T., Benoit, A., Bredin, H., Quenot, G., and Lambert, P. (2012) Hierarchical late fusion for concept detection in videos, in *European Conference on Computer Vision (ECCV) 2012. Workshops and Demonstrations, Lecture Notes in Computer Science*, vol. 7585, Springer, pp. 335–344.

- 24 Sidiropoulos, P., Mezaris, V., and Kompatsiaris, I. (2014) Video tomographs and a base detector selection strategy for improving large-scale video concept detection. *IEEE Transactions on Circuits and Systems for Video Technology*, 24 (7), 1251–1264, doi: 10.1109/TCSVT.2014.2302554.
- 25 Nguyen, C., Vu Le, H., and Tokuyama, T. (2011) Cascade of multi-level multi-instance classifiers for image annotation, in *KDIR'11*, pp. 14–23.
- 26 Cheng, W.C. and Jhan, D.M. (2011) A cascade classifier using adaboost algorithm and support vector machine for pedestrian detection, in *IEEE International Conference on SMC*, pp. 1430–1435, doi: 10.1109/ICSMC.2011.6083870.
- 27 Markatopoulou, F., Mezaris, V., and Patras, I. (2015) Cascade of classifiers based on binary, non-binary and deep convolutional network descriptors for video concept detection, in *2015 IEEE International Conference on Image Processing (ICIP)*, pp. 1786–1790, doi: 10.1109/ICIP.2015.7351108.
- 28 Mousavi, H., Srinivas, U., Monga, V., Suo, Y., et al. (2014) Multi-task image classification via collaborative, hierarchical spike-and-slab priors, in *Proceedings of the IEEE International Conference on Image Processing (ICIP 2014)*, pp. 4236–4240.
- 29 Daumé, H., III (2009) Bayesian multitask learning with latent hierarchies, in *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence (UAI '09)*, AUAI Press, Arlington, Virginia, US, pp. 135–142.
- 30 Argyriou, A., Evgeniou, T., and Pontil, M. (2008) Convex multi-task feature learning. *Machine Learning*, 73 (3), 243–272.
- 31 Zhou, J., Chen, J., and Ye, J. (2011) Clustered multi-task learning via alternating structure optimization, in *Advances in Neural Information Processing Systems*. J. Shawe-Taylor, R.S. Zemel, P.L. Bartlett, F. Pereira and K.Q. Weinberger (eds). pp. 702–710, Curran Associates, Inc., available at <http://papers.nips.cc/paper/4292-clustered-multi-task-learning-via-alternating-structure-optimization.pdf>.
- 32 Sun, G., Chen, Y., Liu, X., and Wu, E. (2015) Adaptive multi-task learning for fine-grained categorization, in *Proceedings of the IEEE International Conference on Image Processing (ICIP 2015)*, pp. 996–1000.
- 33 Smith, J., Naphade, M., and Natsev, A. (2003) Multimedia semantic indexing using model vectors, in *2003 International Conference on Multimedia and Expo. (ICME)*, IEEE, NY, pp. 445–448, doi: 10.1109/ICME.2003.1221649.
- 34 Tsoumakas, G., Dimou, A., Spyromitros, E., et al. (2009) Correlation-based pruning of stacked binary relevance models for multi-label learning, in *Proceedings of the 1st International Workshop on Learning from Multi-Label Data*, pp. 101–116.
- 35 Jiang, W., Chang, S.F., and Loui, A.C. (2006) Active context-based concept fusion with partial user labels, in *IEEE International Conference on Image Processing*, IEEE, NY.
- 36 Weng, M.F. and Chuang, Y.Y. (2012) Cross domain multicue fusion for concept-based video indexing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34 (10), 1927–1941.
- 37 Qi, G.J. et al. (2007) Correlative multi-label video annotation, in *Proceedings of the 15th International Conference on Multimedia*, ACM, NY, pp. 17–26.
- 38 Markatopoulou, F., Mezaris, V., and Patras, I. (2016) *Ordering of Visual Descriptors in a Classifier Cascade Towards Improved Video Concept Detection*, Springer International Publishing, pp. 874–885.

- 39 Tibshirani, R. (1996) Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, 267–288.
- 40 Zhou, J., Chen, J., and Ye, J. (2011) *MALSAR: Multi-tAsk Learning via Structural Regularization*, Arizona State University.
- 41 Tsoumakas, G., Katakis, I., and Vlahavas, I. (2010) Mining multi-label data, in *Data Mining and Knowledge Discovery Handbook*, Springer, Berlin, pp. 667–686.
- 42 Markatopoulou, F., Mezaris, V., Pittaras, N., and Patras, I. (2015) Local features and a two-layer stacking architecture for semantic concept detection in video. *IEEE Transactions on Emerging Topics in Computing*, 3 (2), 193–204, doi: 10.1109/TETC.2015.2418714.
- 43 Zhang, M.L. and Zhou, Z.H. (2007) ML-KNN: A lazy learning approach to multi-label learning. *Pattern Recognition*, 40 (7), 2038–2048, doi: 10.1016/j.patcog.2006.12.019.
- 44 Over, P., Awad, G., Fiscus, J., Sanders, G., and Shaw, B. (2013) Trecvid 2013 – an overview of the goals, tasks, data, evaluation mechanisms and metrics, in *Proceedings of TRECVID 2013*, NIST, USA.
- 45 Yilmaz, E., Kanoulas, E., and Aslam, J.A. (2008) A simple and efficient sampling method for estimating AP and NDCG, in *31st ACM SIGIR International Conference on Research and Development in Information Retrieval*, ACM, USA, pp. 603–610.
- 46 Russakovsky, O., Deng, J., Su, H., and et al. (2015) ImageNet large-scale visual recognition challenge. *International Journal of Computer Vision*, 115 (3), 211–252, doi: 10.1007/s11263-015-0816-y.
- 47 Chellapilla, K., Shilman, M., and Simard, P. (2006) Combining multiple classifiers for faster optical character recognition, in *7th International Conference on Document Analysis Systems*, Springer, Berlin, pp. 358–367.
- 48 Bao, L., Yu, S.I., and Hauptmann, A. (2011) CMU-informedia @ TRECVID 2011 semantic indexing, in *TRECVID 2011 Workshop*, Gaithersburg, MD, USA.
- 49 Blanken, H.M., de Vries, A.P., Blok, H.E., and Feng, L. (2005) *Multimedia Retrieval*, Springer, Berlin, Heidelberg, NY.
- 50 Gkalelis, N., Mezaris, V., Dimopoulos, M., Kompatsiaris, I., and Stathaki, T. (2013) Video event detection using a subclass recoding error-correcting output codes framework, in *IEEE International Conference on Multimedia and Expo (ICME)*, IEEE, pp. 1–6.
- 51 Over, P., Fiscus, J., Sanders, G., et al. (2015) TRECVID 2015 – an overview of the goals, tasks, data, evaluation mechanisms and metrics, in *TRECVID 2015 Workshop*, NIST, USA.
- 52 Elhoseiny, M., Saleh, B., and Elgammal, A. (2013) Write a classifier: Zero-shot learning using purely textual descriptions, in *IEEE International Conference on Computer Vision (ICCV)* pp. 2584–2591.
- 53 Fu, Z., Xiang, T., Kodirov, E., and Gong, S. (2015) Zero-shot object recognition by semantic manifold distance, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2635–2644.
- 54 Norouzi, M., Mikolov, T., Bengio, S., et al. (2013) Zero-shot learning by convex combination of semantic embeddings. *arXiv preprint arXiv:1312.5650*.
- 55 Jiang, Y.G., Bhattacharya, S., Chang, S.F., and Shah, M. (2012) High-level event recognition in unconstrained videos. *International Journal of Multimedia Information Retrieval*, pp. 1–29.

- 56 Habibian, A., Mensink, T., and Snoek, C.G. (2014) Videostory: A new multimedia embedding for few-example recognition and translation of events, in *Proceedings of the ACM International Conference on Multimedia*, ACM, pp. 17–26.
- 57 Elhoseiny, M., Liu, J., Cheng, H., Sawhney, H., and Elgammal, A. (2015) Zero-shot event detection by multimodal distributional semantic embedding of videos. *arXiv preprint arXiv:1512.00818*.
- 58 Lu, Y.J., Zhang, H., de Boer, M., and Ngo, C.W. (2016) Event detection with zero example: select the right and suppress the wrong concepts, in *Proceedings of the 2016 ACM International Conference on Multimedia Retrieval*, ACM, pp. 127–134.
- 59 Tzelepis, C., Galanopoulos, D., Mezaris, V., and Patras, I. (2016) Learning to detect video events from zero or very few video examples. *Image and Vision Computing*, 53, 35–44.
- 60 Gkalelis, N., Mezaris, V., Kompatsiaris, I., and Stathaki, T. (2013) Mixture subclass discriminant analysis link to restricted Gaussian model and other generalizations. *IEEE Transactions on Neural Networks and Learning Systems*, 24 (1), 8–21.
- 61 Gkalelis, N. and Mezaris, V. (2014) Video event detection using generalized subclass discriminant analysis and linear support vector machines, in *International Conference on Multimedia Retrieval (ICMR '14)*, Glasgow, UK, April 01–04, p. 25.
- 62 Smeaton, A.F., Over, P., and Kraaij, W. (2009) High-level feature detection from video in TRECVID: a 5-year retrospective of achievements, in *Multimedia Content Analysis, Theory and Applications* (ed. A. Divakaran), Springer Verlag, Berlin, pp. 151–174.
- 63 Ye, G., Li, Y., Xu, H., Liu, D., and Chang, S.F. (2015) Eventnet: A large scale structured concept library for complex event detection in video, *Proceedings of the 23rd ACM International Conference on Multimedia, MM '15*, Brisbane, Australia, pp. 471–480, available at <http://doi.acm.org/10.1145/2733373.2806221>, doi: 10.1145/2733373.2806221.
- 64 Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., and Fei-Fei, L. (2014) Large-scale video classification with convolutional neural networks, *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1725–1732, IEEE.
- 65 Zhou, B., Lapedriza, A., Xiao, J., et al. (2014) Learning deep features for scene recognition using places database, in *Proceedings of the 27th International Conference on Neural Information Processing Systems (NIPS '14)*, Volume 1, Montreal, Canada, pp. 487–495.
- 66 Mettes, P., Koelma, D., and Snoek, C. (2016) The imagenet shuffle: Reorganized pre-training for video event detection. *arXiv preprint arXiv:1602.07119*.
- 67 Wang, H. and Schmid, C. (2013) Action recognition with improved trajectories, in *IEEE International Conference on Computer Vision*, Sydney, Australia.
- 68 Galanopoulos, D., Markatopoulou, F., Mezaris, V., and Patras, I. (2017) Concept language models and event-based concept number selection for zero-example event detection, in *Proceedings of the 2017 ACM on International Conference on Multimedia Retrieval*, ACM.
- 69 Younessian, E., Mitamura, T., and Hauptmann, A. (2012) Multimodal knowledge-based analysis in multimedia event detection, in *Proceedings of the 2nd ACM International Conference on Multimedia Retrieval*, ACM, pp. 51:1–51:8.

- 70 Jiang, L., Mitamura, T., Yu, S.I., and Hauptmann, A.G. (2014) Zero-example event search using multimodal pseudo relevance feedback, in *Proceedings of International Conference on Multimedia Retrieval*, ACM, p. 297.
- 71 Wu, S., Bondugula, S., Luisier, F., Zhuang, X., and Natarajan, P. (2014) Zero-shot event detection using multi-modal fusion of weakly supervised concepts, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR) IEEE* pp. 2665–2672.
- 72 Yu, S.I., Jiang, L., Mao, Z., Chang, X., et al. (2014) Informedia at TRECVID 2014 MED and MER, in *NIST TRECVID Video Retrieval Evaluation Workshop*.
- 73 Gabrilovich, E. and Markovitch, S. (2007) Computing semantic relatedness using Wikipedia-based explicit semantic analysis, in *Proceedings of the 20th International Joint Conference on Artificial Intelligence, IJCAI'07, Hyderabad, India*, pp. 1606–1611, Morgan Kaufmann Publishers Inc., San Francisco, CA.
- 74 Jolliffe, I. (2002) *Principal Component Analysis*, Wiley Online Library.
- 75 Awad, G., Fiscus, J., Michel, M., et al. (2016) Trecvid 2016: Evaluating video search, video event detection, localization, and hyperlinking, in *Proceedings of TRECVID*.

3

Big Data Multimedia Mining: Feature Extraction Facing Volume, Velocity, and Variety

Vedhas Pandit, Shahin Amiriparian, Maximilian Schmitt, Amr Mousa and Björn Schuller

3.1 Introduction

With several hundred hours of naturalistic, in-the-wild videos and music being uploaded to the web per minute and millions of short texts being uploaded every day on social media, the *big data* era brings a plethora of opportunities yet also challenges to the field of *multimedia mining*. A modern multimedia mining system needs to be able to handle large databases with varying formats at extreme speeds. These three attributes, *volume*, *velocity* and *variety*, together define big data primarily. After a general introduction to the topic highlighting the big data challenges in terms of the three named Vs, we give an insight into traditional techniques and deep learning methodologies to cope with the scalability challenges in all these three respects. The inherent qualities of the data driven deep learning approach – which make it a promising candidate in terms of scalability – are then discussed in detail, along with a brief introduction to its constituent components and different state-of-the-art architectures. To give some insight into the actual effectiveness of the deep learning method for feature extraction, we present the latest original research results of a showcase *big data multimedia mining* task by evaluating the pretrained CNN-based feature extraction through process parallelization, providing insight into the effectiveness and high capability of the proposed approach.

The internet and smart devices today, coupled with social media and e-commerce avenues, have made data abundant, ubiquitous and far more valuable. No matter what activity one is involved in at any time of the day, whether watching TV, jogging or just stuck in traffic, each activity can create a digital trace. The upsurge in social media users (e.g., Facebook, YouTube, and Twitter), with increasingly diverse, huge amounts of content uploaded every second continuously from all over the world, has made *multimedia big data* far more relevant than ever before. This includes a huge variety of photographs, sketches, home videos, music content, live video streams, news bulletins, product reviews, reviews of local businesses and tourist places, tweets

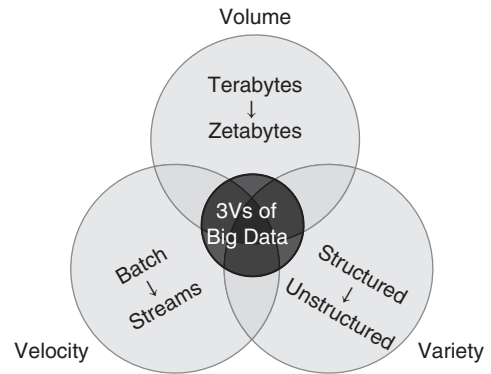
and posts with one's political or social views, movies, gameplays, and podcasts, to name just a few. Organizations today, from industries to political parties, rely on such data to get the low-down on current trends and assess their future strategies [1]. All sorts of users can benefit from big data multimedia mining for all kinds of novel services.

All this data comes from a vast number of sources all the time from across the globe. Data analysis and feature extraction methods need to cope with these astounding rates and quantities of data streams. Data-mining tools first need to translate the data into a native representation that can be worked on, then into a meaningful representation that can effectively be handled for further data-mining and querying, which, in most cases, is in the form of feature vectors. The feature extraction or data abstraction encapsulates the essential information in a compact form suitable for certain tasks. Because what qualifies as "suitable" or "essential" can vary from task to task, this feature extraction step is often governed by typicality of the data-mining queries subject to the data modality. More often than not, therefore, the tools also need to compute feature transformations that enhance their usability by improving the discriminative and predictive scope.

Just as the term implies, one of the most defining characteristics of multimedia big data is the enormous *volume* of data in terms of both the feature dimensions and the number of instances. This defining quality, and thus availability of such a quantity of data, becomes a game changer for businesses and mining such data through a network effect. As the number of instances or the number of features go up, however, it can become exponentially harder to make predictions with the same level of accuracy. This is one of the most commonly encountered problem in the machine learning literature, called the *curse of dimensionality*. For example, the popular distance measures used to better understand data, such as the Euclidean distance, fail when the dimensionality of the data is too large. This is because with too many co-ordinates in use there is (only) a little difference in the distances between different pairs of data samples. Irrelevant feature dimensions further add to the unreliability of this similarity or the distance measure, as each individual dimension affects the distance computation directly. Ironically, even the thousands of instances existing in 100-dimensional space is very sparse data. The dataset may appear like a bunch of random points in space with no specific underlying pattern. To have any better prediction capability, one will likely need millions of instances, thanks to the high dimensionality of the feature vectors (100s in this case). A high volume of data also necessitates higher processing memory and storage capabilities.

Also, as discussed earlier, the tools first need to translate the data into native representation, and then into its abstraction as the features, before any querying or data-mining can even begin [2]. Storing the data as native representation itself requires some non-zero time. In view of the enormous *velocity* of the data streams as discussed previously, while it is unreasonable to expect to finish even the storing of data, fast processing and encapsulation become rather a necessity in the multimedia big data context. In benchmarking studies on big data, therefore, computation time is often the prime performance measure.

The ubiquity of data generators presents us with another challenge. Because the data is gathered from a huge range of devices and sources, and features a rich diversity in terms of multimedia formats, also in part due to manifold different multimedia codecs in use these days, it is often an alphabet soup of data with a *variety* of file formats and hierarchical structures. This is especially true for multimedia data where it also features multitudes of modalities such as images, videos, audios, documents, tweets, binary system

Figure 3.1 The Vs of big data.

logs, graphs, maps, overlaid live traffic or weather data and so on. Affordable new technology devices, such as Kinect, continue to introduce new modalities of data. More specifically, in the case of Kinect-like devices with 3D sensing technology, the data also consists of 3D point clusters, varying in space and in time [3].

In summary, all machine learning approaches rely heavily on the features that represent the data. A scalable data-mining approach thus requires that all of its different components are able to handle huge volume, velocity, and variety in the data (Figure 3.1) – right from the feature extraction step.

In section 3.2, we first discuss the common strategies adopted to make data-mining scalable in terms of volume and velocity, when the variety of the data has been duly considered, i.e., when the framework to represent the data in a consistent form is in place just as necessary. Next, in section 3.3, we discuss “scalability through feature engineering,” which is just the process of intelligently picking the most relevant features going by the data modality and common queries. We also discuss the popular feature transformation methodologies and the contexts in which each of those are ideal. This becomes highly relevant for *model-based* approaches in particular, where the queries are attended to using explicit heuristics on the features. Section 3.4 introduces an implicit feature and representation learning paradigm, also called the *data-driven* approach, of deep learning, where the most relevant features are implicitly learnt by establishing mapping between the inputs and outputs through nonlinear functions. We argue that the inherent qualities of this model make it a great candidate for highly scalable machine learning as well as a feature extraction paradigm. A few of these qualities are (i) high flexibility in terms of the number of simultaneous outputs and the variability in terms of what each can represent, (ii) the breadth of high-level concepts it can model, e.g., the temporal and spatial correlations along with the intertwined contexts, (iii) the high modularity and integrability of the models, (iv) the wide scope for innovation through mixing and matching of various model topologies and the constituent elements, (v) the scope for velocity scalability through parallelization of the recursive and repeated elements and functions in the model hierarchy, which are also its indispensable components, such as the activation functions and kernels. We also discuss the key elements, most common architectures, and state-of-the-art methods that have evolved through this mix-and-match approach. Keeping the uninitiated reader in mind, there are detailed graphical illustrations accompanying the text to help easy intuitive understanding. We present benchmarking experiments in section 3.5 on testing; for the very first time, the runtimes of pretrained CNN-based feature

extraction on audio data, with and without process parallelization, are presented. Finally, in section 3.6 we present our concluding remarks.

3.2 Scalability through Parallelization

Due to the high volume and velocity of the data, the main challenges posed by big (multimedia) data are the data processing overheads and the added memory requirements. To meet these challenges, the data-mining approach needs to be highly scalable. The common strategies adapted for scalability fall into two categories: (i) improve the scalability of the machine learning algorithm itself using, for example, kernel approximations, parallelization of the processes or the data, or a combination of all three methods, and (ii) reduce the dimensionality of the data by generating the most compact and useful data representation possible, alleviating formidable memory and processing requirements, also called *feature engineering*.

For the first approach, for example, one of the most widely used machine learning algorithm is the support vector machine (SVM). An SVM (Figure 3.2) separates different clusters in the data using hyperplanes (linear kernel), hypersurfaces (polynomial kernel), or hyperspheres (radial basis function kernel). Being such a popular classical approach, several implementations parallelising the SVM process chain have been proposed [4–8]. Scalable implementations of several other approaches also exist, e.g., the random forest [10, 11] and or linear discriminant analysis [12, 13].

There are two main ways in which parallelization is achieved (Figure 3.3). These two parallelization techniques, discussed next, could also be combined.

3.2.1 Process Parallelization

In this approach, an algorithm is split into different smaller tasks, and these tasks are divided among the computing machines for faster processing. This way, parts of the program are executed simultaneously on different processors, and the program takes much less time to finish.

3.2.2 Data Parallelization

In this approach, the data are split into different batches. These data batches are then sent to the different processing units available, all of which house the same set of execution

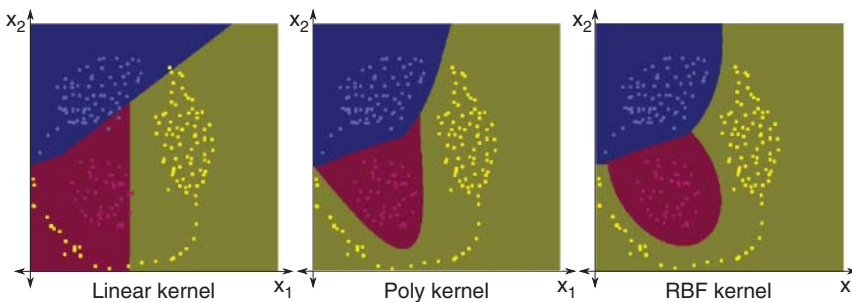


Figure 3.2 SVM kernels illustrated using web-based implementation of LibSVM [9].

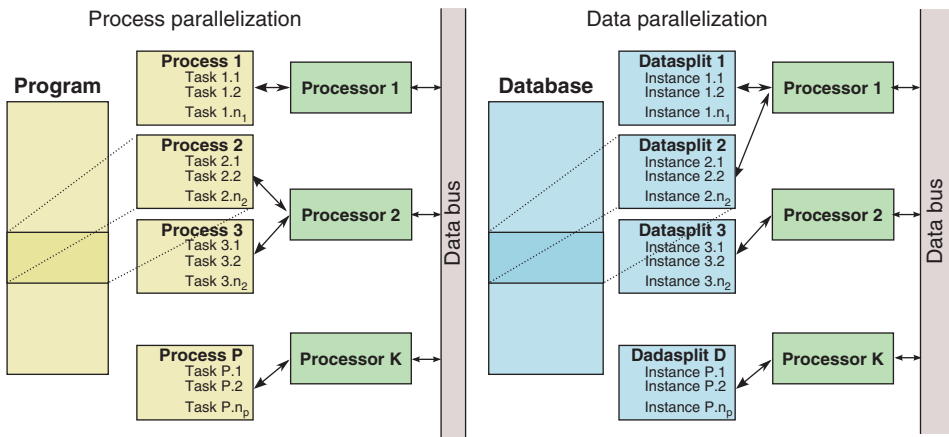


Figure 3.3 In the process parallelization scheme, a task is split into difference processes to be handled by different processing units simultaneously. In the data parallelization paradigm, the input data is split into different chunks or batches, each of which is handled by a separate processing unit independent of the other.

commands. This way, a large amount of data is processed simultaneously and sequential processing of the data is avoided. In the MapReduce programming model likewise, the data are divided into independent chunks which are processed by the mapping tasks simultaneously in a completely parallel manner. This is similar to data parallelism. These chunks then act as an input to the reducing tasks, whose prime job is to summarize the mapped information.

3.3 Scalability through Feature Engineering

Feature engineering attempts to reduce the dimensionality of the raw data through customized feature computations or through feature transformations, e.g., handcrafted feature extraction, principal component analysis, or use of pretrained autoencoders for optimal data representation.

Depending on the type of the data, and knowing the typical queries for that modality of data, the best defining attributes or features are often extracted. As an example, for an image, detecting the edges or blobs (i.e., the regions that differ in properties, e.g., colour, compared to the surrounding regions) is often of great interest and helps identify objects in the image. Similarly, for an audio data, the energy and dominant frequency of the signal together translate to the perceived loudness, subject to the equal loudness contour. The frequency domain representation of the audio provides us with information about the array of dominant frequencies, and consequently the formants and the harmonic structure. As for the speech signal, knowing the relative location of the formants helps to estimate the vowel being spoken. For video sequences, optical flow is the feature of interest that helps the relative velocities of the objects in an image to be estimated. Table 3.1 lists the most common handcrafted features particular to the common modalities of the data.

Table 3.1 Common handcrafted features.

Audio	Image	Video	Text
Energy	Edges	Optical flow	Tokenization
Mel frequency cepstral coefficients	Blobs	Image and audio features per unit time step	Stemmed words
Formant locations	Ridges		Capitalization pattern
Zero crossing rate	Corners	Feature differences per unit time step	Stemmed words
Fundamental frequency	Interest points		Corrected spellings

3.3.1 Feature Reduction through Spatial Transformations

This is done by decorrelating variables through matrix factorization (e.g., non-negative matrix factorization (NMF)), analysis of variances (e.g., principal component analysis (PCA) and linear discriminant analysis (LDA)). For example, PCA transforms observations of possibly correlated variables into a set of linearly uncorrelated variables called the *principal components*. Data is first transformed into a new co-ordinate system such that the greatest variance by some projection comes to lie on the first co-ordinate, the second greatest on the second and so on. In many cases, reconstruction using only the top few principal components is an accurate enough description of the data. PCA is purely statistical in nature, and it takes into account all of the data samples without discriminating between the classes the samples belong to. This approach is commonly known as the *unsupervised* approach. Certain other techniques use the *supervised* approach. These make use of the “class label” information identifying most discriminating attributes, in other words, the most useful features for classification tasks. LDA is one of these techniques. This method also relies on a linear transformation of the features similar to PCA, but it attempts to compute the axes that maximize the separation between multiple classes, rather than maximizing the variance across all of the data samples. The difference between the two approaches can be visualized from Figure 3.4. However, PCA and LDA are primitive techniques, they are not as useful when it comes to feature reduction on massive amounts of data. Several methods for feature selection have now been implemented that are based on variance preservation or use SVM that are aimed at data-mining on large-scale data [14–16]. The open source toolkit *openBliSSART* was the first to bring non-negative matrix factorization to GPU [17].

3.3.2 Laplacian Matrix Representation

In the simplest terms, the motivation behind algorithms like spectral clustering approaches is to take into account the adjacencies of every data point with respect to the dataset as a whole, rather than merely computing the pairwise distances across the data. Figure 3.5 presents an example case where this can be particularly useful. The two data points A and B, belonging to the same cluster, are far away from each other. The pairwise distance of the point C from B is much smaller in comparison. Yet, intuitively speaking, it makes sense to assign the points A and B to the same cluster,

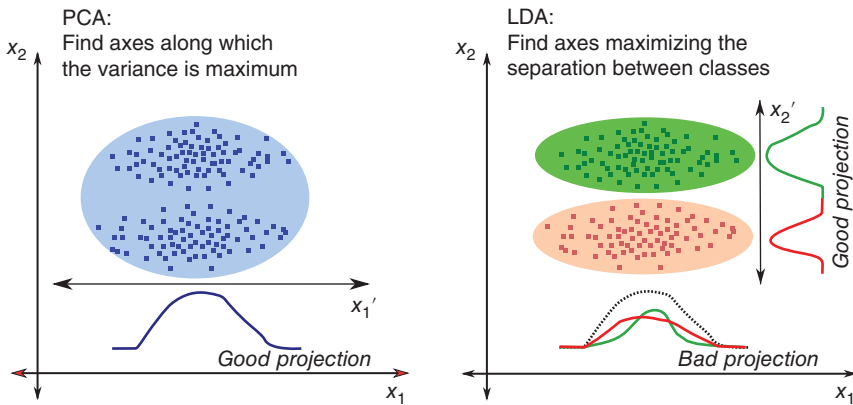


Figure 3.4 Difference between PCA and LDA feature reduction techniques, reducing the feature vector dimensionality from 2 to 1 in the example above. PCA aims to find the axes along which the variance of the data is maximum. The corresponding transformation thus generates feature dimensions corresponding to linearly uncorrelated variables. PCA, however, does not take into account the class labels, so the variance maximization alone therefore may result in a non-ideal projection (axis X_1 or X_1'). LDA aims to find axes maximizing separation between the classes (axis X_2 or X_2').

and to bin the point C into another, looking at how the two data points A and B are related or “connected” through the other data points in the feature space.

To achieve this, first a *Laplacian matrix* (L) is built using the pairwise distances between the individual data points v_i and v_j (called the *adjacency matrix* (A)) and the cluster they belong to (represented by the *degree matrix* (D)):

$$L = D - A \quad (3.1)$$

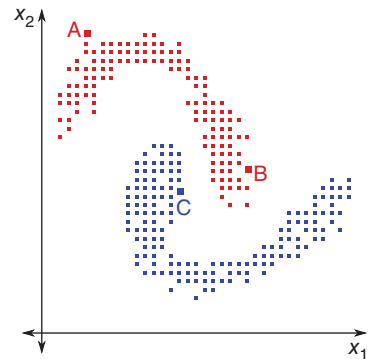
or in the symmetric normalized form:

$$L^{sym} = I - D^{-1/2} A D^{-1/2} \quad (3.2)$$

The elements of L and L^{sym} are given by following equations:

$$L_{ij} = \begin{cases} \deg(v_i) & \text{if } i = j \\ -1 & \text{if } i \neq j \text{ and } v_i \text{ is adjacent to } v_j \\ 0 & \text{otherwise} \end{cases} \quad (3.3)$$

Figure 3.5 Spectral clustering takes into account data adjacencies in addition to the pairwise distances between the data points. The points A and B are far apart, but belong to the same class. They relate to each other through other data points in their close vicinity (adjacency) and the adjacencies of those data points in iteration. While the points B and C are much closer in terms of euclidean distance, the data adjacency consideration makes it clear that they do not belong to the same class.



$$L_{i,j}^{sym} = \begin{cases} 1 & \text{if } i = j \\ \frac{-1}{\sqrt{\deg(v_i)\deg(v_j)}} & \text{if } i \neq j \text{ and } v_i \text{ is adjacent to } v_j \\ 0 & \text{otherwise} \end{cases} \quad (3.4)$$

The eigenvalues (λ_i) and eigenvectors (V_i) of the Laplacian matrix reduce the dimensionality of the dataset, effectively representing the entire dataset. The eigenvalues, thus obtained, also tell us the number of “graphs” or the number of connected data instances in the dataset, in addition to how this eigenvector representation linearly sums up representing the data effectively. Parallelization of this process has been proposed [18], splitting the data with n entries to p different machine nodes. Each node computes n/p eigenvectors and eigenvalues as necessary through parallelization of the Arnoldi factorization, using PARPACK [19] (the parallel version of the popular eigensolver ARPACK [20]).

3.3.3 Parallel latent Dirichlet allocation and bag of words

The latent Dirichlet allocation and the bag of words approaches are very similar, where every “document” (this can be also a non-textual sequence of audio, video, or other data) is viewed as a mixture of “topics”, each defined by a multinomial distribution over a W -word vocabulary. The documents (or the data content) with similar distribution in terms of topics (feature vectors) are thus considered to be similar. The paper that first proposed this approach in 2003 [21] discusses the evolution of this approach from the classical *tf-idf*-based feature reduction technique, and its close relationship with other intermediate approaches in its evolution such as *latent semantic indexing (LSI)* and the *probabilistic LSI model* (also called the *aspect model*). Interestingly enough, essentially the same model was independently proposed in the study of population genetics in 2000 [22, 23]. Bag of words feature representation has proved to be useful for machine learning tasks on multiple modalities of data, such as large-scale image mining [24], text data [25], audio [26], and videos [27]. A scalable implementation of this approach was proposed in 2011 [28] through both data and process parallelization, building on some of the previous work in this domain [29, 30]. Recently, our group released an open-source toolkit, called *openXBOW*, that can be easily interfaced with multiple data modalities to generate a summarized crossmodal bag of words representation [31] (Figure 3.6).

3.4 Deep Learning-Based Feature Learning

To make sense of the data, traditional machine learning approaches (including the ones listed above) rely heavily on the representation of the available data in the appropriate feature space. The feature extraction step therefore needs to address extraction of *all* and *only* the most relevant features given the machine learning task at hand. The redundant features simply add to the dimensionality of the feature vectors unnecessarily, which severely affects their discriminating power. For example, given a speech sample, the feature sets necessary for automatic speech recognition (ASR),

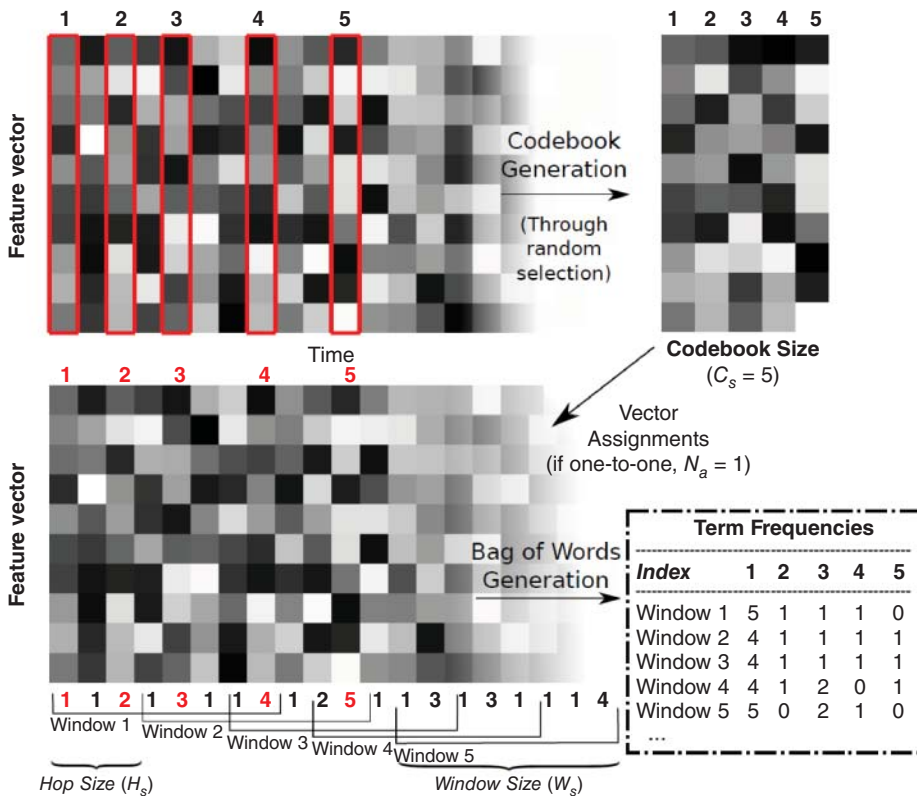


Figure 3.6 Bag of words approach. When the size of the moving window matches the size of the incoming input (or the document) itself, i.e., when the entire incoming input is represented in terms of the counts of the types of features (or the words) it exhibits, the approach closely resembles the latent Dirichlet allocation representation.

speaker/gender identification, and emotion recognition are not identical. While speaker identification is often solved by extraction of the universal background model (UBM) and i-vectors, the statistics associated with fundamental frequency (F_0) are popular for gender identification. The more complex paralinguistic tasks might entail extraction of thousands of features [32–35].

Interestingly enough, a new approach to feature extraction has evolved that attempts to *learn* what features are most useful when given a task. This is called the *deep learning approach*, and it uses a network of cells similar to neurons in the brain. In addition to big data, this deep learning has also been a buzzword in both the commercial world and the research community in recent times. The beauty of the deep learning approach is that it abstractifies the feature extraction step as part of the training phase itself. Depending on the task at hand, this artificial neural network-based model attempts to reorient the intermediate computations (also therefore the “features” in the abstract sense) to best map the given inputs to the single or array of desired outputs.

3.4.1 Adaptability that Conquers both Volume and Velocity

At the heart of any deep learning architecture is a biology-inspired artificial neural network model with interconnected groups of nodes, akin to neurons in the brain as discussed previously. In its simplest form, the nodes are organized in the form of layers (Figure 3.7), passing on the individual outputs from the input layer to the final output layer. Each node applies a predefined mathematical transformation to the weighted sum of its inputs to generate an output, or the activation value. This activation in turn acts as an input for the next set of nodes the given node is connected to. In its rudimentary form, therefore, each node is defined only by three types of parameters: (i) the nodes its inputs and outputs are connected to, (ii) the weights applied to these interconnections, and (iii) the activation function that converts the weighted inputs to the output value it generates. This most simplistic model is known as the *multilayer perceptron model (MLP)*. The weights that map the inputs to the expected output values are learnt through iterations, by attempting to minimize the difference between the expected outputs and the computed outputs. This cost is mostly minimized using different versions of the gradient descent algorithm. The term *deep* refers to the number of layers that the network features. The higher the number of hidden layers, the more the degrees of freedom (weight coefficients) with which the model can perform mapping on large-scale data. While fairly simplistic, it has been theorized that the model can approximate any continuous function on compact subsets of \mathbb{R}^n (*universal approximation theorem* [36]) using only a single hidden layer with a finite number of neurons. Depending on the training strategy, the hyperparameters involved with the training process are the cost function, optimization algorithm, learning rate, dropout factor, and activation functions used.

Such great adaptability in mapping any inputs to outputs also entails that, given a very small set of inputs and outputs with the cost function, there likely are multiple possibilities for the “optimal” mapping. Thus, achieving the most generalizable and robust mapping necessitates also the use of a large number of training samples in order to avoid

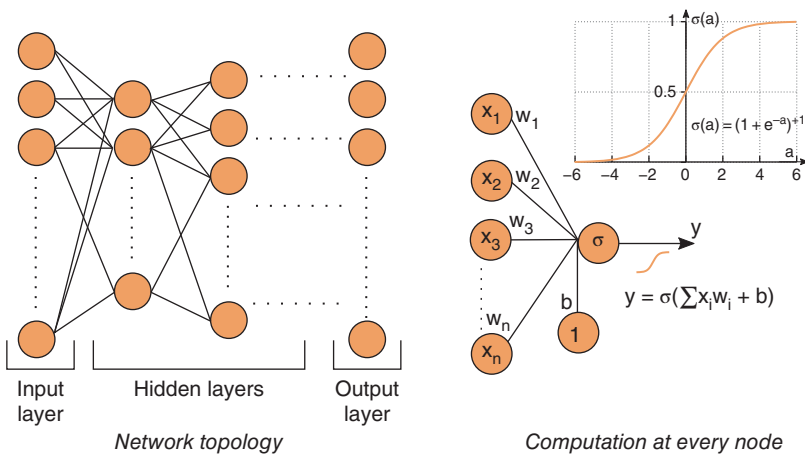


Figure 3.7 The multilayer perceptron model. For every node, the output y is computed by applying a functional transformation $\sigma(a)$ to the weighted sum of its inputs, where weights are denoted by w_i corresponding to input values x_i .

overfitting of the model to a very small set of instances. Also, because this training is done iteratively or incrementally, the training step inherently does not require the model to know all of the feature space in its entirety at once. This is opposite to SVM-like algorithms, which often rely heavily on the data clusters present in the feature space when generating the discriminating hyper-surfaces. Therefore, while large-scale data is often a problem due to the memory and computation bandwidth available, it is often looked on as the great advantage when it comes to the neural network-based models. These models can also afford to split the data into smaller chunks without compromising as much on the generalization of the mapping, and thus deal more easily with the *volume* issue.

As we will see later, the deep learning model can handle the variety of data modalities, and can take into account higher concepts such as time and the temporal correlations. Therefore, it is possible to feed in not only scalar values of the data to the input nodes, but also the streams of scalars (i.e., feature vectors), streams of feature vectors (i.e., feature matrices), and streams of feature matrices, and so on. This hierarchical mathematically generalizing structure is formally known as a tensor, where scalars, vectors, and matrices are tensors of rank 0, 1, 2 and so on. The uniqueness of the deep learning approach is that it is perhaps the only approach to date that can work with tensors of rank higher than 2 as its inputs.

Because these models can easily reorient and adapt to incoming inputs, they can be used in online systems that are continuously fed with large streams of data in real time. Big data velocities are way too stupendous for even the storage systems to catch up in many cases; it is impractical to expect the machine learning algorithms to finish data processing before the next samples show up. Because the incoming input gets evaluated using the pretrained weight parameters with simple mathematical functions alone (without looking at the subspace of the dataset, or the decision trees, like in other algorithms), the feature-set extraction and the prediction task can be performed at a much faster rate. Also, because the implicit feature-set extraction and prediction are both mathematical functions involving repeated use of subfunctions, such as sigmoid or tanh on the matrices or tensors as their argument, advanced matrix manipulation techniques and parallelization can in theory be applied for faster throughput. The developers of one of the most popular and influential toolkits today, called *Tensorflow*, have released a white paper recently which briefly mentions their plans on introducing just-in-time (JIT) compilations of the subgraphs [37].

The autoencoders present a special case of deep neural networks, where the output is identical to the input and therefore a different/compressed (or rarely also expanded) representation of the data is learnt intrinsically. Because the inputs always map to themselves, the intermediate layers (generally smaller in dimension compared to the input dimension in the case of a compression autoencoder) represent their compact or “noise-free” representation. To make the model (and consequently the reduced representation) more robust against the noise and spurious inputs, the network is often trained using original, clean input as its expected output, while feeding in the noisy version of the input to the network (Figure 3.8). This class of autoencoders is called denoizing autoencoders. Because they map inputs to themselves through mostly nonlinear transformations, they are often said to perform “implicit learning” of the data. Such compressed encoding of the input is often referred as the “bottleneck” representation. The autoencoders can be hierarchically stacked by making the encoded

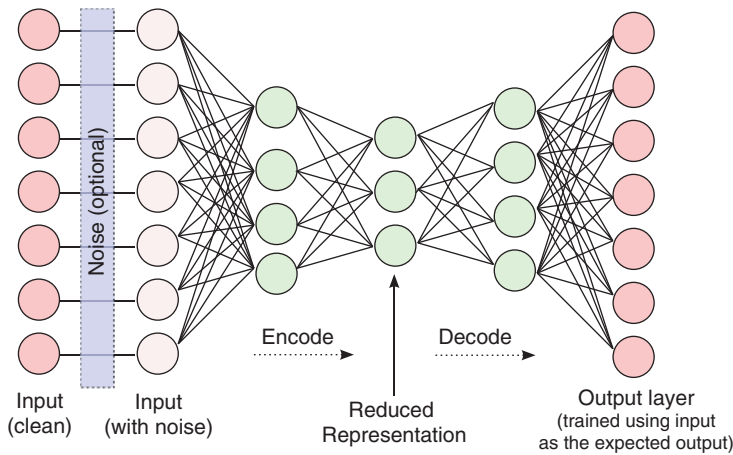


Figure 3.8 The autoencoder (compression encoder). The hidden layers represent the essence of the data provided, a meaningful compact representation of the inputs, also called the “bottleneck” representation.

representation run through a compressing autoencoder, implicitly learning the encoded representation. This approach has been useful in domain-adaptive sentiment recognition from speech [38], content-based image retrieval [39], estimating visual tracking for videos [40], paraphrase detection in texts [41], and a myriad of further tasks.

3.4.2 Convolutional Neural Networks

While the simple neural networks manage to map inputs to their outputs, such a network often features a fully connected graph (albeit not in both directions), which completely misses out on important dependencies or relationships between the node groups. To perform an image recognition task, for example, a simplistic MLP model might use a flattened single dimensional array generated from 2D pixel values, without any anticipatory consideration for the spatial relationships existing in the data. We know from experience, however, that the edges in an image are often defined by the differences and deviations between the intensities of the neighbouring pixels. Their spatial relationships define how slanted the edges are with respect to the horizontal or vertical axes. It makes sense to assign high magnitude weights (irrespective of the sign) to the neighbouring pixels rather than those very far away. A carefully weighted sum of the pixel values, with weights that represent a certain preselected edge orientation (called the *receptive field kernel*), tells us whether or not the pixel neighbourhood features a preselected image pattern, e.g., an edge, or a blob (Figure 3.9). High correlation between the pixel neighbourhood and the receptive field translates to a high activation value. Different receptive fields moving across the image through fixed steps (called *strides*) in both dimensions can effectively localize different edges and patterns in the image. Understanding of the spatial relationships between these patterns, and consequently localization of the high and low activation values, effectively translates to nothing but an object recognition task itself. Typically, the “pooling layers” are also introduced in between the layers. The pooling layers reduce the dimensionality of the input matrix size, while preserving the salient information obtained in the earlier computations

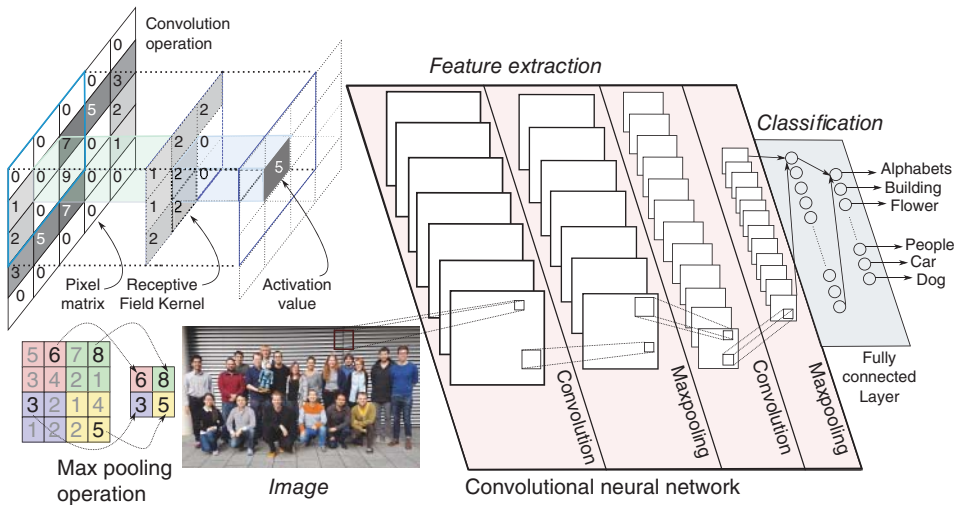


Figure 3.9 In a convolution operation, the receptive field kernel (size: 3×3) is multiplied element-wise with the subset of a pixel matrix of the same size, and these weighted values are summed together to compute the activation value. A max pooling operation reduces the dimensionality by computing the maximum value in the kernel of a given size (size: 2×2).

through a non-linear transformation, such as a summing or a maximizing function. The typical example of a pooling layer is the max pooling layer that applies a *max* operation for the stated kernel or window. Figure 3.9 features a max pooling layer of a size 2×2 kernel and a convolution kernel of size 3×3 for illustration purposes. The feature representation in the form of activations by the first few convolutional layers often translates to the likelihood of certain pixel patterns being present in the image. This, in some sense, is similar to the bag of words approach. The spatial correlation between these patterns (captured using the convolutional kernel) translates to detection of higher order patterns. Because these weights get trained, this allows for much more complex mappings and feature manipulations beyond the standard bag of words approach.

3.4.3 Recurrent Neural Networks

The simple multilayer perceptron model fails to take into account time-related dependencies and relationships existing in the data, which is of utmost importance when it comes to usual multimedia data that is of a sequential form, e.g., an audio clip, frames of a video, or even textual data. In these cases, the expected output is dependent not only on the current input, but also on the sequence of preceding and potentially following inputs. To model this kind of relationship, the outputs of the hidden layer are often connected back to its own input. In this way, each node in the layer computes a weighted sum of its inputs coming from a previous layer and the last output it generated. The inputs from the previous time step, therefore, also affect the activation computation in the current step. In a bidirectional architecture, the future observations are also used.

Such a network that uses simple perceptrons as its nodes, computing mere non-linear transformation of the weighted inputs, fails to model long-term dependencies. To provide the network with the desired capability, alternate topologies for the nodes

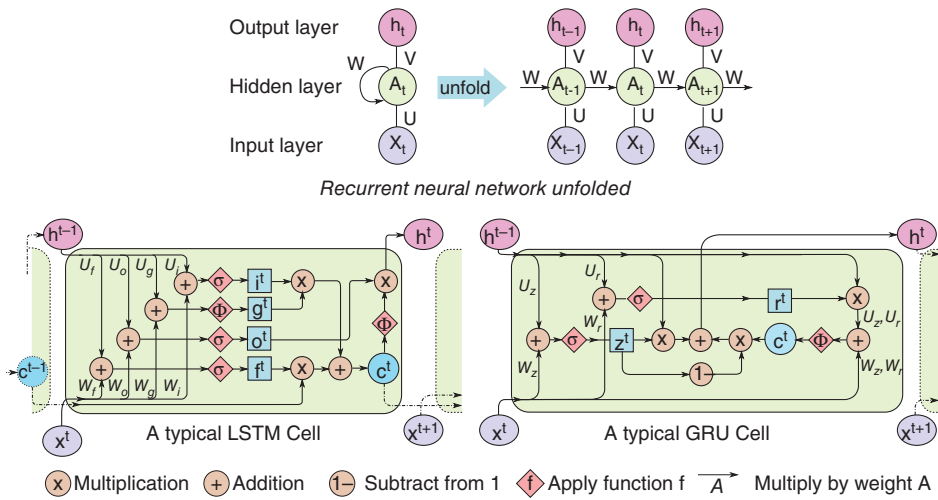


Figure 3.10 In a recurrent neural network, output from a node is fed back to itself to make the past inputs affect the outputs at the next time step. A simple perceptron, however, fails to capture the long-term dependencies. The modified cells with internal memory state (e.g., LSTM and GRU cells) are used to alleviate this problem.

were proposed which feature an internal memory state. The different node topologies only differ in the way this internal memory state is calculated, and the way this state is used to compute the outputs in the current and the next time steps. Figure 3.10 illustrates two of the most popular topologies in use today, namely *long short-term memory* (LSTM) cells and *gated recurrent unit* (GRU) cells. By mapping the temporal relationship of the samples in the sequential data with respect to one another, and also with the respect to the sequential (or the summarized static) output values through reiteration and re-usage of the mapping functions, we get the more compact and most essential representation of the input sequence in terms of far fewer parameters when using recurrent neural networks. The open source toolkit *CURRENNT* was the first one to have GPU implementation of the bidirectional LSTM (BLSTM) [42].

3.4.4 Modular Approach to Scalability

The deep learning approach is characteristically modular and therefore multiplicatively scalable. One can mix and match different numbers and types of layers (e.g., fully connected, convolutional, recurrent, max pooling) with different numbers and types of nodes in each layer (e.g., perceptron, LSTM cell, GRU cell) with different activation functions (e.g., sigmoid, tanh, linear) with a variety of interconnections (densely connected, sparsely connected neural network), also deciding how the output relates to the inputs (single or multiple class labels, single or multiple regression values, or simply de-noised inputs) when training the network. This presents a multitude of possibilities in terms of the network architecture itself, and what all it can possibly model. Depending on how complex the problem is, and/or how sparse the data are, and what higher level

concepts and context it should model, the hyper-parameters such as the type of network elements, governing equations, and quantity of layers and nodes can be determined and experimented with.

Such an approach has opened up a plethora of opportunities for research in compact data representation and understanding. It has spawned several new interesting architectures and findings by blending ideas from different basic topologies discussed earlier and coming up with innovative architectures. Because in sequential data the outputs and inputs are temporally correlated, some of the early publications on LSTMs made use of inputs from the future time steps to compute the output at the current time step. This gave rise to *BLSTMs*, which is popular today. These have been especially useful in helping computing systems better understand sequential media such as speech, for tasks such as speech overlap detection, language, and acoustic modeling [43, 44]. Some of the recent publications propose network architectures in which the convolutional neural networks model the sequential data through what are called *convolutional recurrent neural networks* [45–48]. Some have also attempted compact representation of the sequential data using *recurrent autoencoders* [49]. A variant of such autoencoders, called variational autoencoders, employ a special kind of cost function that forces the bottleneck representation to be as close as possible to a Gaussian distribution [50]. Recently, drawing inspiration from their earlier work on image generation, called PixelCNN [51], researchers from *DeepMind* came up with a generative model for speech synthesis called *WaveNet*, using what they called *causal convolutional layers* [52]. Through training using real-life examples, the network is able to generate samples which when put together resemble natural speech. Because one can also have generative models beyond the predictive ones, using deep learning networks, recently researchers mixed the two, generating an interesting unsupervised machine learning model called *generative adversarial networks (GANs)* [53]. In this approach, the two networks, one generative and the other predictive, compete against each another in a zero-sum game framework. The basic idea is that the generative network is trained to generate synthesized data instances using the training instances from a latent variable space, while the predictive network is simultaneously trained to differentiate between the synthesized instances from the true instances. At the other end of the spectrum, in terms of their simplicity, are *extreme learning machines (ELMs)* in which the inputs to hidden layer weights are randomly assigned and are never updated [54]. In ELMs, only the hidden layer to output mapping is learnt through iterations. ELM training thus becomes essentially a linear learning problem [55], significantly reducing the computational burden. While controversial [56–58], these simplistic models seem to work on complex tasks such as hand-written character recognition [59] or excitation localization for the snore sounds [60]. Another variant of neural networks, called the residual network, makes use of “shortcut connections” to skip over a few layers to feed the signal to the desired layer for summation. Thus, instead of directly training the network for the mappings desired (say, $[H(x)]$), the residual network instead is trained to learn the mapping between the input and the desired output that is offset by the input itself (i.e., $[H(x) - x]$). This modification in the topology has been shown to alleviate the problem of *degradation*, which is the saturation in accuracy with an increase in the number of layers or the depth of a network [61].

3.5 Benchmark Studies

The deep learning approach can effectively and intelligently process various kinds of data. As discussed, it can easily be customized to account for temporal dependencies and the contexts, which that are both intrinsic and vital when it comes to the sequential data, e.g., raw audio clippings, tweets, and news articles. One can also process spatial information, e.g., photographs, plots, and maps. Even data where the spatial and sequential contexts are highly intertwined, e.g., videos, console games, and live traffic maps, can be dealt with effectively using deep learning approaches through little adaptations in the network and the cells in use. The data can therefore be used in almost its original form (called *end-to-end learning*), without having to introduce any customized feature extraction step. The classical approach requires every data sample to be represented in feature vector form, and overall data as a 2D matrix (i.e., a rank 2 tensor). Because the neural network based models can handle higher level concepts (such as the time) effectively, the data is often represented as a tensor of rank even greater than 2. By extension therefore, the data learning process is inherently scalable, not only in terms of the number of features in use or the number of samples, but also the modalities and rank of the tensor it can handle at any given time. We discuss previous benchmarking studies on deep learning in this section.

Due to its huge potential and the consequent drive from both the industry and the research community, there is now huge growth in the number of deep learning toolkits available. Some of these were once the in-house frameworks for industry giants like Google. The most popular frameworks are Caffe, 2018, CURRENNT [42], Microsoft Cognitive Toolkit [62], Tensorflow [37], Theano [63], and Torch [64]. Some of these, like Keras [65], come with an API capable of running on top of other frameworks, e.g., Theano and Tensorflow.

Benchmark comparisons between these deep learning frameworks have been conducted in the past. The computational performance of the single framework depends highly on the available hardware architecture. One of the most critical parameters is the type of processors in use, i.e., whether *central processing units (CPUs)* or the *graphics processing units (GPUs)*, or, in the near future, *neuromorphic processing units (NPU)*s. The latter provide a much larger degree of parallelization and are therefore suitable for training deep neural networks. As the actual performance also depends on the employed version of the corresponding software tools and new frameworks are published on a regular basis, the reader is referred to online resources to catch up on the latest benchmarks.

Whereas these deep learning frameworks are well-studied in terms of their performance in machine learning tasks of all kinds, this is not the case for the feature-extraction step. In the following sections, we provide experiments and results for feature extraction via deep learning. For our benchmarking studies, we run feature extraction on the audio data using pretrained convolutional neural networks.

3.5.1 Dataset

We use the TUT Acoustic Scenes 2017 dataset, currently in use for the DCASE2017 Acoustic scene classification challenge, that was recorded in Finland by Tampere University of Technology. The dataset consists of 4680 samples recorded in 15 acoustic

scenes such as the forest, library, metro station, and restaurant indoors. At each recording location, a 3–5 minute high-quality audio was captured which was then split into 10-second segments. Each audio clip was recorded using 44.1 kHz sampling rate and 24 bit resolution. While this is by no means big data, we upsampled the data through repetition to a high data volume to showcase the principles.

3.5.2 Spectrogram Creation

When it comes to working with audio data, the dynamics in the frequency domain are often a good indicator. Therefore, features based on the frequency domain representation of the audio signal are almost always extracted for any machine learning tasks, e.g., Mel-frequency cepstral coefficients (MFCC), formant locations, or fundamental frequency (F_0). A varying spectrum of signal over time is often represented using a spectrogram, computed using short-time Fourier transform (STFT) or the wavelet transform. The horizontal axis represents time, the vertical axis represents frequency, and the magnitude or intensity of the continuous-time complex exponential of a particular frequency at a particular time is represented by the color or intensity of the point at that location. As long as one chooses the ‘right’ frame and hop size, taking into account the time frequency uncertainty principle for a given task, the spectrogram representation captures all of the quintessential information necessary. Instead of the raw audio therefore, we used the spectrogram representation as our input in this example. Summarizing, we translated the audio inputs into a format that can be best processed by the pretrained CNNs we aim to use.

Based on our previous experiments [66], we used the Hanning window of size 256 samples and traversed the audio data with a hop of 128 samples. We computed the power spectral density in decibels using the Python package *numpy* [67] and spectrogram representation using the *matplotlib* [68] package, in the perceptually uniform sequential colour mapping called *Viridis*. *Viridis* uses blue to represent the low-range values, and green and yellow for the mid-range and high-range values, respectively. As in our previous experiments, we got rid of the axes and the margins presenting redundant information that was common across all images by cropping programmatically. The original spectrogram images were then 387×387 pixels, which we rescaled to 227×227 pixels and 224×224 pixels, to comply with the AlexNet and VGG19 requirements, respectively. Figure 3.11 contains a few of the spectrograms we generated through this process from audios from four different classes. As can be seen, the spectrograms differ a lot visually, so much so that some clear distinctions can be made even with the human eye. The CNNs pretrained for image classification tasks in particular are, therefore, expected to perform well on this data, extracting the summarized feature representation with good discriminative scope.

3.5.3 CNN-Based Feature Extraction

Instead of explicitly computing the higher level audio features, e.g., F_0 and MFCC, through a dedicated feature extractor, we fed the spectrograms to pretrained CNNs that have been proven to work in image classification tasks. More specifically, we extracted the features using the pretrained AlexNet and VGG19 CNN (see chapter 1, Refs. 11, 23). The AlexNet was trained using ImageNet data, which featured more than

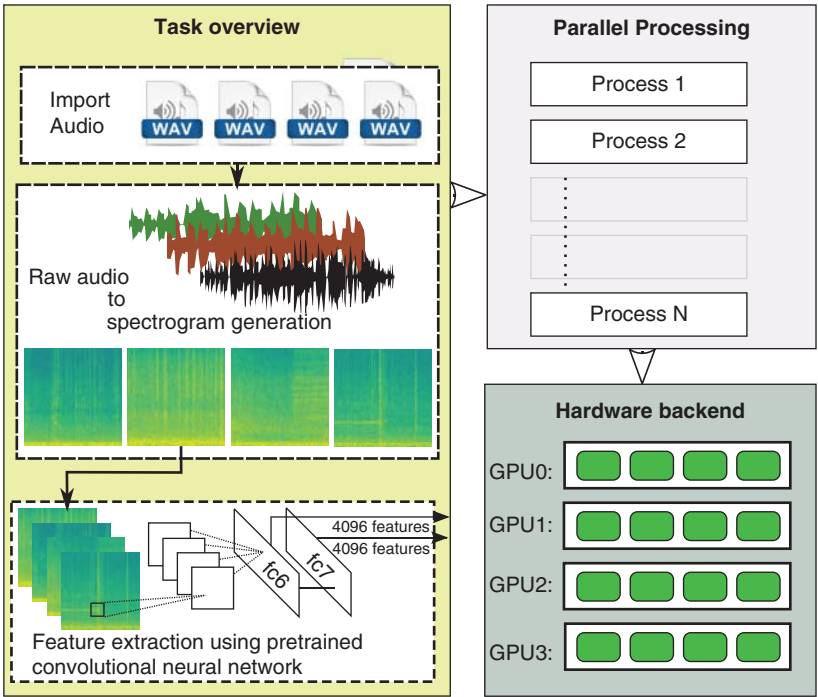


Figure 3.11 Overview of the system with CNN-based feature extraction through parallelization among GPU cores. In the example, spectrograms are generated from the audio data and plotted using the Python matplotlib package. The preprocessed plots are used as an input to the pretrained CNNs. The activations of fully connected layers are then extracted as large deep spectrum feature vectors, using the Caffe framework. The overall task is split into several processes and is handled by four GPU cores. Another thread writes the feature vectors into an output file.

15 million annotated everyday images from over 22 thousand categories, and we tested efficacy of this network in classifying spectrograms likewise [66, 69].

Both AlexNet and VGGNet provide us with 4096 features per spectrogram. This is huge data compression in itself, considering that every audio clip consists of close to 4 million data points ($44\,100\text{ Hz} \times 10\text{ seconds}$), and the sequential structure of the data points is also very critical. While such compression necessitates spectrogram generation first, this intermediate step is not a severe bottleneck.

3.5.4 Structure of the CNNs

Both AlexNet and VGG19 use a combination of convolutional layers, max pooling, fully connected layers and rectified linear units as the activation functions (see chapter 1, ref. [26]). AlexNet consists of five convolutional layers, followed by three fully connected layers. Softmax is applied to the last layer to perform 1000-way classification. VGG19 consists of 19 layers that are grouped in five stacks of convolutional layers with max pooling. Another key difference between the two networks is that VGG19 uses only 3×3 kernels across all its convolutional layers, while AlexNet uses varying kernel sizes. Table 3.2 summarizes the two network architectures for comparison.

Table 3.2 Overview of the architectures of the two CNNs used for the extraction of spectral features, AlexNet, and VGG19. *ch* stands for channels.

AlexNet	VGG19
Input = RGB image	
size: 227×227 pixels	size: 224×224 pixels
1 \times Convolution size: 11; ch: 96; stride: 4	2 \times Convolution size: 3; ch: 64; stride: 1
Max pooling	
1 \times Convolution size: 5; ch: 256	2 \times Convolution size: 3; ch: 128
Max pooling	
1 \times Convolution size: 3; ch: 384	4 \times Convolution size: 3; ch: 256
	Max pooling
1 \times Convolution size: 3; ch: 384	4 \times Convolution size: 3; ch: 512
	Max pooling
1 \times Convolution size: 3; ch: 256	4 \times Convolution size: 3; ch: 512
Max pooling	
Fully connected FC6 layer, 4096 neurons	
Fully connected FC7 layer, 4096 neurons	
Fully connected 1000 neurons	
Output = probabilities for 1000 object classes through softmax	

Once the spectrogram plots are forwarded through the pretrained networks, the activations from the neurons on the first or second fully connected layers (called *fc6* and *fc7*) are extracted as the feature vectors. The resulting feature vector thus presents 4096 attributes, one for every neuron in the CNN's fully connected layer. These can then be passed on to either the traditional or deep learning techniques to perform automatic scene event classification, similar to previous studies on audio classification through spectrogram image classification [66, 69].

3.5.5 Process Parallelization

We use the Caffe framework to build the CNN models and for process parallelization (see chapter 1, ref. [43]). We tested the run times without and with process parallelization (six processes). The GPU in use was a GeForce GTX Titan X (GPU clock rate: 1.06 GHz). The program run includes importing the audio data, creation and initialization of the pretrained convolutional neural network, generation of the cropped and resized spectrograms, forward pass computations through the pretrained network, and writing of the 4096 output features per audio input to an output csv file. The results are tabulated in Table 3.3. To measure the run time for experiments involving more samples than what we actually have, we upsampled through repetition

the audio instances. Because we were interested in benchmarking for feature extraction process, and how it scales with process parallelization, we did not run an evaluation of the fully connected classifier section of the network outputting class probabilities (and thus, we do not present accuracy results), avoiding additional process overhead.

3.5.6 Results

VGG19 has a higher number of layers compared to AlexNet (i.e., 19 compared to 8). Just as expected, therefore, it takes longer to finish VGG19-based feature extraction, as can be observed from Table 3.3 and Figure 3.12. Because there is a higher number of processes, speed up due to process parallelization is more visible for VGG19 than for AlexNet. On average, the feature extraction process is twice as fast for AlexNet with six processes run in parallel, while it is 2.6 times as fast for VGG19. Without parallelization, the degradation in data processing rate (average number of samples processed per second) is a lot more severe, dropping from almost 9 samples/second to about 5 samples/second. With parallelization, however, data processing rates are not as badly affected with a higher number of samples. With a very high number of samples, this gain also translates to huge savings in the total computation time. The run times stated here are for the parallelization of the algorithm where raw audio file to feature vector conversion is done sample by sample in iteration. Feature extraction could be further expedited by implementing data parallelization, by splitting the data into different chunks, with each program run processing a separate batch of data in parallel.

We used our own toolkit called *auDeep* for this study. It can be downloaded from the repository located at <https://github.com/auDeep/auDeep/>.

Table 3.3 Convolutional neural network speed up through process parallelization.

CNN	Number of samples	Run time (seconds)		Average samples/second		Speed up
		Single process	Six Processes	Single process	Six Processes	
AlexNet	500	55	27	9.09	18.52	2.04
	1 000	102	70	9.8	14.29	1.46
	5 000	520	340	9.62	14.71	1.53
	10 000	1 684	692	5.94	14.45	2.43
	25 000	4 424	1 782	5.65	14.03	2.48
	50 000	9 352	3 518	5.35	14.21	2.66
VGG19	500	91	37	5.49	13.51	2.46
	1 000	180	62	5.56	16.13	2.90
	5 000	921	343	5.43	14.58	2.69
	10 000	1 919	725	5.21	13.79	2.65
	25 000	4 900	2 218	5.10	11.27	2.21
	50 000	10 203	3 646	4.90	13.71	2.80

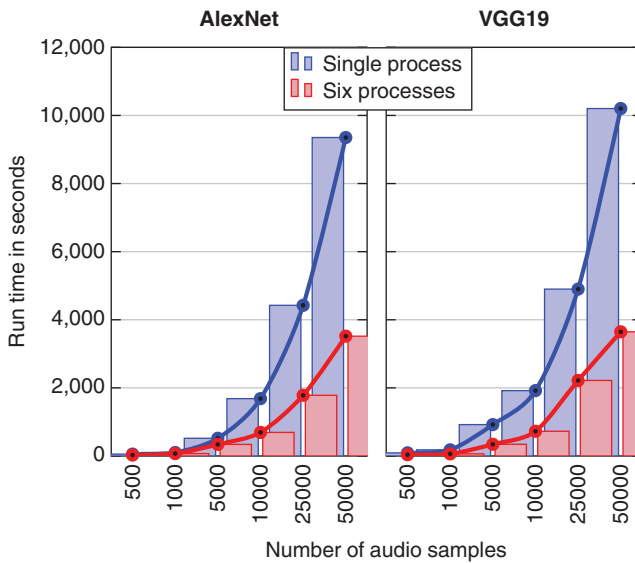


Figure 3.12 Convolutional neural network speed up through process parallelization (equal to that of Table 3.3).

3.6 Closing Remarks

Mining on big data multimedia content is a challenge in itself, mainly because of the three defining attributes of big data (volume, velocity and variety) and the multimodal nature of the multimedia. This makes it mandatory for the analytical tools to attain scalability in all three respects right from the feature extraction step. With limitations on how much hardware and software technology can scale through miniaturization and parallelization-like strategies, it comes down to inherent qualities of the competing data-mining methods as to which ones will survive the test of time.

The machine learning approaches rely heavily on the representation of the data in the feature space, with *only* the relevant discriminative features retrained and *all* of the redundant or irrelevant features removed. The redundant and irrelevant features are likely to meddle with the predictive ability of the model, adding to processing overheads. Unlike traditional methods, the deep learning approach is able to learn the most relevant features, or even the feature transforms implicitly, which makes it a promising candidate for scalability in terms of both *volume* and *variety*. Many traditional approaches do not take into account the sequential structure of the data (intrinsic to some of the multimedia data, e.g., audio, live maps, videos) or the spatial relationships within the data structure (e.g., in images, plots). The deep learning approach can model temporal, spatial, and intertwined correlations as useful features. In theory, it can also take in the data in its pure raw form, without an explicit feature extraction step. These factors put the approach at much greater advantage in terms of what concepts and relationships it can model, adding much more to its potential scalability in terms of variety. The inherent modularity of the approach only adds further to the scope for innovations through a mixing and matching approach, and the versatility of the models.

In terms of *velocity*, the gain due to data and process parallelizations is significant, especially when the machine learning approach uses the same set of sub-functions iteratively at different hierarchical levels. For sub-functions with more involved computations, one can speed up the processing through hash-tables. Unfortunately, most traditional approaches do not feature such repeated use of functions, but are able to attain speed up in implementation through some intelligent approximations and assumptions, in terms of the matrix operations or the kernels in use, or through use of hash-tables for the most frequent computations. The deep learning approach, on the other hand, can make use of all these approaches, while also featuring repeated use of predefined element-wise matrix multiplications, summation operations, and activation functions like sigmoid, tanh, and the convolution kernels in iteration.

In the context of multimedia big data, we present the results by testing scalability of the pretrained CNN-based feature extraction, using our *auDeep* toolkit. We find that the results for the scalability in terms of *velocity* are promising, with a speed up factor that is almost always more than twice, thanks to process parallelization. The versatility of the deep learning framework is evidenced by the fact that we used CNNs pretrained on image classification tasks to obtain the deep spectrum features from the audio data, which have also proven to be useful for audio classification task in our earlier studies. We intend to achieve further speed up through data parallelization and just-in-time compilation of the reused functions in our future experiments, reducing the computation times further by great margins.

Likewise, the current investment trends of the industry giants, the growth in big data, data economy, and the growing competition within the deep learning frameworks unequivocally imply deep learning is the future for big data analytics.

3.7 Acknowledgements



The research leading to these results received funding from the EU's Horizon 2020 Programme through the Innovative Action No. 645094 (SEWA) and from the German Federal Ministry of Education, Science, Research and Technology (BMBF) under the grant agreement no. 16SV7213 (EmotAsS). We further thank the NVIDIA Corporation for their support of this research by Tesla K40-type GPU donation.

References

- 1 Mayer-Schönberger, V. and Cukier, K. (2013) *Big data: A revolution that will transform how we live, work, and think*, Houghton Mifflin Harcourt.
- 2 Madden, S. (2012) From databases to big data. *IEEE Internet Computing*, 16 (3), 4–6.
- 3 Ioannidou, A., Chatzilari, E., Nikolopoulos, S., and Kompatsiaris, I. (2017) Deep learning advances in computer vision with D data: A survey. *ACM Computing Surveys*, 50 (2), 20.
- 4 Graf, H.P., Cosatto, E., Bottou, L., Durdanovic, I., and Vapnik, V. (2004) Parallel support vector machines: The cascade SVM, *Proceedings of the 17th International*

- Conference on Neural Information Processing Systems (NIPS '04)*, Vancouver, British Columbia, Canada, pp. 521–528, MIT Press.
- 5 Sun, Z. and Fox, G. (2012) Study on parallel SVM based on MapReduce, in *Proceedings of the International Conference Parallel and Distribution Processing Techniques and Applications* The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), pp. 495–502.
 - 6 Caruana, G., Li, M., and Qi, M. (2011) A MapReduce based parallel SVM for large scale spam filtering, in *2011 Eighth International Conference on Fuzzy Systems and Knowledge Discovery (FSKD)*, Volume 4, pp. 2659–2662, IEEE.
 - 7 Zhang, J.P., Li, Z.W., and Yang, J. (2005) A parallel SVM training algorithm on large-scale classification problems, in *International Conference on Machine Learning and Cybernetics*, pp. 1637–1641.
 - 8 Chang, E., Zhu, K., Wang, H., Bai, H., Li, J., Qiu, Z., and Cui, H. (2008) Parallelizing support vector machines on distributed computers, in *Neural Information Processing Systems*, pp. 257–264.
 - 9 Chang, C.C. and Lin, C.J. (2011) LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2 (3), 27:1–27:27.
 - 10 Mitchell, L., Sloan, T.M., Mewissen, M., Ghazal, P., Forster, T., Piotrowski, M., and Trew, A.S. (2011) A parallel random forest classifier for R, in *Proceedings of the 2nd International Workshop on Emerging Computational Methods for the Life Sciences*, ACM, pp. 1–6.
 - 11 Wright, M.N. and Ziegler, A. (2015) ranger: A fast implementation of random forests for high dimensional data in C++ and R. *arXiv preprint:1508.04409*.
 - 12 Pang, S., Ozawa, S., and Kasabov, N. (2005) Incremental linear discriminant analysis for classification of data streams. *IEEE Transactions on Systems, Man, Cybernetics, and Systems, Part B (Cybernetics)*, 35 (5), 905–914.
 - 13 Hubert, M. and Van Driessen, K. (2004) Fast and robust discriminant analysis. *Computational Statistics and Data Analysis*, 45 (2), 301–320.
 - 14 Pudil, P., Novovičová, J., and Kittler, J. (1994) Floating search methods in feature selection. *Pattern Recognition Letters*, 15 (11), 1119–1125.
 - 15 Siedlecki, W. and Sklansky, J. (1989) A note on genetic algorithms for large-scale feature selection. *Pattern Recognition Letters*, 10 (5), 335–347.
 - 16 Bradley, P.S. and Mangasarian, O.L. (1998) Feature selection via concave minimization and support vector machines, in *International Conference on Machine Learning*, pp. 82–90.
 - 17 Weninger, F. and Schuller, B. (2012) Optimization and parallelization of monaural source separation algorithms in the openBliSSART toolkit. *Journal of Signal Processing Systems*, 69 (3), 267–277.
 - 18 Chen, W.Y., Song, Y., Bai, H., Lin, C.J., and Chang, E.Y. (2011) Parallel spectral clustering in distributed systems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33 (3), 568–586.
 - 19 Maschho, K.J. and Sorensen, D. (1996) A portable implementation of ARPACK for distributed memory parallel architectures, in *Proceedings of the Copper Mountain Conference on Iterative Methods*, April, pp. 9–13.
 - 20 Lehoucq, R.B., Sorensen, D.C., and Yang, C. (1998) *ARPACK users' guide: solution of large-scale eigenvalue problems with implicitly restarted Arnoldi methods*, SIAM.

- 21 Blei, D.M., Ng, A.Y., and Jordan, M.I. (2003) Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3, 993–1022.
- 22 Pritchard, J.K., Stephens, M., and Donnelly, P. (2000) Inference of population structure using multilocus genotype data. *Genetics*, 155 (2), 945–959.
- 23 Blei, D.M. (2012) Probabilistic topic models. *Communications of the ACM*, 55 (4), 77–84.
- 24 Philbin, J., Chum, O., Isard, M., Sivic, J., and Zisserman, A. (2007) Object retrieval with large vocabularies and fast spatial matching, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–8.
- 25 Dhillon, I.S., Fan, J., and Guan, Y. (2001) Efficient clustering of very large document collections. *Data Mining for Scientific and Engineering Applications*, 2, 357–381.
- 26 Schmitt, M., Janott, C., Pandit, V., Qian, K., Heiser, C., Hemmert, W., and Schuller, B. (2016) A Bag-of-audio-words approach for snore sounds' excitation localisation, in *Proceedings of the 14th ITG Conference on Speech Communication*, Paderborn, Germany, pp. 230–234.
- 27 Deng, J., Cummins, N., Han, J., Xu, X., Ren, Z., Pandit, V., Zhang, Z., and Schuller, B. (2016) The University of Passau Open Emotion Recognition System for the Multimodal Emotion Challenge, in *Proceedings of the 7th Chinese Conference on Pattern Recognition*, CCPR, Springer, Chengdu, P.R. China, pp. 652–666.
- 28 Liu, Z., Zhang, Y., Chang, E.Y., and Sun, M. (2011) Plda+: Parallel latent Dirichlet allocation with data placement and pipeline processing. *ACM Transactions on Intelligent Systems and Technology*, 2 (3), 26.
- 29 Wang, Y., Bai, H., Stanton, M., Chen, W.Y., and Chang, E.Y. (2009) PLDA: Parallel latent Dirichlet allocation for large-scale applications, in *International Conference on Algorithmic Applications in Management*, Springer, pp. 301–314.
- 30 Newman, D., Asuncion, A.U., Smyth, P., and Welling, M. (2007) Distributed inference for latent Dirichlet allocation, *Proceedings of the 20th International Conference on Neural Information Processing Systems (NIPS '07)*, Vancouver, British Columbia, Canada, pp. 1081–1088, Curran Associates Inc.,
- 31 Schmitt, M. and Schuller, B. (2017) openXBOW – Introducing the Passau Open-Source Crossmodal Bag-of-Words Toolkit. *Journal of Machine Learning Research*, 18, 3370–3374.
- 32 Schuller, B., Steidl, S., Batliner, A., Vinciarelli, A., Scherer, K., Ringeval, F., Chetouani, M., Weninger, F., Eyben, F., Marchi, E., Mortillaro, M., Salamin, H., Polychroniou, A., Valente, F., and Kim, S. (2013) The INTERSPEECH 2013 Computational Paralinguistics Challenge: Social Signals, Conflict, Emotion, Autism, in *Proceedings of the 14th Annual Conference of the International Speech Communication Association (INTERSPEECH)*, ISCA, Lyon, France, pp. 148–152.
- 33 Schuller, B., Steidl, S., Batliner, A., Hantke, S., Hönl, F., Orozco-Arroyave, J.R., Nöth, E., Zhang, Y., and Weninger, F. (2015) The INTERSPEECH 2015 Computational Paralinguistics Challenge: Degree of Nativeness, Parkinson's & Eating Condition, in *Proceedings of the 16th Annual Conference of the International Speech Communication Association (INTERSPEECH)*, ISCA, Dresden, Germany, pp. 478–482.
- 34 Schuller, B., Steidl, S., Batliner, A., Hirschberg, J., Burgoon, J.K., Baird, A., Elkins, A., Zhang, Y., Coutinho, E., and Evanini, K. (2016) The INTERSPEECH 2016 Computational Paralinguistics Challenge: Deception, Sincerity & Native Language, in

- Proceedings of the 17th Annual Conference of the International Speech Communication Association (INTERSPEECH)*, ISCA, San Francisco, CA, pp. 2001–2005.
- 35 Schuller, B., Steidl, S., Batliner, A., Bergelson, E., Krajewski, J., Janott, C., Amatuni, A., Casillas, M., Seidl, A., Soderstrom, M., Warlaumont, A., Hidalgo, G., Schnieder, S., Heiser, C., Hohenhorst, W., Herzog, M., Schmitt, M., Qian, K., Zhang, Y., Trigeorgis, G., Tzirakis, P., and Zafeiriou, S. (2017) The INTERSPEECH 2017 Computational Paralinguistics Challenge: Addressee, Cold & Snoring, in *Proceedings of the 18th Annual Conference of the International Speech Communication Association (INTERSPEECH)*, ISCA, Stockholm, Sweden.
 - 36 Zainuddin, Z. and Pauline, O. (2007) Function approximation using artificial neural networks, in *Proceedings of the 12th WSEAS International Conference on Applied Mathematics*, World Scientific and Engineering Academy and Society (WSEAS), MATH'07, pp. 140–145.
 - 37 Abadi, M. et al. (2016) Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*.
 - 38 Deng, J., Zhang, Z., Eyben, F., and Schuller, B. (2014) Autoencoder-based Unsupervised Domain Adaptation for Speech Emotion Recognition. *IEEE Signal Processing Letters*, 21 (9), 1068–1072.
 - 39 Krizhevsky, A. and Hinton, G.E. (2011) Using very deep autoencoders for content-based image retrieval., in *ESANN 2011, 19th European Symposium on Artificial Neural Networks*, Bruges, Belgium, April 27–29.
 - 40 Wang, N. and Yeung, D.Y. (2013) Learning a deep compact image representation for visual tracking, in *Advances in Neural Information Processing Systems*, pp. 809–817.
 - 41 Socher, R., Huang, E.H., Pennington, J., Ng, A.Y., and Manning, C.D. (2011) Dynamic pooling and unfolding recursive autoencoders for paraphrase detection, in *Neural Information Processing Systems*, vol. 24, pp. 801–809.
 - 42 Weninger, F., Bergmann, J., and Schuller, B. (2015) Introducing CURRENNT: the Munich Open-Source CUDA RecurREnt Neural Network Toolkit. *Journal of Machine Learning Research*, 16, 547–551.
 - 43 Mousa, A.E.D. and Schuller, B. (2016) Deep bidirectional long short-term memory recurrent neural networks for grapheme-to-phoneme conversion utilizing complex many-to-many alignments, in *Proceedings of the 17th Annual Conference of the International Speech Communication Association (INTERSPEECH)*, ISCA, San Francisco, CA, pp. 2836–2840.
 - 44 Hagerer, G., Pandit, V., Eyben, F., and Schuller, B. (2017) Enhancing LSTM RNN-based speech overlap detection by artificially mixed data, in *Proceedings of the AES 56th International Conference on Semantic Audio*, AES, Audio Engineering Society, Erlangen, Germany, pp. 1–8.
 - 45 Liang, M. and Hu, X. (2015) Recurrent convolutional neural network for object recognition, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3367–3375.
 - 46 Pinheiro, P.H. and Collobert, R. (2014) Recurrent convolutional neural networks for scene labeling, in *International Conference on Machine Learning*, pp. 82–90.
 - 47 Donahue, J., Anne Hendricks, L., Guadarrama, S., Rohrbach, M., Venugopalan, S., Saenko, K., and Darrell, T. (2015) Long-term recurrent convolutional networks for visual recognition and description, in *Proceedings of the IEEE Conference Computer Vision and Pattern Recognition (CVPR)*, pp. 2625–2634.

- 48 Keren, G. and Schuller, B. (2016) Convolutional RNN: an enhanced model for extracting features from sequential data, in *Proceedings of the International Joint Conference on Neural Networks (IJCNN) as part of the IEEE World Congress on Computational Intelligence (IEEE WCCI)*, Vancouver, Canada, pp. 3412–3419.
- 49 Maas, A.L., Le, Q.V., O’Neil, T.M., Vinyals, O., Nguyen, P., and Ng, A.Y. (2012) Recurrent neural networks for noise reduction in robust ASR, in *Proceedings of the 13th Annual Conference of the International Speech Communication Association (INTERSPEECH)*, ISCA, Portland, OR, pp. 22–25.
- 50 Chung, J., Kastner, K., Dinh, L., Goel, K., Courville, A.C., and Bengio, Y. (2015) A recurrent latent variable model for sequential data, in *Proceedings of the 28th International Conference on Neural Information Processing Systems (NIPS ’15)*, Volume 2, Montreal, Canada, pp. 2980–2988, MIT Press, Cambridge, MA.
- 51 van den Oord, A., Kalchbrenner, N., Espeholt, L., Vinyals, O., Graves, A. et al. (2016) Conditional image generation with pixelcnn decoders, in *Proceedings of the 30th International Conference on Neural Information Processing Systems (NIPS’16)*, pp. 4790–4798, Curran Associates Inc.
- 52 van den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A., and Kavukcuoglu, K. (2016) Wavenet: A generative model for raw audio. *CoRR abs/1609.03499*.
- 53 Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014) Generative adversarial nets, *Proceedings of the 27th International Conference on Neural Information Processing Systems (NIPS ’14)*, Volume 2, Montreal, Canada, pp. 2672–2680, MIT Press, Cambridge, MA.
- 54 Huang, G.B., Zhu, Q.Y., and Siew, C.K. (2006) Extreme learning machine: theory and applications. *Neurocomputing*, 70 (1), 489–501.
- 55 Lin, S., Liu, X., Fang, J., and Xu, Z. (2015) Is Extreme Learning Machine Feasible? A Theoretical Assessment (Part II). *IEEE Transactions on Neural Network Learning Systems*, 26 (1), 21–34.
- 56 Liu, X., Lin, S., Fang, J., and Xu, Z. (2015) Is an extreme learning machine feasible? A theoretical assessment (Part I). *IEEE Transactions on Neural Network Learning Systems*, 26 (1), 7–20.
- 57 Wang, L.P. and Wan, C.R. (2008) Comments on “The Extreme Learning Machine”. *IEEE Transactions on Neural Networks*, 19 (8), 1494–1495.
- 58 Huang, G.B. (2008) Reply to ‘Comments on “The Extreme Learning Machine”’. *IEEE Transactions on Neural Networks*, 19 (8), 1495–1496.
- 59 Chacko, B.P., Krishnan, V.V., Raju, G., and Anto, P.B. (2012) Handwritten character recognition using wavelet energy and extreme learning machine. *International Journal of Machine Learning and Cybernetics*, 3 (2), 149–161.
- 60 Qian, K., Janott, C., Pandit, V., Zhang, Z., Heiser, C., Hohenhorst, W., Herzog, M., Hemmert, W., and Schuller, B. (2016) Classification of the excitation location of snore sounds in the upper airway by acoustic multi-feature analysis. *IEEE Transactions on Biomedical Engineering* 64 (8), 1731–1741.
- 61 He, K., Zhang, X., Ren, S., and Sun, J. (2016) Deep residual learning for image recognition, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778.

- 62 Agarwal, A., Akchurin, E., Basoglu, C., Chen, G., Cyphers, S., Droppo, J., Eversole, A., Guenter, B., Hillebrand, M., Hoens, T.R., Huang, X., Huang, Z., Ivanov, V., Kamenev, A., Kranen, P., Kuchaiev, O., Manousek, W., May, A., Mitra, B., Nano, O., Navarro, G., Orlov, A., Parthasarathi, H., Peng, B., Radmilac, M., Reznichenko, A., Seide, F., Seltzer, M.L., Slaney, M., Stolcke, A., Wang, H., Wang, Y., Yao, K., Yu, D., Zhang, Y., and Zweig, G. (2014) An Introduction to Computational Networks and the Computational Network Toolkit. *Microsoft Technical Report MSR-TR-2014-112*.
- 63 Bastien, F., Lamblin, P., Pascanu, R., Bergstra, J., Goodfellow, I., Bergeron, A., Bouchard, N., Warde-Farley, D., and Bengio, Y. (2012) Theano: new features and speed improvements. *arXiv preprint arXiv:1211.5590*.
- 64 Collobert, R., Kavukcuoglu, K., and Farabet, C. (2011) Torch7: A Matlab-like environment for machine learning, in *BigLearn*, NIPS Workshop, EPFL-CONF-192376.
- 65 Chollet, F. et al. (2015) Keras, <https://github.com/fchollet/keras>.
- 66 Amiriparian, S., Gerczuk, M., Ottl, S., Cummins, N., Freitag, M., Pugachevskiy, S., and Schuller, B. (2017) Snore sound classification using image-based deep spectrum features, in *Proceedings of the 18th Annual Conference of the International Speech Communication Association (INTERSPEECH)*, ISCA, Stockholm, Sweden.
- 67 Walt, S.V.D., Colbert, S.C., and Varoquaux, G. (2011) The numpy array: a structure for efficient numerical computation. *Computing in Science & Engineering*, 13 (2), 22–30.
- 68 Hunter, J.D. (2007) Matplotlib: A 2D graphics environment. *Computing In Science & Engineering*, 9 (3), 90–95.
- 69 Freitag, M., Amiriparian, S., Gerczuk, M., Cummins, N., and Schuller, B. (2017) An ‘End-to-Evolution’ Hybrid Approach for Snore Sound Classification, in *Proceedings of the 18th Annual Conference of the International Speech Communication Association (INTERSPEECH)*, ISCA, Stockholm, SE.

Part II

Learning Algorithms for Large-Scale Multimedia

4

Large-Scale Video Understanding with Limited Training Labels

Jingkuan Song, Xu Zhao, Lianli Gao and Liangliang Cao

4.1 Introduction

The challenge of video understanding lies in the gap between large-scale video data and the limited resource we can afford in both label collection and online computing stages. This chapter discusses both unsupervised and semi-supervised methods to facilitate video understanding tasks. In particular we consider two general research problems: video retrieval and video annotation/summarization. While the former focuses on designing efficient algorithms for retrieving useful information from large-scale data, the latter aims to reduce the size of the data. Specifically, video retrieval provides new models and methods for effectively and efficiently searching through the huge variety of video data that are available in different kinds of repositories (digital libraries, Web portals, social networks, multimedia databases, etc.). On the other hand, video annotation/summarization generates annotations or a short summary of a video. It provides efficient storage and quick browsing of a large collection of video data without losing important aspects. Depending on whether or not human labels are involved, video retrieval and annotation/summarization can be further categorized into unsupervised and semi-supervised based methods. In this work, we cover video retrieval and annotation/summarization, two approaches to fight the consequences of the big video data. We present the state-of-the-art research in each research field. Promising future trends on big video understanding are also proposed.

4.2 Video Retrieval with Hashing

4.2.1 Overview

With the rapid development of Internet techniques, video capturing devices, and video editing software, the number of online videos continues to grow at an astonishing speed. Meanwhile, video-related services, such as video sharing, monitoring, advertising, and recommendation, inspire online users' interests and participation in video-related activities, including uploading, downloading, commenting, and searching. A substantial number of videos are uploaded and shared on social websites every day.

Content-based video hashing is a promising direction for content-based video retrieval, but it is more challenging than image hashing [1–4]. Essentially, the rich temporal information in videos is a barrier for directly utilizing image-based hashing methods [5]. A few attempts take a video as a set of images and ignore the temporal information [6–8]. Although the performance is promising, these methods learn hash codes for each frame where the structure of a video is not introduced.

Recently, deep learning has dramatically improved the state of the art in speech recognition, visual object detection and image classification (see chapter 1, ref. [11], [9, 10]). Inspired by this, researchers have started to apply deep features extracted from various deep convolutional neural networks (DCNNs) (e.g., VGG (see chapter 1, ref. [23]) and ResNet (see chapter 3, ref. [60]) to support video hashing. In general, they first conduct pooling on frame-level deep features to generate video-level features, and then project these high-dimensional features into low-dimensional hash codes [11]. Although pooling provides a solution for video hashing generation, it inevitably results in suboptimal binary codes, since video temporal information is significantly ignored. To capture the temporal information, the recurrent neural network (RNN) is used to achieve the state-of-the-art performance in video captioning [12, 13]. RNN-based hashing methods [14] utilize RNN and video labels to learn discriminative hash codes for videos. However, human labels are time- and labor-consuming, especially for large-scale datasets. More recently, Zhang et al. proposed self-supervised temporal hashing (SSTH) [15], which aims to improve video hashing by taking temporal information into consideration. They proposed a stacking strategy to integrate long short-term memory networks (LSTMs) [16] with hashing to simultaneously preserve a video's original temporal and spatial information using a short hash code. In this section, we introduce several video retrieval methods using hashing.

First, we integrate multiple features into hashing to improve the accuracy of hashing for unsupervised video retrieval. Most existing approaches use only a single feature to represent a video for retrieval. However, a single feature is often insufficient to characterize the video content. Besides, while the accuracy is the main concern in previous literature, the scalability of video retrieval algorithms for large-scale video datasets has been rarely addressed. In this section, we present a novel approach, multiple feature hashing (MFH), to tackle both the accuracy and the scalability issues of video retrieval. MFH preserves the local structure information of each individual feature and also globally considers the local structures for all the features to learn a group of hash functions which map the video keyframes into the Hamming space and generate a series of binary codes to represent the video dataset. We evaluate our approach on a public video dataset and a large-scale video dataset consisting of 132,647 videos, which was collected from YouTube by ourselves. The experiment results show that the proposed method outperforms the state-of-the-art techniques in accuracy.

Then, we propose a method called submodular video hashing (SVH), which can be either unsupervised or supervised. Unlike the previous methods, which capitalize on only one type of hash code for retrieval, SVH combines heterogeneous hash codes to effectively describe the diverse and multi-scale visual contents in videos. Our method integrates feature pooling and hashing in a single framework. In the pooling stage, we cast video frames into a set of pre-specified components, which capture a variety of semantics of video contents. In the hashing stage, we represent each video component as a compact hash code, and combine multiple hash codes into hash tables for effective search. To speed up the retrieval while retaining most informative codes, we propose a

graph-based influence maximization method to bridge the pooling and hashing stages. We show that the influence maximization problem is submodular, which allows a greedy optimization method to achieve a nearly optimal solution. Our method is extensively evaluated in both unsupervised and supervised scenarios, and the results on TRECVID multimedia event detection and columbia consumer video datasets demonstrate the success of our proposed technique.

4.2.2 Unsupervised Multiple Feature Hashing

4.2.2.1 Framework

The proposed framework based on MFH [17] for video retrieval is shown in Figure 4.1 and comprises two phases. In the first phase, which is offline, we use the proposed MFH algorithm to learn a series of s hash functions $\{h_1(\cdot), \dots, h_s(\cdot)\}$, each of which generates one bit hash code for a keyframe according to the given multiple features. Each keyframe has s bits. Using the derived hash functions $\{h_1(\cdot), \dots, h_s(\cdot)\}$, each keyframe for a dataset video can be represented by the generated s -sized hash codes in linear time. In the second phase, which is online, the query video's keyframes are also represented by s -sized hash codes mapped from the s hash functions. Video retrieval can be efficiently achieved where only efficient XOR operation on the hash codes is performed to compute the similarity between two videos.

The key research issue is how to train the hash functions which affect both accuracy and efficiency. In this section, we detail the proposed MFH algorithm. It is worthy noting that MFH is a general one which can be potentially applied to other large-scale search applications where the data are represented by multiple features.

4.2.2.2 The Objective Function of MFH

Let v be the number of features.¹ Given an integer $g \leq v$, $(x^g)_t \in \mathbb{R}^{d_g \times 1}$ denotes the g th feature of t th training data, where d_g is the dimensionality of the g th feature. Suppose there are n training keyframes in total. $X^g = [(x^g)_1, (x^g)_2, \dots, (x^g)_n] \in \mathbb{R}^{d_g \times n}$ is the feature matrix corresponding to the g th feature of all the training data.

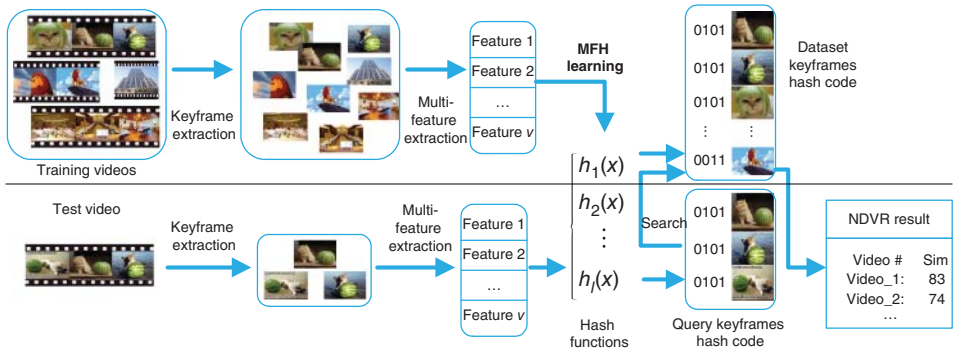


Figure 4.1 The proposed framework for video retrieval.

¹ In our system, each keyframe is represented by the local feature Local binary pattern (LBP) and the global feature hue, saturation, value (HSV) color histogram. Therefore, $v = 2$ here. Yet, MFH is able to deal with more feature types when $v > 2$.

$x_t = [(x_1^t)^T, (x_2^t)^T, \dots, (x_v^t)^T]^T \in \mathbb{R}^{d \times 1}$, where $d = \sum_{g=1}^v d_g$ is the vector representation of the t th training keyframe using all of the features and $X = [x_1, x_2, \dots, x_n] \in \mathbb{R}^{d \times n}$. MFH aims to learn a series of hash functions $\{h_1(\cdot), \dots, h_s(\cdot)\}$, where s is the number of hash functions, i.e., the length of the hash code. Each of the s hash functions generates one bit hash code of the input keyframe. We further denote $Y = [(y_1)^T, \dots, (y_n)^T]^T \in \mathbb{R}^{s \times n}$ as the hash codes of the training keyframes corresponding to all features and $Y^g = [(y_1^g)^T, \dots, (y_n^g)^T]^T \in \mathbb{R}^{s \times n}$ as the hash codes of the training keyframes derived from the g th feature.

To exploit the individual structural information of each individual feature, we define v affinity matrices $A^g \in \mathbb{R}^{n \times n}$ ($1 \leq g \leq v$), one for each feature as follows:

$$(A^g)_{pq} = \begin{cases} 1, & \text{if } (x^g)_p \in \mathcal{N}_k((x^g)_q) \text{ or } (x^g)_q \in \mathcal{N}_k((x^g)_p) \\ 0, & \text{else} \end{cases} \quad (4.1)$$

where $\mathcal{N}_k(\cdot)$ is the k -nearest-neighbor set and $1 \leq (p, q) \leq n$.

To learn the hash codes $(y^g)_t$ from the g th feature, a reasonable criterion is that similar items in the original space should have similar binary codes, which can be formulated as the following minimization problem:

$$\min_{Y^g} \sum_{p,q=1}^n (A^g)_{pq} \|(y^g)_p - (y^g)_q\|_F^2. \quad (4.2)$$

Given a training keyframe x_t , the overall hash codes y_t should be consistent with the hash codes $y_t^g|_{g=1}^v$ derived from each feature. In this way, the local structure information on each single feature is also globally considered and optimized. Therefore, we have the following minimization problem:

$$\min_{Y^g, Y} \sum_{g=1}^v \sum_{p,q=1}^n (A^g)_{pq} \|(y^g)_p - (y^g)_q\|_F^2 + \gamma \sum_{g=1}^v \sum_{t=1}^n \|y_t - (y^g)_t\|_F^2 \quad (4.3)$$

where γ is the parameter to balance the two parts. It is easy to see from Eq. 4.3 that the individual manifold structure of each feature type is preserved (the first term) and all the manifold structures are also globally considered in the optimization (the second term).

Recall that we intend to learn the hash codes of the training keyframes and a series of hash functions $\{h_1(\cdot), \dots, h_s(\cdot)\}$ in a joint framework. We propose to simultaneously learn the hash codes of the training keyframes and the hash functions by minimizing the empirical error of the hash functions *w.r.t.* the learned hash codes Y . The proposed final objective function of MFH is given by:

$$\begin{aligned} \min_{Y, Y^g, W, b} \quad & \sum_{g=1}^v \sum_{p,q=1}^n (A^g)_{pq} \|(y^g)_p - (y^g)_q\|_F^2 + \gamma \sum_{g=1}^v \sum_{t=1}^n \|y_t - (y^g)_t\|_F^2 \\ & + \alpha (\|X^T W + \mathbf{1}b - Y\|_F^2 + \beta \|W\|_F^2) \\ \text{s.t.} \quad & y_t \in \{-1, 1\}^s, (y^g)_t \in \{-1, 1\}^s, Y Y^T = I \end{aligned} \quad (4.4)$$

where α and β are the parameters, $\mathbf{1} \in \mathbb{R}^{n \times 1}$ is a column vector with all ones, and y_{li} is l th bit hash code for x_i . In Eq. 4.4, $y_t \in \{-1, 1\}^s$ and $(y^g)_t \in \{-1, 1\}^s$ force the hash codes of y_t and $(y^g)_t$ to be binary codes. As we will show later on, the constraint $Y Y^T = I$ is imposed to avoid the trivial solution. For the simplicity of implementation for large-scale datasets, we adopt the linear transformation as the hash function. More specifically,

given a keyframe represented by its feature x_t , $w_l \in \mathbb{R}^{d \times 1}$ is the transformation matrix and $b_l \in \mathbb{R}$ is the bias term. $W = [w_1, w_2, \dots, w_s] \in \mathbb{R}^{d \times s}$, $b = [b_1, b_2, \dots, b_s] \in \mathbb{R}^{1 \times s}$ and $\mathbf{1}$ is a vector of all ones.

4.2.2.3 Solution of MFH

In this subsection, we detail the approach to optimize the objective function of MFH. By setting the derivative of Eq. 4.4 *w.r.t.* b and W to zero, we get:

$$b = \frac{1}{n}(\mathbf{1}^T Y - \mathbf{1}^T X^T W), \quad W = (XL_c X^T + \beta I)^{-1} XL_c Y, \quad (4.5)$$

where $L_c = I - \frac{1}{n}\mathbf{1}\mathbf{1}^T$ is the centering matrix. Note that $L_c = (L_c)^T = L_c(L_c)^T$. Replacing W and b by Eq. 4.5, we have

$$\begin{aligned} X^T W + \mathbf{1}b - Y &= X^T W + \mathbf{1}\frac{1}{n}(\mathbf{1}^T Y - \mathbf{1}^T X^T W) - Y \\ &= L_c X^T W - L_c Y = L_c X^T (XL_c X^T + \beta I)^{-1} XL_c Y - L_c Y \end{aligned} \quad (4.6)$$

Let $M = (XL_c X^T + \beta I)^{-1}$. $\|X^T W + \mathbf{1}b - Y\|_F^2 + \beta\|W\|_F^2$ can be rewritten as:

$$\|X^T W + \mathbf{1}b - Y\|_F^2 + \beta\|W\|_F^2 = \text{tr} Y^T B Y, \quad B = (L_c - L_c X^T (XL_c X^T + \beta I)^{-1} XL_c) \quad (4.7)$$

Meanwhile, we have

$$\sum_{p,q=1}^n (A^g)_{pq} \|(y^g)_p - (y^g)_q\|_F^2 = \text{tr} ((Y^g)^T L^g Y^g) \quad (4.8)$$

where $L^g = N^g - A^g$ is the Laplacian matrix of the g th feature and N^g is the diagonal matrix with its diagonal element $N_{ii}^g = \sum_j A_{ij}^g$. Note that $\sum_{g=1}^v \sum_{t=1}^n \|y_t - (y^g)_t\|_F^2$ can also be written as $\sum_{g=1}^v \|Y - Y^g\|_F^2$. Then the objective function shown in Eq. 4.4 becomes

$$\begin{aligned} \min_{Y, Y^g} \quad & \sum_{g=1}^v (\text{tr}((Y^g)^T L^g Y^g) + \gamma \|Y - Y^g\|_F^2) + \alpha \text{tr}(Y^T B Y) \\ \text{s.t.} \quad & Y^T Y = I \end{aligned} \quad (4.9)$$

Setting the derivative of Eq. 4.9 *w.r.t.* Y^g to be zero, we have

$$2L^g Y^g - 2\gamma(Y - Y^g) = 0 \Rightarrow Y^g = \gamma(L^g + \gamma I)^{-1} Y \quad (4.10)$$

Let $C^g = \gamma(L^g + \gamma I)^{-1}$. Note that $C^g = (C^g)^T$. Then we arrive at

$$Y^g = C^g Y \quad (4.11)$$

Replacing Y^g with Eq. 4.11, the objective function in Eq. 4.9 becomes:

$$\min_{Y, Y^T=I} \text{tr}(Y^T D Y) \quad (4.12)$$

where D is defined as

$$D = \sum_{g=1}^v (C^g L^g C^g + \gamma(I - C^g)^2) + \alpha B = \gamma \sum_{g=1}^v (I - C^g) + \alpha B \quad (4.13)$$

Algorithm 4.1 The MFH algorithm.

```

for  $g = 1$  to  $v$  do
  for  $p = 1$  to  $n$  do
    for  $q = 1$  to  $n$  do
      Compute  $L_{p,q}^g$  according to Eq. 4.8;
    end for
  end for
end for
Compute  $B$  according to Eq. 4.7;

Compute  $D$  according to Eq. 4.13;
Compute  $Y$  by performing eigenvalue decomposition on  $D$ .
Output  $W$  and  $b$  according to Eq. 4.5.

```

The optimization problem shown in Eq. 4.12 can be solved by performing the eigen-value decomposition of D . Y can be obtained by the s eigenvectors of D corresponding to the s smallest eigenvalues.² In summary, the training processing of the proposed MFH algorithm is listed as follows.

Once W and b are obtained, they can be readily used to generate the hash codes of database keyframes as well as query keyframes. For example, given a keyframe x_t , we first compute its relaxed hash code y_t in continuous domain by $y_t = W^T x_t + b$. Then, we binarize it by comparing each dimension of y_t with its median of all dimensions. If it is greater than the median, we set it as 1. Otherwise, we set it as -1 . To represent a video clip, Wu et al. proposed averaging the visual feature of all its keyframes as the representation of the whole video clip [18]. Following Wu's way, we can also generate the hash codes for a video clip by averaging the relaxed codes of all its keyframes and then binarizing them. As a result, each video clip is represented by a single s -bit hash code. In this work, we use the above video representation to avoid pairwise keyframe comparisons. The number of common bits along all dimensions shared by two videos is approximated as the video similarity.

4.2.2.3.1 Complexity Analysis

In the previous subsections we proposed a multi-feature hashing algorithm MFH for near-duplicate video retrieval (NDVR). During the training phase, to compute the matrix A_1^g , we need to perform a k -nearest-neighbors search for each feature type whose time complexity is $O(k \times n^2)$. Since we have v feature types, the time complexity is $O(v \times k \times n^2)$. Because $v \ll n$ and $k \ll n$, the time complexity for computing A_1^g is $O(n^2)$. To compute the matrix A_2^g , we need to search the n training data points for each data point whose time complexity is $O(n^2)$. Also, we need to perform an eigenvalue decomposition to get Y , whose time complexity is n^3 . Then, we need to solve a linear system when computing W and b , whose time complexity is $O(n^2)$ approximately. Therefore, the time complexity for the training of MFH is $O(n^3)$ approximately. Once the hash functions, i.e., W and b , are obtained, we only need to compute the hash code

2 The trivial solution corresponding to the eigenvalue of zero is removed.

to map the whole database into the Hamming space with a linear time complexity. All of the above steps are done offline.

4.2.3 Submodular Video Hashing

4.2.3.1 Framework

The proposed framework based on SVH for video retrieval is shown in Figure 4.2, which comprises two phases. In the first phase, which is the pooling stage, we cast video frames into a set of pre-specified components, which capture a variety of semantics of video contents. In the second stage, which is the hashing stage, we represent each video component as a compact hash code, and combine multiple hash codes into hash tables for effective search.

4.2.3.2 Video Pooling

The basic idea of our hashing algorithm is to exploit the rich semantics of videos and construct binary codes corresponding to different characteristics of video contents. Our methods contrast with previous image-based hashing methods by considering video feature pooling in frame-level. Suppose there is a video with T frames, where each frame is represented by a feature vector x_t . We generate the temporal pooling component k as:

$$z^k = \sum_{t=1}^T p_t^k x_t^k \quad (4.14)$$

Here p_t^k stands for the probability $Pr(k|x_t)$ that captures the amount of contribution of frame t in forming the scene component k . We approximate the probability using soft

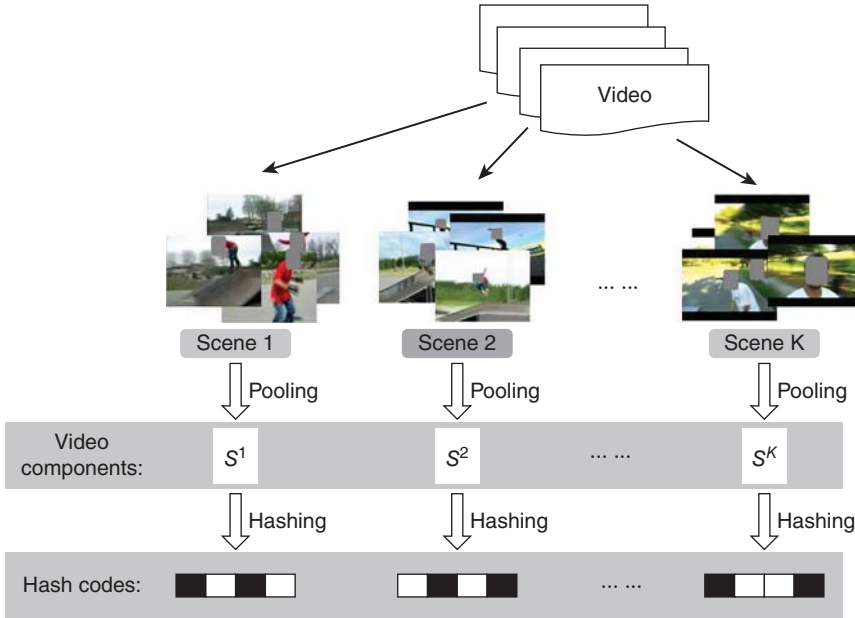


Figure 4.2 The proposed framework of SVH for video retrieval.

vector quantization on GIST features. From Eq. 4.14, we can see that to obtain different scene components $z^k = \sum_{t=1}^T p_t^k x_t$, in fact is equivalent to decomposing each frame feature x_t into different scenes

$$x_t \rightarrow [p_t^1 x_t, p_t^2 x_t, \dots], \quad (4.15)$$

From Eqs 4.14 and 4.15 one can see that our new model is a feature pooling method [19]. The unique characteristic of our method is that our pooling weights p_t^k are on video frames instead of the spatial domain. Here we force the pooling weights p_t^k to follow the constraint $\sum_k p_t^k = 1$. We use GIST features to represent the scenes of video frames and then to compute p_t^k . Note that the pooling weights are computed based on Euclidean distance, which is not reliable for sparse histogram features like scale-invariant feature transform (SIFT). On the other hand, GIST is easy to compute and gives a reliable measure of how different two frames look holistically (see chapter 1, ref. [29]). With all the training data, we compute the GIST features for all frames and cluster them into C centers using the K-means algorithm where the number of C is empirically set. The C center is represented as $g_c^1, g_c^2, \dots, g_c^C$. To compute the pooling weight p_t^k for frame t in a training or test video clip, we extract its GIST feature vector g_t . The pooling weight is derived by comparing g_c^k and g_t . One simple way to compute pooling weights is by vector quantization (VQ), which forces all but the nearest p_t^k to be zero. However, VQ is well-known to be sensitive to noises. Here, we consider the soft voting pooling weights as

$$p_t^k = \frac{1/(d_t^k)^3}{\sum_{l=1}^C 1/(d_t^l)^3} \quad (4.16)$$

where d_t^k is the Euclidean distance between centers g_c^k and frame feature g_t . It is not difficult to see that our pooling weights satisfy the constraint $\sum_k p_t^k = 1$.

4.2.3.3 Submodular Video Hashing

Suppose we have nf features indexed by $f = 1, \dots, nf$, where each (f) is captured by multiple (C) components (v_f^1, \dots, v_f^C) and each component (v_f^j) is expressed by a compact binary hash code¹. Our first observation is that not every code is equally informative and there can be significant redundancy among the codes. On the other hand, it is critical to use compact representation for large-scale video retrieval. Inspired by these observations, we aim to select a small number of “informative” codes which can well “represent” the entire set of codes. We found these concepts can be effectively modeled using a graph over the codes whose edge weights capture the degree of overlapping or similarity between pairwise codes. In this section, we assume that the mechanism to compute pairwise code similarity is known (which will be addressed in the next subsection).

Let $\mathbb{G} = \{\mathbb{V}, \mathbb{W}\}$ be a graph with node set \mathbb{V} consisting of the hash codes considered, i.e., $\mathbb{V} = \{v_f^k | k = 1, \dots, C; f = 1, \dots, nf\}$, and similarity matrix $\mathbb{W} = [w_{ij}]$, where w_{ij} captures the similarity between codes i and j . Our goal is to select m representative nodes from the graph, where m can be manually specified or determined automatically. We cast this node selection problem as an influence maximization problem where the influence of the selected m nodes can be maximally propagated to the rest of the nodes in the graph.

Denote \mathbb{A} as the set of selected nodes whose influences are assigned and fixed to 1's. We use a score $u_{\mathbb{A}}(i) \in [0, 1]$ to quantify the influence node i received from \mathbb{A} . For technical reason, that will be clear soon, we introduce a sink node s to the graph that is

connected to each node with a small constant weight. The sink node s is very “cool” in that it is never influenced by others or tries to influence others. So its influence is fixed to 0. For simplicity and a little abuse of notations, we still denote the graph as $\mathbb{G} = \{\mathbb{V}, \mathbf{W}\}$ but keep in mind that $s \in \mathbb{V}$ and \mathbf{W} is expanded accordingly. We also denote $\hat{\mathbb{A}} = \mathbb{A} \cup s$ and \mathbb{A} as the set of remaining nodes, i.e., $\hat{\mathbb{A}} \cap \mathbb{N} = \emptyset$ and $\hat{\mathbb{A}} \cap \mathbb{N} = \mathbb{V}$. Formally, we propose the following influence maximization framework:

$$\max_{\mathbb{A} \subset \mathbb{V}} \Omega(\mathbb{A}) := \sum_{i \in \mathbb{N}} u_{\mathbb{A}}(i) \quad (4.17)$$

To instantiate the above framework, a concrete model for influence propagation is needed. Specifically, we use the concept of harmonic fields from Zhu et al. [20], which has been shown to be a highly effective label propagation model. Denote $u_{\mathbb{A}}(i) = 1$ if $i \in \mathbb{A}$ and $u_{\mathbb{A}}(s) = 0$. Our influence model enjoys a “harmonic” interaction among the nodes as follows:

$$u_{\mathbb{A}}(i) = \frac{1}{d_i} \sum_{j \in \mathbb{N}(i)} w_{ij} u_{\mathbb{A}}(j), i \in \mathbb{N} \quad (4.18)$$

where $\mathbb{N}(i)$ denotes the set of neighbors of node i and $d_i = \sum_{j \in \mathbb{N}(i)} w_{ij}$ is the degree of node i . In other words, the influence on node i is the weighted sum of the influences on its neighbors.

Since our objective function (4.17) for the hash function selection problem is submodular [8], we can obtain a nearly optimal combination of hash codes by a greedy algorithm. We start from an empty set $A_0 := \emptyset$ and iteratively add a code y to \mathbb{A}_i that maximizes the increase in influence of Ω . Specifically, we select the node y_{i+1} in step $i + 1$ using the following criterion:

$$y_{i+1} = \arg \max_{y \in \mathbb{N}_i} \Omega(\mathbb{A}_i \cup \{y\}) - \Omega(\mathbb{A}_i) \quad (4.19)$$

4.2.4 Experiments

We experimentally evaluated the performance of MFH and SVH and compared it with existing state-of-the-art methods.

4.2.4.1 Experiment Settings

4.2.4.1.1 Video Datasets

We used two datasets in the experiments.

The CC_WEB_VIDEO Dataset [21] is provided by City University of Hong Kong and Carnegie Mellon University. It consists of 24 sets of video clips (13,129 video clips in total) downloaded from video-sharing websites.

The Combined Dataset is a larger video dataset created by ourselves by adding CC_WEB_VIDEO to our video dataset downloaded from YouTube.

4.2.4.1.2 Visual Features

In the CC_WEB_VIDEO dataset, a shot-based sampling method was used to extract a total of 398,078 keyframes. These keyframes are provided in the dataset. We further extracted the LBP feature on the keyframes as a local feature, which will be used together with the existing global feature HSV provided by the dataset.

In the Combined Dataset, we firstly detected the shots from which the keyframes were extracted. Then we extracted two features for each keyframe, HSV and LBP, to be used as a global feature and a local feature, respectively. HSV is a global color histogram with 162 dimensions and LBP is a local content feature with 256 dimensions.

4.2.4.1.3 Algorithms for Comparison

To evaluate video retrieval accuracy and efficiency, we present an extensive comparison of the proposed method with a set of existing video retrieval methods, which are briefly described as follows.

We use the global color histogram method as a baseline method. Each keyframe is represented as a color histogram feature vector. Each video is finally defined as a feature vector of a normalized color histogram over all the keyframes in the video. We also compared the proposed method with the local feature based method [18], the spatiotemporal feature based method [22], self-taught hashing [23] and spectral hashing [24].

Following the work in [22], we also adopted the mean average precision (MAP) metric to evaluate the performance. We further used a precision-recall curve to evaluate the performance on both datasets.

4.2.4.2 Results

4.2.4.2.1 CC_WEB_VIDEO

We first tested different methods on the CC_WEB_VIDEO dataset. The same 24 queries as in [18, 22] were used. The MAP results of different methods can be seen in Table 4.1.

From Table 4.1, we can see that the proposed MFH achieves the best results on MAP, although the margins are minor. This is probably because the dataset is not big enough and thus less noise can be introduced for most methods. The performance of different methods cannot be fully explored. Next, we focus on the large dataset.

4.2.4.2.2 Combined Dataset

To further test the accuracy and scalability of MFH, we tested MFH on the Combined Dataset and compare it with existing methods. More specifically, we used the same 24 queries and ground truth as in [18, 22] to search the whole Combined Dataset. Several parameters need to be tuned for some of the algorithms. For each parameter we have a set of candidate values and we iteratively combined the parameters to form various parameter settings. For each parameter combination we ran the same experiment to evaluate its impact on the performance. Finally, the parameter setting which results in the best performance was selected for each method. For the parameters in MFH, SPH and STH, including γ, α, β , we tuned them from $10^{-6}, 10^{-3}, 10^0, 10^3, 10^6$ and tuned the

Table 4.1 Comparison of MAP on CC_WEB_VIDEO.

Methods	GF	LF	ST_lbp	ST_ce	MFH
MAP	0.892	0.952	0.953	0.950	0.954

GF, global feature; LF, ST_lbp, ST_ce, MFH, see [18] and [22].

Table 4.2 Comparison of MAP and time for the Combined Dataset.

Methods	MAP	Time(s)
SPH	0.5941	0.4907
GF	0.6466	1.3917
STH	0.7536	0.6439
MFH_lbp	0.7526	0.6445
MFH_hsv	0.8042	0.4508
MFH	0.8656	0.5533

GF, global feature; SPH, (see chapter 9, ref. [24]); STH, see [23]; MFH, see [7].

length of the binary code from 200 to 400. On this large Combined Dataset, we cannot compare with the local feature based method LF proposed in [18] because this method utilizes interest points matching to measure the similarity of two keyframes. This is impractical in such a large dataset with 2,570,554 keyframes. We were also unable to extract the spatiotemporal features ST_lbp and ST_ce because it would take an extremely long time for our computer to extract the features from 132,647 videos. As a result, only the GF, STH, SPH, MFH_hsv, MFH_lbp, and MFH methods are compared.

The results for MAP and search time are shown in Table 4.2. We have the following observations:

- MFH achieves the highest MAP and outperforms existing methods by a large margin. MFH can have more than 86% MAP, while SPH has less than 60% MAP.
- SPH, STH, and MFH all construct binary codes for the videos and perform the search using the Hamming space. Even in the Matlab environment they can search the large dataset within 1 second. However, SPH's performance is really unsatisfactory probably because the video dataset is not in accordance with the assumption that all the data points are uniformly distributed in a hyper-rectangle in high-dimensional space. This also shows that MFH is more capable of preserving local data distribution information.
- GF performs the search using the Euclidean distance and its performance is not good. Its accuracy is much worse than that of MFH. Furthermore, its search time is two to three times longer than that of the hashing methods.
- MFH outperforms MFH_lbp and MFH_hsv in general, which demonstrates the effectiveness of combining multiple features in video retrieval.

4.2.4.3 Evaluation of SVH

We also experimentally evaluated the performance of SVH and compare it with existing state-of-the-art methods on two datasets.

The TRECVID Multimedia Event Detection (MED 2011) dataset is a large annotated dataset specifically designed to model complex video events. The evaluation of MED is separated into two test sets: the mid-size dryrun evaluation and the large final evaluation set.

The Columbia Consumer Video (CCV) dataset [25] contains 9317 YouTube videos over 20 semantic categories.

We applied the unsupervised measure for the MED dataset and the semi-supervised measure for CCV dataset. Since each video in the CCV dataset is assigned a label from 20 semantic categories, we evaluated retrieval performance for all categories.

4.2.4.3.1 Results

MED dataset The results in Figure 4.3a show that our submodular hashing method is significantly better. This confirms the crucial benefit of employing multiple scene components for video representation and the submodular selection strategy for hash table construction. The results also show that the performance of low-level features such as LBP and color histogram are inferior to SIFT histogram, which motivated us to focus on powerful features like SIFT and its variants in the final run.

The results for the final evaluation dataset are shown in Figure 4.3b, which shows that semantic feature vectors work better than SIFT features for video retrieval, but our submodular hash methods still clearly boost the performance over these features. The results also show that random projection works better than spectral hashing, which is probably because the objective function of spectral hashing applies to the Hamming distances between all the samples, of which a large proportion has been ignored in our hash table structure. Despite that, the contributions of the video component and submodular selection are consistent in both cases.

In this dataset, we employed three features: sparse sift, dense sift, and pyramid histogram of oriented gradients (PHOG). Since the similarity of different hash codes depends on the category label, we also compared the retrieval performance for each category. To make a fair comparison, we report the results using four hash tables. The average recall is used to measure retrieval accuracy, as we did on the MED dataset. As shown in Figure 4.4, our submodularity based video hashing is consistently better than hashing accumulated video features.

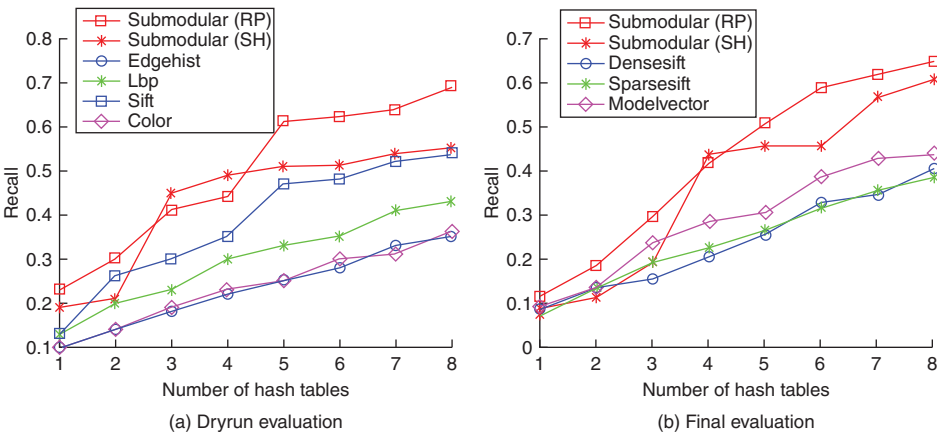


Figure 4.3 Comparison of recall on MED dataset.

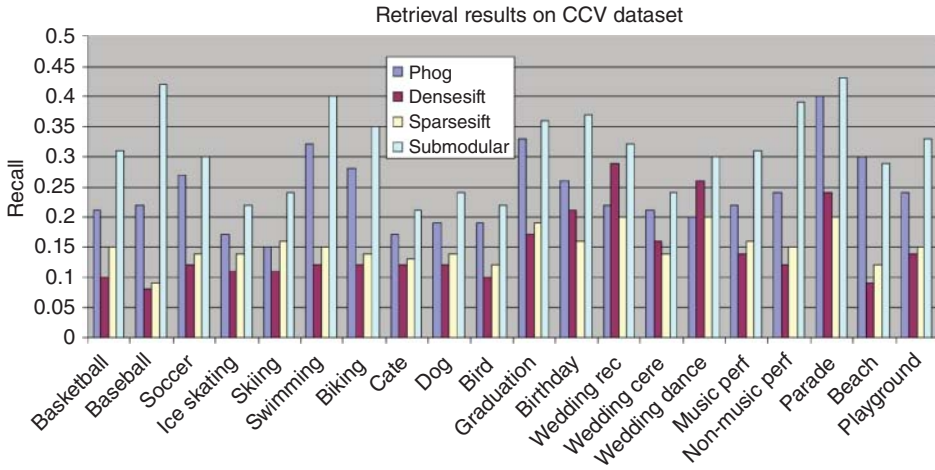


Figure 4.4 Video retrieval performance on the CCV dataset.

4.3 Graph-Based Model for Video Understanding

4.3.1 Overview

Recently, we have witnessed an exponential growth of user-generated videos and images due to the booming use of social networks such as Facebook and Flickr. There are also some well-known public datasets for large-scale video understanding [26]. Consequently, there are increasing demands to effectively organize and access these multimedia data via annotation/summarization. While video retrieval provides methods for effectively and efficiently searching through the huge variety of video data that are available in different kinds of repositories, video annotation/summarization/captioning generates annotations or a short summary of a video. It provides efficient storage and quick browsing of large collection of video data without losing important aspects.

Graph-based learning is an efficient approach for modeling data in various machine learning schemes, i.e., unsupervised learning [27, 28], supervised learning [29], and semi-supervised learning (SSL) [28, 30]. An important advantage of working with a graph structure is its ability to naturally incorporate diverse types of information and measurements, such as the relationship between unlabeled data, labeled data, or both labeled and unlabeled data.

By exploiting a large number of unlabeled data with reasonable assumptions, SSL can reduce the need for expensive labeled data and thus achieve promising results, especially for noisy labels. The harmonic function approach [20] and local and global consistency (LGC) [31] are two representative graph-based SSL methods. The harmonic function approach [20] emphasizes the harmonic nature of the energy function and LGC considers the spread of label information in an iterative way. While these two methods are transductive, manifold regularization (MR) [32] is inductive. In practice, MR extends regression and SVM to SSL methods such as Laplacian regularized least

squares (LapRLS) and Laplacian support vector machines (LapSVM), respectively. By adding a geometrically based regularization term [33]. In [34], a so-called correlation component manifold space learning (CCMSL) was developed to learn a common feature space by capturing the correlations between heterogeneous databases. In [35], a content similarity based fast reference frame selection algorithm was proposed for reducing the computational complexity of the multiple reference frames based inter-frame prediction.

First, we propose to annotate videos in a semi-supervised way. Specifically, we propose learning an optimal graph (OGL) from multi-cues (i.e., partial tags and multiple features), which can more accurately encode the relationships between data points. Then, we will incorporate the learned optimal graph with the SSL model and further extend this model to address out-of-sample extension and noisy label issues.

We specifically address the problem of annotating actions in videos using a graph-based model. We present a context-associative approach to recognize activity with human-object interaction. The proposed system can recognize incoming visual content based on the previous experienced activities.

Finally, we address the problem of video summarization. We have developed a novel approach, termed *event video mashup* (EVM), to automatically generate a unified short video from a collection of Web videos to describe the storyline of an event. To advance research on animated GIF understanding, we collected a new dataset, Tumblr GIF (TGIF), with 100K animated GIFs from Tumblr and 120K natural language descriptions obtained via crowdsourcing.

4.3.2 Optimized Graph Learning for Video Annotation

4.3.2.1 Framework

In this section we introduce our OGL method [36], which consists of two phases (see Figure 4.5). First, a similarity graph is constructed on each feature (multiple feature graph) and also on the partial tags (partial label graph) to exploit the relationship among the data points. Partial tags means that tags are provided only for a part of the training data. Then, optimal graph learning is applied to these graphs to construct an optimal graph, which is integrated with SSL for the task of image and video annotation. Note that in theory our approach can be integrated with all kinds of graph-based algorithms.

4.3.2.2 OGL

4.3.2.2.1 Terms and Notations

We first introduce the notations which will be used in the rest of the section. $X = \{x_1, x_2, \dots, x_n\}$ represents a set of n images, $y_i = \{0, 1\}^c$ is the label for the i th image ($1 \leq i \leq n$), and c is the number of annotations/classes. The first l points x_i ($i \leq l$) are labeled as Y_l , and the remaining u points x_i ($l + 1 \leq i \leq n$) are unlabeled. The goal of transductive graph-based SSL is to predict the label F_u of the unlabeled points. Define a $n \times c$ matrix $F = \begin{bmatrix} F_l \\ F_u \end{bmatrix}$ with $F_l = Y_l$ and $F_u = \{0\}^{u \times c}$.

Suppose that for each image, we have v features. Let $X^t = \{x_i^t\}_{i=1}^n$ denote the feature matrix of the t th view of training images, where $t \in \{1, \dots, v\}$.

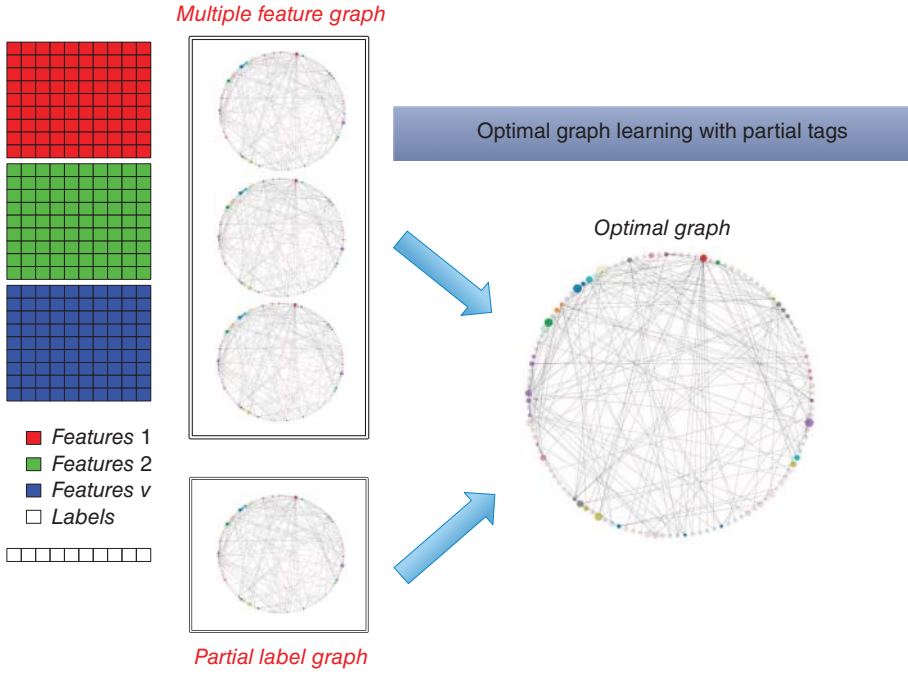


Figure 4.5 The overview of OGL.

4.3.2.2.2 Optimal Graph-Based SSL

The traditional graph-based SSL usually solves the following problem:

$$\min_{F, F_i=Y_i} \sum_{ij} \|f_i - f_j\|_2^2 s_{ij} \quad (4.20)$$

where f_i and f_j are the labels for the i th and j th images, and S is the affinity graph with each entry s_{ij} representing the similarity between two images. The affinity graph $S \in \mathbb{R}^{n \times n}$ is usually defined as follows:

$$s_{ij} = \begin{cases} e^{-\|x_i - x_j\|_2^2 / 2\sigma^2}, & \text{if } x_i \in \mathcal{N}_K(x_j) \text{ or } x_j \in \mathcal{N}_K(x_i) \\ 0, & \text{else} \end{cases} \quad (4.21)$$

where $\mathcal{N}_K(\cdot)$ is the K nearest-neighbor set and $1 \leq (i, j) \leq n$. The variance σ will affect the performance significantly and is usually empirically tuned. Also, the similarity graph is often derived from a single information cue. To address these issues, we propose to learn an optimal graph S from multiple cues.

The multiple cues include the given label information F and the multiple features $X^t = \{x_i^t\}_{i=1}^n$. An optimal graph S should be smooth on all these information cues, which can be formulated as:

$$\min_{S, \alpha} g(F, S) + \mu \sum_{t=1}^v \alpha^t h(X^t, S) + \beta r(S, \alpha) \quad (4.22)$$

where $g(F, S)$ is the penalty function to measure the smoothness of S on the label information F and $h(X^t, S)$ is the loss function to measure the smoothness of S on the feature X^t . $r(S, \alpha)$ are regularizers defined on the target S and α . μ and β are balancing parameters, and α^t determines the importance of each feature.

The penalty function $g(F, S)$ should be defined in such a way that close labels have high similarity and vice versa. In this section, we define it as follows:

$$g(F, S) = \sum_{ij} \|f_i - f_j\|_2^2 s_{ij} \quad (4.23)$$

where f_i and f_j are the labels of data points x_i and x_j . Similarly, $h(X^t, S)$ can be defined as:

$$h(X^t, S) = \sum_{ij} \|x_i^t - x_j^t\|_2^2 s_{ij} \quad (4.24)$$

Note that for simplicity we use the distance-based method to learn the similarity graph here. Other options based on reconstruction coefficients methods can be utilized to achieve better performance, which is discussed in the next section. Instead of preserving all the pairwise distances, we consider preserving the pair distances of the K nearest-neighbors here, i.e., if x_i^t and x_j^t (or f_i and f_j) are not K nearest-neighbors of each other, their distance will be set to a large constant. The regularizer term $r(S, \alpha)$ is defined as:

$$r(S, \alpha) = \frac{\mu\gamma}{\beta} \|S\|_F^2 + \|\alpha\|_2^2 \quad (4.25)$$

We further constrain that $S \geq 0$, $S\mathbf{1} = \mathbf{1}$, $\alpha \geq 0$ and $\alpha^T \mathbf{1} = 1$, where $\mathbf{1} \in \mathbb{R}^{N \times 1}$ is a column vector with all ones. Then we can obtain the objective function for learning the optimal graph by replacing $g(F, S)$, $h(X^t, S)$ and $r(S, \alpha)$ in Eq. 4.22 using Eqs. 4.23, 4.24 and 4.25. By combining Eq. 4.20 with Eq. 4.22, we can obtain the objective function for optimal-graph based SSL, as follows:

$$\begin{aligned} \min_{S, F, \alpha} \quad & \sum_{ij} \|f_i - f_j\|_2^2 s_{ij} + \beta \|\alpha\|_2^2 + \mu \left(\sum_{tij} (\alpha_t \|x_i^t - x_j^t\|_2^2 s_{ij}) + \gamma \|S\|_F^2 \right) \\ \text{s.t.} \quad & \{ S \geq 0, S\mathbf{1} = \mathbf{1}, F\mathbf{1} = Y, \alpha \geq 0, \alpha^T \mathbf{1} = 1 \} \end{aligned} \quad (4.26)$$

4.3.2.2.3 Iterative Optimization

We propose an iterative method to minimize the above objective function in Eq. 4.26. First, we initialize $S = \sum_t S^t / v$ with each S^t being calculated using Eq. 4.21, and we initialize $\alpha^t = 1/v$. We further normalize S as $S = (D^{1/2})^T S D^{1/2}$. Once these initial values are given, in each iteration, we update F given S and α , and then update S and α by fixing the other parameters. These steps are described below.

Update F By fixing S and α , we can obtain F by optimizing Eq. 4.26. This is equivalent to optimizing the following objective function:

$$\min_{F, F\mathbf{1}=Y} \sum_{ij} \|f_i - f_j\|_2^2 s_{ij} = \min_{F, F\mathbf{1}=Y} \|F(I - S)F^T\|_F^2 \quad (4.27)$$

where I is an identity matrix. Let $L = I - S$, and differentiate the objective function in Eq. 4.27 with respect to F , to obtain:

$$LF = 0 \Rightarrow \begin{bmatrix} L_{ll} & L_{lu} \\ L_{ul} & L_{uu} \end{bmatrix} \begin{bmatrix} F_l \\ F_u \end{bmatrix} = 0, \Rightarrow L_{ll}F_l + L_{lu}F_u = 0, L_{ul}F_l + L_{uu}F_u = 0 \quad (4.28)$$

Then we can obtain:

$$F_u^* = -L_{uu}^{-1}L_{ul}F_l \quad (4.29)$$

Update S By fixing F and α , we can obtain S by optimizing Eq. 4.26. This is equivalent to optimizing the following objective function:

$$\sum_{ij} \|f_i - f_j\|_2^2 s_{ij} + \mu \sum_{tij} (\alpha_t \|x_i^t - x_j^t\|_2^2 s_{ij}) + \mu\gamma \|S\|_F^2 \quad (4.30)$$

and it is equivalent to:

$$\min_{S, S \geq 0, S \mathbf{1} = \mathbf{1}} \sum_i \left\| s_i + \frac{a_i + \mu b_i}{2\mu\gamma} \right\|_2^2 \quad (4.31)$$

where $b_i = \{b_{ij}, 1 \leq j \leq n\}$ with $b_{ij} = \sum_t \alpha_t \|x_i^t - x_j^t\|_2^2$ and $a_i = \{a_{ij}, 1 \leq j \leq n\} \in R^{1 \times n}$ with $a_{ij} = \|y_i - y_j\|_2^2$.

The problem in Eq. 4.31 is simplex and we use the accelerated projected gradient method to linearly solve it. The critical step of the projected gradient method is to solve the following proximal problem:

$$\min_{x \geq 0, x^T \mathbf{1} = 1} \frac{1}{2} \|x - c\|_2^2 \quad (4.32)$$

This proximal problem can be solved using the Karush-Kuhn-Tucker approach. Then each s_i can be efficiently solved, and we can get the updated graph S .

Update α By fixing F and S , we can obtain α by optimizing Eq. 4.26. This is equivalent to optimizing the following objective function:

$$\min_{\alpha \geq 0, \alpha^T \mathbf{1} = 1} \mu \sum_t \alpha_t \left(\sum_{ij} \|x_i^t - x_j^t\|_2^2 s_{ij} \right) + \beta \|\alpha\|_2^2, \Rightarrow \min_{\alpha \geq 0, \alpha^T \mathbf{1} = 1} \mu d\alpha + \beta \|\alpha\|_2^2 \quad (4.33)$$

where $d = \{d_t, 1 \leq t \leq v\}$ with $d_t = \sum_{ij} \|x_i^t - x_j^t\|_2^2 s_{ij}$. It can be reformulated in the form of the problem in Eq. 4.32 and can be solved similarly to obtain α .

We update F , S and α iteratively until the objective function Eq. 4.26 converges.

4.3.3 Context Association Model for Action Recognition

While the previous section discusses video annotation, this section focuses on action recognition in videos. It is inspired by the context memory model (CMM) [37] and we can improve CMM to explore video sequences and organize the episodic information for recognition by recollection. In this section, we describe two different memory units in CMM, and propose our framework [38] based on them. The data in memory are processed and maintained under this framework.

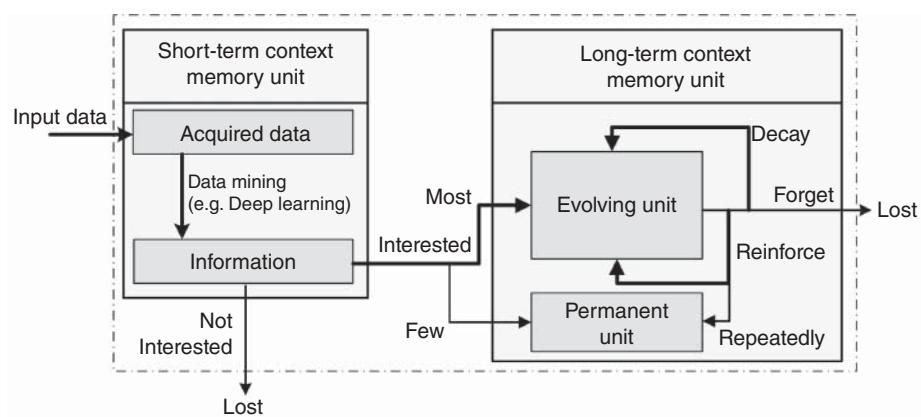


Figure 4.6 Framework of our context memory. The sensory memory is presented for acquiring data and incorporated into the short-term context memory.

4.3.3.1 Context Memory

There are two different memory units in CMM: short-term context memory (SCM) and long-term context memory (LCM) units, as shown in Figure 4.6. The sensory memory [39] is mainly used for extracting information from input data, but is omitted in CMM. We incorporated the data mining part as the sensory model into SCM for acquiring representations. Because permanent memory is rare, we focus on SCM and *evolving units*.

In this work, data acquiring, processing, and maintaining are incorporated into one framework, as shown in Figure 4.6. SCM acquires and processes data to find meaningful and useful information. Many algorithms fit into SCM, such as those in human detection, object tracking, action recognition, and so on. Only the messages of interest from the obtained information will be structured and stored in LCM, and these are used for retrieval. The data in LCM are always under memory evolution, where some can decay or be forgotten, and some can be reinforced.

Hierarchical structure The evolving unit is a key part. Its context structure organizes the detected context elements into an entity based on the relationships. Figure 4.7 shows the summary of the hierarchy.

The basic element of context memory is *attribute* (e.g., an object, a person). An *instance* is formed by several related attributes (e.g., a person is moving a box). Several temporally consecutive instances can form a *cluster* (e.g., an event “arrange objects” consists of “a person is reaching a box,” “a person is moving a box,” “a person is putting a box” and so on). At a certain time, the memory comprises many clusters, and we call this a *snapshot*, which is used for memory evolution. The context memory is dynamic along the timeline, which means that the data in the memory can decay, be reinforced, or be forgotten.

Temporal relations Only the static relations of attributes and instances are modeled in CMM, where the context cluster is defined as a group of a set of context instances with the similarity over a threshold, as shown in Figure 4.8a. However, the temporal relations are important to depict an event in memory, and our context cluster is composed of an ordered sequence of instances, which represents a whole event with temporal relations,

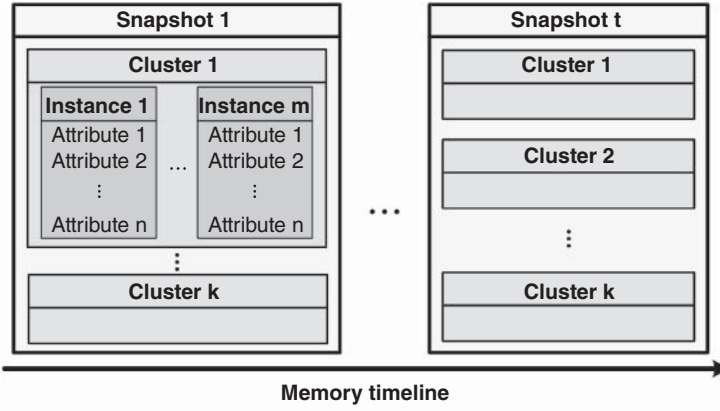


Figure 4.7 Context structure in evolving units of LCM. A snapshot is all the contents in memory at a specific time. The context snapshot evolves by time.

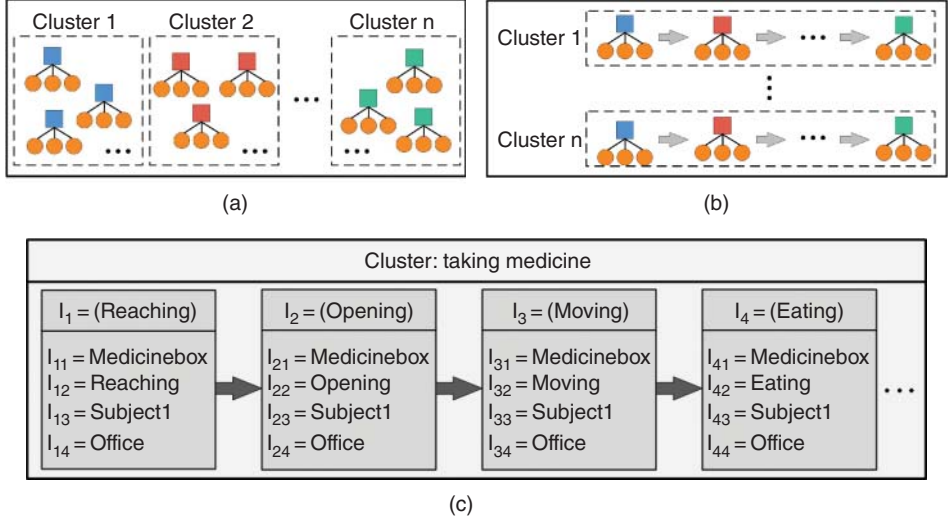


Figure 4.8 Different definitions of the *context cluster*. The similar instances are grouped into a cluster in CMM [37], as shown in (a). (b) shows the proposed cluster, which is an ordered sequence of instances, modeling the temporal relations. (c) shows an example of our cluster.

as shown in Figure 4.8b. One example is given in Figure 4.8c, where the activity of taking medicine is described by a series of consecutive semantic segments (instances). The organized structures and relations are then used for activity recognition.

4.3.4 Graph-based Event Video Summarization

4.3.4.1 Framework

Figure 4.9 gives an overview of our proposed mashup system [40]. Given a collection of videos, the shots of videos are first grouped into shot cliques using a graph-based near-duplicate detection approach. Then the semantics of those identified shot cliques

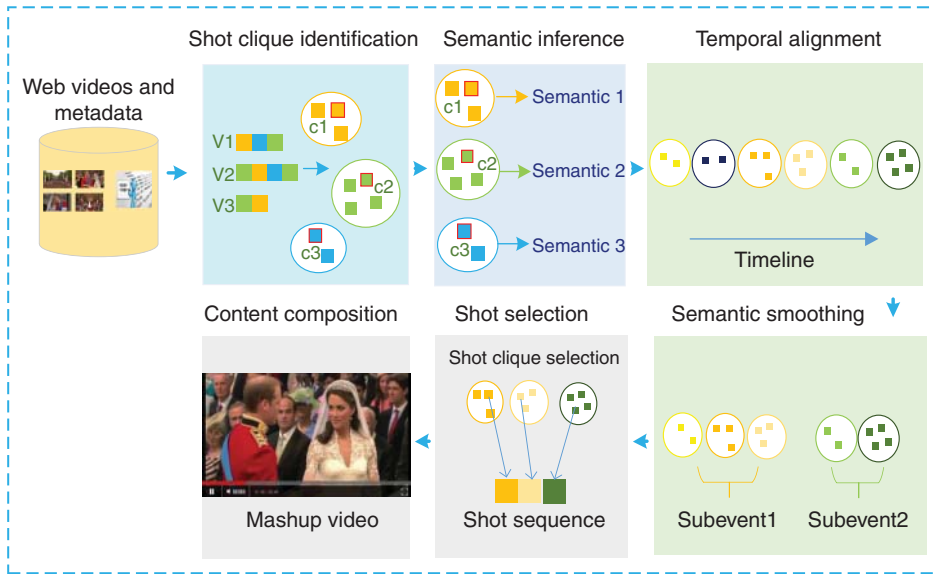


Figure 4.9 The flowchart of the proposed event video mashup approach.

are inferred to unveil their relevance to key semantic facets of the event. Next, the shot cliques are temporally aligned, followed by a semantic smoothing step to refine the alignment and discover key subevents of the event. Taking into consideration both importance and diversity factors, a shot clique selection method is performed. From each top ranked shot clique, a shot is chosen to represent the clique. All the above steps can be generalized as content selection. Finally, a sequence of selected shots in their previously determined order compose the final mashup video, with transition smoothness being considered. The mashup video length can be adaptively adjusted by varying the number of top ranked shot cliques. Among these components, graph-based temporal alignment is a key component. Next, we introduce it.

4.3.4.2 Temporal Alignment

In this subsection, we aim to align the shot cliques in a temporal order.

First, we build a matrix $L \in \mathbb{R}^{n \times n}$ based on the pairwise temporal orders of shot cliques obtained from the original videos as:

$$L_{i,j} = \begin{cases} 1 & \text{if } s_i \text{ is before } s_j, \\ -1 & \text{if } s_i \text{ is after } s_j, \\ 0 & \text{if not determined} \end{cases} \quad (4.34)$$

where $L_{i,j}$ is element (i, j) of L , s_i denotes the i th shot clique, and there are n shot cliques. The temporal orders of some shot clique pairs cannot be directly observed from the original videos, but they may be inferred via some bridging shot cliques. Based on this, we conduct matrix completion for L :

$$L_{i,j} = \begin{cases} 0 \rightarrow +1 & \text{if } L_{i,p} = 1, L_{p,j} = 1 \\ 0 \rightarrow -1 & \text{if } L_{i,p} = -1, L_{p,j} = -1 \end{cases} \quad (4.35)$$

Next, we can align the shot cliques using the local temporal information contained in L . Specifically, we assign a temporal position t_i ($t_i \in \{1, 2, \dots, n\}, t_i \neq t_j$ if $i \neq j$) to each shot clique s_i in order to maximize the match between the mined temporal orders L and the estimated temporal orders:

$$t = \arg \max_{t_i, t_j \in \{1, 2, \dots, n\}} \sum_{i=1}^n \sum_{j=1}^n \text{sgn}(t_j - t_i) L_{i,j} \quad (4.36)$$

where $\text{sgn}(x)$ is a sign function:

$$\text{sgn}(x) = \begin{cases} 1 & \text{if } x > 0, \\ -1 & \text{if } x < 0 \end{cases} \quad (4.37)$$

Eq. (4.36) is a non-deterministic polynomial-time hardness (NP-hard) problem. Alternatively, we propose an algorithm to approximate the optimal solution. To be specific, we first initialize the temporal positions of the shot cliques randomly. And then swap the temporal positions t_i and t_j of s_i and s_j iteratively as long as (i) t_i and t_j contradict with $L_{i,j}$ and (ii) the swap can increase the objective function.

We regard all the shot cliques as a graph, where the vertices are shot cliques and an edge is added between two vertices s_i and s_j , if $L_{i,j} \neq 0$. Then the graph can be divided into a set of connected subgraphs, which means the inter-temporal-orders of the subgraphs cannot be determined from the videos. In fact, these subgraphs often correspond to independent components of the events. Since users' interest and focus towards an event change with the evolution of the event, we can employ users' interest trend to estimate the temporal orders of the independent sets of shot cliques. Upload time of video is a good indicator of users' interest trend about an event. Therefore, we calculate the average upload time of the videos related to the subgraphs to determine the temporal orders.

4.3.5 TGIF: A New Dataset and Benchmark on Animated GIF Description

In the previous subsections, we described our methods for video understanding, while in this subsection, we introduce a dataset collected by ourselves. With the recent popularity of animated GIFs on social media, there is need for ways to index them with rich metadata. To advance research on animated GIF understanding, we collected a new dataset, Tumblr GIF (TGIF) [41], with 100K animated GIFs from Tumblr and 120K natural language descriptions obtained via crowdsourcing. The motivation for this work is to develop a testbed for image sequence description systems, where the task is to generate natural language descriptions for animated GIFs or video clips.

4.3.5.1 Data Collection

We extracted a year's worth of GIF posts from Tumblr using the public API,³ and cleaned up the data with four filters:

- **Cartoon.** Filters out cartoon content by matching popular animation keywords to user tags.
- **Static.** Discards GIFs that show little to no motion (basically static images). To detect static GIFs, we manually annotated 7K GIFs as either static or dynamic, and trained a random forest classifier based on C3D features [42]. The 5-fold cross validation accuracy for this classifier is 89.4%.

³ <https://www.tumblr.com/docs/en/api/v2>

- **Text.** Filters out GIFs that contain text, e.g., memes, by detecting text regions using the extremal regions detector [43] and discarding a GIF if the regions cover more than 2% of the image area.
- **Dedup.** Computes 64bit discrete cosine transform (DCT) image hash using pHash [44] and applies multiple index hashing [45] to perform a k -nearest-neighbor search ($k = 100$) in the Hamming space. A GIF is considered a duplicate if there are more than 10 overlapping frames with other GIFs. On a held-out dataset, the false alarm rate is around 2%.

Finally, we manually validated the resulting GIFs to see whether there was any cartoon, static, or textual content. Each GIF was reviewed by at least two annotators. After these steps, we obtained a corpus of 100K clean animated GIFs.

4.3.5.2 Data Annotation

We annotated animated GIFs with natural language descriptions using the crowdsourcing service CrowdFlower. We carefully designed our annotation task with various quality control mechanisms to ensure the sentences were both syntactically and semantically of high quality.

A total of 931 workers participated in our annotation task. We only allowed workers from Australia, Canada, New Zealand, UK and the USA in an effort to collect fluent descriptions from native English speakers. Figure 4.10 shows the instructions given to

Task

Below you will see five animated GIFs. Your task is to describe each animated GIF in one English sentence. You should focus solely on the **visual content** presented in the animated GIF. Each sentence should be grammatically correct. It should describe the main characters and their actions, but NOT your opinions guesses or interpretations.

DOs

- Please use only English words. No digits allowed (spell them out, e.g., three).
- Each sentence must contain between 8 and 25 words. Try to be concise.
- Each sentence must contain a verb.
- If possible, include adjectives that describe colors, size, emotions, or quantity.
- Please pay attention to grammar and spelling.
- Each sentence must express a complete idea, and make sense by itself.
- The sentence should describe the main characters, actions, setting, and relationship between the objects.

DONTs

- The sentence should **NOT** contain any digits.
- The sentence should **NOT** mention the name of a movie, film, and character.
- The sentence should **NOT** mention invisible objects and actions.
- The sentence should **NOT** make subjective judgments about the GIF.

Remember, please describe only the visual content presented in the animated GIF. Focus on the main characters and their actions.

Figure 4.10 The instructions for the crowdworkers.

the workers. Each task showed five animated GIFs and asked the worker to describe each with one sentence. To promote language style diversity, each worker could rate no more than 800 images (0.7% of our corpus). We paid 0.02 USD per sentence; the entire crowdsourcing cost less than 4K USD.

Syntactic validation Since the workers provided free-form text, we automatically validated the sentences before submission. We checked that each sentence

- contains at least 8, but no more than 25 words (white space separated)
- contains only ASCII characters
- does not contain profanity (checked by keyword matching)
- is typed, not copy/pasted (checked by disabling copy/paste on the task page)
- contains a main verb (checked by using standard position tagging [46])
- contains no named entities, such as the name of an actor/actress, movie, or country (checked by the Named Entity Recognition results from DBpedia spotlight [47]) and
- is grammatical and free of typographical errors (checked by the LanguageTool⁴).

This validation pipeline ensures sentences are *syntactically* good. But it does not ensure their *semantic* correctness, i.e., there is no guarantee that a sentence accurately describes the corresponding GIF. We therefore designed a semantic validation pipeline, described next.

Semantic validation Ideally, we wanted to validate the semantic correctness of every submitted sentence (as we did for syntactic validation), but this was impractical. Instead we used the “blacklisting” approach, where we identified workers who underperformed and blocked them accordingly.

We annotated a small number of GIFs and used them to measure the performance of workers. We collected a validation dataset with 100 GIFs and annotated each with 10 sentences using CrowdFlower. We carefully hand-picked the GIFs whose visual story was clear and unambiguous. After collecting the sentences, we manually reviewed and edited them to make sure they met our standard.

Using the validation dataset, we measured the semantic relatedness of sentence to GIF using METEOR [48], a metric commonly used within the natural language processing (NLP) community to measure machine translation quality. We compare a user-provided sentence to 10 reference sentences using the metric and accepted a sentence if the METEOR score was above a threshold (empirically set at 20%). This filtered out junk sentences, e.g., “this is a funny GIF taken in a nice day,” but retained sentences with similar semantic meaning as the reference sentences.

We used the dataset in both the qualification and the main tasks. In the qualification task, we provided five GIFs from the validation dataset and approved a worker if they successfully described at least four tests. In the main task, we randomly mixed one validation question with four main questions; a worker was blacklisted if the overall approval rate on validation questions fell below 80%. Because validation questions were indistinguishable from normal task questions, workers had to continue to maintain a high level of accuracy in order to remain eligible for the task.

As we ran the CrowdFlower task, we regularly reviewed failed sentences and, in the case of a false alarm, we manually added the failed sentence to the reference

⁴ <https://languagetool.org/>

sentence pool and removed the worker from the blacklist. Rashtchian et al. [49] and Chen et al. [50] used a similar prescreening strategy to approve crowdworkers; our strategy to validate sentences *during* the main task is unique to our work.

4.3.6 Experiments

We evaluated our algorithm on the task of image and video annotation. Due to the space limitations, we do not show the results for sections 4.3.3 and 4.3.4. First, we studied the influence of the parameters in our algorithm. Then, we compared our results with state-of-the-art algorithms on four standard datasets.

4.3.6.1 Experimental Settings

4.3.6.1.1 Datasets

We considered four publicly available datasets that have been widely used in previous work.

IXMAS This video dataset consists of 12 action classes (e.g., check watch, cross arms, and scratch head). Each action is executed three times by 12 actors and is recorded with five cameras observing the subjects from very different perspectives.

NUS-WIDE This image dataset contains 269,648 images downloaded from Flickr. Tagging ground truth for 81 semantic concepts is provided for evaluation. Similar to [51], we only use the images associated with the 10 most frequent concept annotations, obtaining 167,577 images in total.

For IXMAS, we use half of the tagged data to infer the tags for the rest of the data. For the other dataset, out-of-sample extension is utilized to annotate the test data points. We randomly selected 10,000 data points for training, and used the rest for testing. In the training dataset, partial data points are with tags.

4.3.6.1.2 Features

For IXMAS, we extracted the 500-D bag of words (BoW) feature based on SIFT for each camera, and the five cameras are taken as five features, following [52]. For NUS-WIDE, six types of low-level features are provided, including 64-D color histogram, 144-D color correlogram, 73-D edge direction histogram, 128-D wavelet texture, 225-D block-wise color moments, and 500-D BoW based on SIFT descriptions.

4.3.6.1.3 Baseline Methods and Evaluation Metrics

To evaluate the performance of the optimal graph, we compared OGL with different graph construction methods, e.g., LGC [31], LLE [53], and L2graph [54]. We further extended and compared all these methods on larger datasets by out-of-sample extension. We also compared them against TagProp [55] and fastTag [56] algorithms, which currently achieve the best performance on the benchmark datasets. To test different fitting models, we combined OGL with fastTag to check the performance. We also compared them with MBRM [57] and JEC [58].

We evaluated our models with standard performance measures:

Precision vs recall over each label is calculated by computing the precision at each recall.

Mean average precision (MAP) over labels is calculated by computing for each label the average of the precisions measured after each relevant image is retrieved.

4.3.6.2 Results

Figure 4.11 shows the comparisons for the two datasets. We also show some qualitative results in Figure 4.11c. From these figures, we have the following observations:

- By learning an optimal graph from partial tags and multiple features, the performance for image and video annotation is improved significantly compared with other graph construction methods. This indicates that using partial tags and multiple features can result in a better graph.
- The improvement of OGL over other methods is more significant for the IXMAS dataset than for the other datasets. This is probably because out-of-sample extension will reduce the performance gaps of different methods. L2Graph is a strong competitor in all cases, while LGC achieves the worst performance in all datasets. This indicates that a reconstruction coefficients-based graph is better than pairwise distance-based graphs in these datasets.
- Compared with other state-of-the-art non-graph-based methods, our OGL outperforms most of the existing approaches and achieves comparable performance with the current best result.

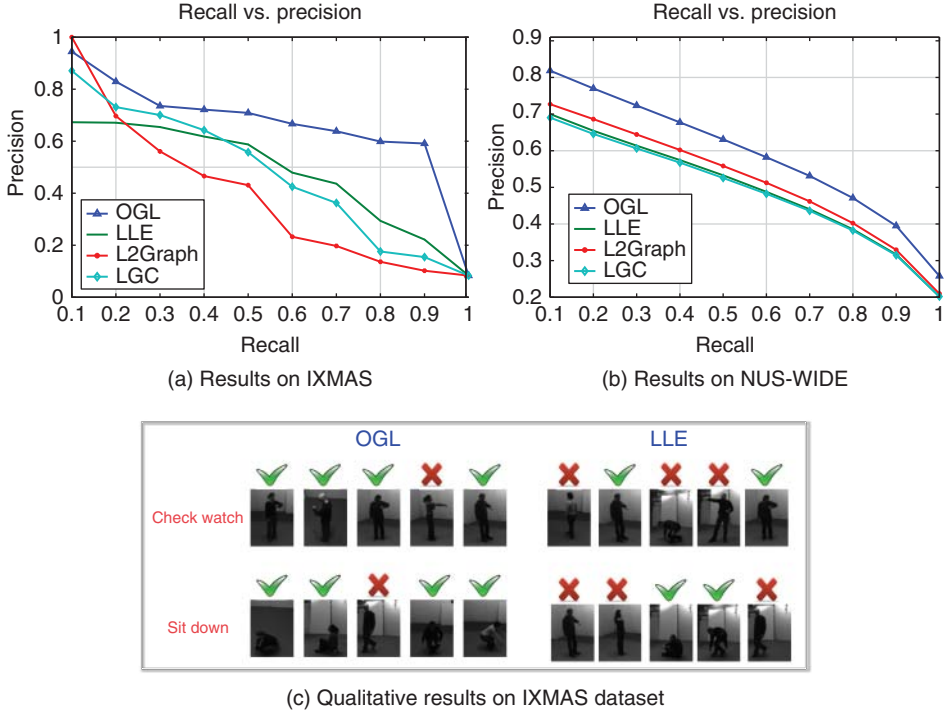


Figure 4.11 Experiment results on four datasets.

- From the qualitative results in Figure 4.11c, we can see that OGL can annotate the videos more precisely in the IXMAS dataset than LLE. This is because OGL makes use of both tags and multiple features, and automatically decides their weights, while LLE considers visual features only. In this case, OGL is more robust to some inaccurate visual features (e.g., light) or the tags, but LLE is more sensitive to these changes.

4.4 Conclusions and Future Work

In this work we cover video annotation, retrieval, and summarization, three approaches to fight the consequences of big video data. We present the state-of-the-art research in each of the research field. Specifically, we discuss both unsupervised and semi-supervised methods to facilitate video understanding tasks. We consider two general research problems: video retrieval and video annotation/summarization. While the former focuses on designing efficient algorithms for retrieving useful information from large-scale data, the latter aims to reduce the size of the data. Video retrieval provides new models and methods for effectively and efficiently searching through the huge variety of video data that are available in different kinds of repositories (digital libraries, Web portals, social networks, multimedia databases, etc.). On the other hand, video annotation/summarization generates annotations or a short summary of a video. It provides efficient storage and quick browsing of large collection of video data without losing important aspects. Depending on whether or not human labels are involved, video retrieval and annotation/summarization can be further categorized into unsupervised and semi-supervised based methods. In the future, we will consider applying our video understanding methods to more real-world applications, e.g., surveillance, video advertising, and short video recommendations.

References

- 1 Song, J., Yang, Y., Yang, Y., Huang, Z., and Shen, H.T. (2013) Inter-media hashing for large-scale retrieval from heterogeneous data sources, in *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data (SIGMOD '13)*, New York, USA, pp. 78–796, ACM, New York.
- 2 Song, J., Yang, Y., Li, X., Huang, Z., and Yang, Y. (2014) Robust hashing with local models for approximate similarity search. *IEEE Transactions on Cybernetics*, 44 (7), 1225–1236.
- 3 Wang, J., Zhang, T., Song, J., Sebe, N., and Shen, H.T. (2017) A survey on learning to hash. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 40 (4), 769–790.
- 4 Song, J., Gao, L., Liu, L., Zhu, X., and Sebe, N. (2017) Quantization-based hashing: a general framework for scalable image and video retrieval. *Pattern Recognition*, 75, 175–187.
- 5 Sidiropoulos, P., Vrochidis, S., and Kompatsiaris, I. (2011) Content-based binary image retrieval using the adaptive hierarchical density histogram. *Pattern Recognition*, 44 (4), 739–750.

- 6 Ye, G., Liu, D., Wang, J., and Chang, S.F. (2013) Large-scale video hashing via structure learning, in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2272–2279.
- 7 Song, J., Yang, Y., Huang, Z., Shen, H., and Hong, R. (2011) Multiple feature hashing for real-time large scale near-duplicate video retrieval, in *ACM Multimedia*, ACM, pp. 423–432.
- 8 Cao, L., Li, Z., Mu, Y., and Chang, S.F. (2012) Submodular video hashing: a unified framework towards video pooling and indexing, in *Proceedings of the 20th ACM International Conference on Multimedia (MM '12)*, Nara, Japan, pp. 299–308, ACM, New York
- 9 Girshick, R. (2015) Fast R-CNN, in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1440–1448.
- 10 You, Q., Jin, H., Wang, Z., Fang, C., and Luo, J. (2016) Image captioning with semantic attention, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4651–4659.
- 11 Liong, V.E., Lu, J., Tan, Y.P., and Zhou, J. (2016) Deep video hashing. *IEEE Transactions on Multimedia*, 19 (6), 1209–1219.
- 12 Venugopalan, S., Rohrbach, M., Donahue, J., Mooney, R., Darrell, T., and Saenko, K. (2015) Sequence to sequence-video to text, in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 4534–4542.
- 13 Pan, P., Xu, Z., Yang, Y., Wu, F., and Zhuang, Y. (2016) Hierarchical recurrent neural encoder for video representation with application to captioning, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1029–1038.
- 14 Gu, Y., Ma, C., and Yang, J. (2016) Supervised recurrent hashing for large scale video retrieval, in *Proceedings of the 2016 ACM on Multimedia Conference*, ACM, pp. 272–276.
- 15 Zhang, H., Wang, M., Hong, R., and Chua, T.S. (2016) Play and rewind: Optimizing binary representations of videos by self-supervised temporal hashing, in *Proceedings of the 2016 ACM on Multimedia Conference*, ACM, pp. 781–790.
- 16 Hochreiter, S. and Schmidhuber, J. (1997) Long short-term memory. *Neural Computation*, 9 (8), 1735–1780.
- 17 Song, J., Yang, Y., Huang, Z., Shen, H.T., and Luo, J. (2013) Effective multiple feature hashing for large-scale near-duplicate video retrieval. *IEEE Transactions on Multimedia*, 15 (8), 1997–2008.
- 18 Wu, X., Hauptmann, A.G., and Ngo, C.W. (2007) Practical elimination of near-duplicates from web video search, in *Proceedings of the 15th ACM International Conference on Multimedia (MM '07)*, Augsburg, Germany, pp. 218–227, ACM, New York.
- 19 Boureau, Y., Roux, N.L., Bach, F.R., Ponce, J., and LeCun, Y. (2011) Ask the locals: Multi-way local pooling for image recognition, in *Proceedings of the 2011 International Conference on Computer Vision (ICCV '11)*, pp. 2651–2658, IEEE Computer Society, Washington, DC.
- 20 Zhu, X., Ghahramani, Z., and Lafferty, J.D. (2003) Semi-supervised learning using Gaussian fields and harmonic functions, in *Proceedings of the 20th International Conference on International Conference on Machine Learning (ICML'03)*, Washington, DC, USA, pp. 912–919, AAAI Press.

- 21 Cc_web_video: Near-duplicate web video dataset, <http://vireo.cs.cityu.edu.hk/webvideo>.
- 22 Shang, L., Yang, L., Wang, F., Chan, K.P., and Hua, X.S. (2010) Real-time large scale near-duplicate web video retrieval, in *Proceedings of the 18th ACM International Conference on Multimedia (MM '10)*, Florence, Italy, pp. 531–540, ACM, New York.
- 23 Zhang, D., Wang, J., Cai, D., and Lu, J. (2010) Self-taught hashing for fast similarity search, in *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '10)*, Geneva, Switzerland, pp. 18–25, ACM, New York.
- 24 Weiss, Y., Torralba, A., and Fergus, R. (2008) *Spectral hashing*, in *Proceedings of the 21st International Conference on Neural Information Processing Systems (NIPS'08)*, Vancouver, British Columbia, Canada, pp. 1753–1760, Curran Associates Inc.
- 25 Jiang, Y.G., Ye, G., Chang, S.F., Ellis, D., and Loui, A.C. (2011) Consumer video understanding: A benchmark database and an evaluation of human and machine performance, in *Proceedings of the 1st ACM International Conference on Multimedia Retrieval (ICMR '11)*, Trento, Italy, pp. 29:1–29:8, ACM, New York.
- 26 Abu-El-Haija, S., Kothari, N., Lee, J., Natsev, P., Toderici, G., Varadarajan, B., and Vijayanarasimhan, S. (2016) Youtube-8m: A large-scale video classification benchmark. *Computing Research Repository (CoRR)*, [abs/1609.08675](https://arxiv.org/abs/1609.08675).
- 27 Yang, Y., Wang, Z., Yang, J., Wang, J., Chang, S., and Huang, T.S. (2014) Data clustering by Laplacian regularized l1-graph, in *Proceedings of the 28th AAAI Conference on Artificial Intelligence (AAAI '14)*, Quebec, Canada, pp. 3148–3149, AAAI Press.
- 28 Cheng, B., Yang, J., Yan, S., Fu, Y., and Huang, T.S. (2010) Learning with l1-graph for image analysis. *IEEE Transactions on Image Processing*, 19 (4), 858–866.
- 29 Deng, C., Ji, R., Tao, D., Gao, X., and Li, X. (2014) Weakly supervised multi-graph learning for robust image reranking. *IEEE Transactions on Multimedia*, 16 (3), 785–795.
- 30 Liu, W., Wang, J., and Chang, S.F. (2012) Robust and scalable graph-based semisupervised learning. *Proceedings of the IEEE*, 100 (9), 2624–2638.
- 31 Zhou, D., Bousquet, O., Lal, T.N., Weston, J., and Schölkopf, B. (2003) Learning with local and global consistency, in *Proceedings of the 16th International Conference on Neural Information Processing Systems, (NIPS '03)*, Whistler, British Columbia, Canada, pp. 321–328, MIT Press, Cambridge, MA, in *Proceedings of the 16th International Conference on Neural Information Processing Systems, (NIPS'03)*, Whistler, British Columbia, Canada, pp. 321–328, MIT Press, Cambridge, MA.
- 32 Belkin, M., Niyogi, P., and Sindhwani, V. (2006) Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of Machine Learning Research*, 7, 2399–2434.
- 33 Nie, F., Xu, D., Tsang, I.W.H., and Zhang, C. (2010) Flexible manifold embedding: A framework for semi-supervised and unsupervised dimension reduction. *TIP*, 19 (7), 1921–1932.
- 34 Tian, Q. and Chen, S. (2017) Cross-heterogeneous-database age estimation through correlation representation learning. *Neurocomputing*, 238, 286–295.
- 35 Pan, Z., Jin, P., Lei, J., Zhang, Y., Sun, X., and Kwong, S. (2016) Fast reference frame selection based on content similarity for low complexity HEVC encoder. *Visual Communication and Image Representation* 40, 516–524.

- 36 Song, J., Gao, L., Nie, F., Shen, H.T., Yan, Y., and Sebe, N. (2016) Optimized graph learning using partial tags and multiple features for image and video annotation. *Transactions Image Processing*, 25 (11), 4999–5011
- 37 Deng, T., Zhao, L., Wang, H., Liu, Q., and Feng, L. (2013) Refinder: A context-based information refinding system. *IEEE Transactions on Knowledge and Data Engineering*, 25 (9), 2119–2132, doi: 10.1109/TKDE.2012.157.
- 38 Wang, L., Zhao, X., Si, Y., Cao, L., and Liu, Y. (2017) Context-associative hierarchical memory model for human activity recognition and prediction. *IEEE Transactions on Multimedia*, 19 (3), 646–659.
- 39 Sperling, G. (1963) A model for visual memory tasks. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 5 (1), 19–31.
- 40 Gao, L., Wang, P., Song, J., Huang, Z., Shao, J., and Shen, H.T. (2017) Event video mashup: From hundreds of videos to minutes of skeleton, in *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, February 4–9, San Francisco, California, USA, pp. 1323–1330.
- 41 Li, Y., Song, Y., Cao, L., Tetreault, J.R., Goldberg, L., Jaimes, A., and Luo, J. (2016) TGIF: A new dataset and benchmark on animated GIF description, in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 27–30, Las Vegas, NV, USA, pp. 4641–4650.
- 42 Tran, D., Bourdev, L.D., Fergus, R., Torresani, L., and Paluri, M. (2015) Learning spatiotemporal features with 3D convolutional networks, in *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV '15)*, pp. 4489–4497, IEEE Computer Society, Washington, DC.
- 43 Neumann, L. and Matas, J. (2012) Real-time scene text localization and recognition, in *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR '12)*, pp. 3538–3545, IEEE Computer Society, Washington, DC.
- 44 Zauner, C. (2010) *Implementation and benchmarking of perceptual image hash functions*, Upper Austria University of Applied Sciences, Hagenberg Campus.
- 45 Norouzi, M., Punjani, A., and Fleet, D.J. (2014) Fast exact search in hamming space with multi-index hashing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36 (6), 1107–1119.
- 46 Toutanova, K. and Manning, C.D. (2010) Enriching the knowledge sources used in a maximum entropy part-of-speech tagger, in *Proceedings of the 2000 Joint SIG-DAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora: Held in Conjunction with the 38th Annual Meeting of the Association for Computational Linguistics (EMNLP '13)*, Volume 13, Hong Kong, pp. 63–70, Association for Computational Linguistics, Stroudsburg, PA.
- 47 Daiber, J., Jakob, M., Hokamp, C., and Mendes, P.N. (2013) Improving efficiency and accuracy in multilingual entity extraction, in *Proceedings of the 9th International Conference on Semantic Systems (I-SEMANTICS '13)*, Graz, Austria, pp. 121–124, ACM, New York.
- 48 Lavie, M.D.A. (2014) Meteor universal: Language specific translation evaluation for any target language, in *Proceedings of the Ninth Workshop on Statistical Machine Translation, Association for Computational Linguistics*, Baltimore, Maryland, USA, pp. 376–380,
- 49 Rashtchian, C., Young, P., Hodosh, M., and Hockenmaier, J. (2010) Collecting image annotations using Amazon's mechanical turk, in *Proceedings of the NAACL HLT*

- 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk (CSLDAMT '10), Los Angeles, California, pp. 139–147, Association for Computational Linguistics, Stroudsburg, PA.
- 50 Chen, D.L. and Dolan, W.B. (2011) Collecting highly parallel data for paraphrase evaluation, in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (HLT '11)*, Volume 1, Portland, Oregon, pp. 190–200, Association for Computational Linguistics, Stroudsburg, PA.
 - 51 Liu, W., Wang, J., Kumar, S., and Chang, S. (2011) Hashing with graphs, in *Proceedings of the 28th International Conference on International Conference on Machine Learning (ICML '11)*, Bellevue, Washington, USA, pp. 1–8, Omnipress.
 - 52 Yan, Y., Ricci, E., Subramanian, R., Liu, G., and Sebe, N. (2014) Multitask linear discriminant analysis for view invariant action recognition. *IEEE Transactions on Image Processing*, 23 (12), 5599–5611.
 - 53 Roweis, S. and Saul, L. (2000) Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290 (5500), 2323–2326.
 - 54 Peng, X., Zhang, L., and Yi, Z. (2012) Constructing l2-graph for subspace learning and segmentation. *Computing Research Repository*, abs/1209.0841.
 - 55 Guillaumin, M., Mensink, T., Verbeek, J.J., and Schmid, C. (2009) Tagprop: Discriminative metric learning in nearest neighbor models for image auto-annotation, in *2009 IEEE 12th International Conference on Computer Vision*, pp. 309–316, IEEE.
 - 56 Chen, M., Zheng, A., and Weinberger, K.Q. (2013) Fast image tagging, in *Proceedings of Machine Learning Research*, 28 (3), 1274–1282.
 - 57 Feng, S., Manmatha, R., and Lavrenko, V. (2004) Multiple Bernoulli relevance models for image and video annotation., in *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), Volume 2*, pp. II–II, IEEE.
 - 58 Makadia, A., Pavlovic, V., and Kumar, S. (2010) Baselines for image annotation. *International Journal of Computer Vision*, 90 (1), 88–105.

5

Multimodal Fusion of Big Multimedia Data

Ilias Gialampoukidis, Elisavet Chatzilari, Spiros Nikolopoulos, Stefanos Vrochidis and Ioannis Kompatsiaris

A plethora of multimedia data is available in the big data era, online or in personal collections, containing images, videos, textual metadata, and spatiotemporal information. Searching, clustering, classifying, and indexing such data is a challenging task, which can be attributed to two factors. First, multimedia data consist of highly heterogeneous information, which on one hand provides supplementary knowledge but on the other hand introduces the additional problem of homogenizing the multiple sources and fusing them into a unique representation. Second, the wide availability of multimedia content hinders the scalability of multimedia processing systems since they have to process very large streams of data.

Due to the diverse modalities that form a multimedia item (e.g., visual, textual, audio modality), multiple features are available to represent each modality. Textual and visual information usually complement each other when associated to the same multimedia item, such as a social media post. For example, there are multimedia retrieval tasks aiming to retrieve social media streams (e.g., from Flickr, Twitter, Wikipedia) that are relevant to flood events, using visual and textual information that complement each other in order to identify relevant posts, such as the Disaster Image Retrieval from Social Media (DIRSM) subtask of MediaEval2017¹.

To address these challenges, the fusion of multiple modalities may take place at the feature level (early fusion) or the decision level (late fusion). Typical early fusion techniques rely on the linear combination (weighted or not) of the multimodal features, while lately non-linear fusion approaches have prevailed. Another fusion tactic relies on graph-based techniques, allowing the construction of random walks, generalized diffusion processes, and cross-media transitions on the formulated graph of multimedia items. Moreover, canonical correlation analysis (CCA) and partial least squares (PLS) regression attempt to map the diverse features or similarities in a common latent space. In the late fusion category, the fusion happens at the decision level and can be based on (i) linear/non-linear combinations of the decisions from each modality, (ii) voting schemes (Borda, Condorcet, Reciprocal rank fusion, etc.), and (iii) rank diffusion processes.

1 <http://www.multimediaeval.org/mediaeval2017/multimediasatellite/>.

Scalability issues in multimedia processing systems typically occur for two reasons: (i) the lack of labeled data which limits the scalability with respect to the number of supported concepts and (ii) the high computational overload in terms of both processing time and memory complexity. For the first problem, methods that learn primarily on weakly labeled data (weakly supervised learning, semi-supervised learning) have been proposed. These methods require only minimum effort since they either rely on freely available annotations (e.g., user tags) or require only a small number of annotated samples. Active learning is a prominent case of semi-supervised methods. Its objective is to minimize the labeling effort by pinpointing the “useful-for-learning” content. This subset of the content is essentially selected so that the “knowledge” acquired by the learning method is maximized. In this way, annotations are required only for this part of the data rather than the whole set. For the second, methodologies typically rely on reducing the data space they work on by using smartly-selected subsets of the data, so that the computational requirements of the systems are optimized.

Towards meeting both challenges of big multimedia data (scalability and diversity), in this chapter we present state-of-the-art techniques in multimodal fusion of heterogeneous sources of data. In this direction, we explore both weakly supervised and semi-supervised approaches that minimize the complexity of the designed systems as well as maximizing their scalability potential to numerous concepts. Finally, we demonstrate the benefits of the presented methods in two wide domains: multimodal fusion in multimedia retrieval and in multimedia classification.

Multimedia retrieval is a challenging problem due to the large and diverse character of video collections (e.g., YouTube, Netflix) and annotated image collections (e.g., Facebook, Flickr). All modalities, usually textual and visual, need to be effectively fused, but in a scalable way. Often, multiple views appear even for the same modality, such as SIFT, DCNN, and Fisher vectors for the visual paradigm.

Additionally, multimedia classification requires a significant amount of labeled multimedia items (e.g., images associated with textual concepts) for the formulation of the training set. The problem of insufficient and incomplete knowledge, especially in large multimedia collections, is tackled by active learning techniques that are used to enrich the training set by fusing textual and visual information.

5.1 Multimodal Fusion in Multimedia Retrieval

Multimodal fusion of heterogeneous sources of information (images, videos, text, audio, and metadata) is an essential part of unsupervised multimedia retrieval, where the problem is to retrieve a ranked list of multimedia items from a multimedia collection \mathcal{M} that are relevant to a multimodal query q , having M modalities. The pairwise similarities between the query q and the items of the collection \mathcal{M} formulate m vectors ($m = 1, 2, \dots, M$) of similarity scores $s_m(q)$, one per modality. The notation followed in this section is presented in Table 5.1.

In the context of multimodal fusion, there are three basic strategies with respect to the level at which fusion is accomplished. The first strategy, called early fusion, performs fusion at the feature level (e.g., [1, 2]), where features from the considered modalities are combined into a common feature vector. The second strategy, known as late fusion, fuses information at the decision level, meaning that each modality is first learned separately and the individual results are aggregated into a final common decision (e.g., [3, 4]). As described in [5], among the advantages of early fusion is the fact that it can utilize

Table 5.1 Notations and definitions.

Notations	Definitions
\mathcal{M}	Multimedia collection
q	Multimodal query
$s(q)$	Fused similarity vector in response to the query q
S_m	Similarity (square) matrix for pairs of documents for the m^{th} modality
s_m	Query-based similarity vector for the m^{th} modality
$\mathbf{K}(\cdot, k)$	k^{th} -nearest-neighbor thresholding operator acting on a similarity vector
C	Multimodal contextual similarity matrix
P	Row stochastic transition probability matrix
$A \cdot B$	Matrix multiplication between the matrices A and B
$p_{\kappa\lambda}$	Transition probability between node κ and node λ
$c_{\kappa\lambda}$	The (κ, λ) element of the matrix C
$\alpha_m, \beta_m, \gamma$	Parameters in $[0,1]$
$x_{(\infty)}, y_{(\infty)}$	Steady state limiting distributions
$x_{(0)}, y_{(0)}$	Initial distribution
T_m	Matrices containing the extracted latent vectors
Q_m	Matrices representing the loadings
E_m	Error matrices

the correlation between multiple features from different modalities at an early stage, but it does not perform well in the case of an increase in the number of modalities because this makes it difficult to learn the cross-correlation among the heterogeneous features. On the other hand, late fusion is much more scalable (in terms of the modalities used in the fusion process) and flexible (it enables the use of the most suitable methods for analyzing each single modality) than early fusion, nevertheless its main drawback is its failure to utilize the feature-level correlation among modalities [5]. The third strategy, called hybrid fusion, aims to exploit the advantages of both early and late fusion (e.g., [6]).

5.1.1 Unsupervised Fusion in Multimedia Retrieval

The fusion of all modalities has been a major challenge in multimedia retrieval to exploit as much information as possible. In the following we present linear, non-linear, cross-modal, and graph-based models that have been used to fuse knowledge from multiple modalities.

5.1.1.1 Linear and Non-linear Similarity Fusion

The classic late fusion $s^w(q)$ of similarity vectors is a weighted linear combination of query-based similarity vectors s_m [5] per modality $m = 1, 2, \dots, M$:

$$s^w(q) = \sum_{m=1}^M \alpha_m s_m \quad (5.1)$$

where $\sum_{m=1}^M \alpha_m = 1$.

The non-linear analogue of Eq. 5.1 has also been considered in multimedia retrieval tasks [7]:

$$s^{nl}(q) = \sum_{m=1}^M (s_m)^{\alpha_m} \quad (5.2)$$

under the same restriction with as Eq. 5.1, i.e., all weights $\alpha_m, m = 1, 2, \dots, M$, must sum to one.

5.1.1.2 Cross-modal Fusion of Similarities

Cross-modal, or cross-media, fusion has been defined in the case of two modalities where the similarity vector of one modality s_1 is propagated to the similarities of the other modality:

$$s_{1 \rightarrow 2}^{cm}(q) = \mathbf{K}(s_2, k) \cdot S_1 \quad (5.3)$$

where $\mathbf{K}(\cdot, k)$ is the operator that takes as input a vector and gives zero value to elements whose score is strictly lower than the k th highest value. The output of $\mathbf{K}(s_2, k)$ is also a vector, multiplied by a similarity matrix of dimensions $l \times l$, to obtain the cross-modal similarity $s_{1 \rightarrow 2}^{cm}(q)$ that exploits knowledge from the other modality.

The linear combination of unimodal similarities with cross-modal similarities was introduced in [8, 9] and is given by:

$$s^{cm} = \alpha_1 s_1 + \alpha_2 s_2 + \alpha_3 s_{1 \rightarrow 2}^{cm} + \alpha_4 s_{2 \rightarrow 1}^{cm} \quad (5.4)$$

under the condition that $\sum_{i=1}^4 \alpha_i = 1$. In that case, the similarity vectors s_1, s_2 were obtained in response to the query q with respect to the textual and visual modality, respectively. In the general case of M modalities, the complexity increases dramatically, since $M(M-1)$ cross-modal similarities need to be linearly combined. To address this issue, a “semantic filtering” stage has been proposed [10] that filters out the non-relevant multimedia items with respect to a dominant modality, usually text-based (raw text or textual concepts). The semantic filtering stage, known also as top- l filtering, is demonstrated in Figure 5.1 for the first modality (Modality 1).

5.1.1.3 Random Walks and Graph-based Fusion

A graph is formulated by the top- l filtered multimedia items, being its nodes, and links are weighted by similarities from node κ to node λ . The pairwise similarity of the pair (κ, λ) for the m th modality is in the (κ, λ) element of the matrix S_m . A multimodal contextual similarity matrix C on the top- l filtered multimedia items is of size $l \times l$, and in the case of $M = 2$ modalities is defined as:

$$C = \beta_1 S_1 + \beta_2 S_2, \quad \beta_1 + \beta_2 = 1 \quad (5.5)$$

where $\beta_1, \beta_2 \in [0, 1]$. This similarity linearly combines all similarities $S_m(\kappa, \lambda)$ and is normalized to the row stochastic matrix P , such that all rows sum up to one. This is possible when defining a diagonal matrix D with elements the inverse of the row sums of C . This row stochastic matrix P has elements $p_{\kappa\lambda}$ which are transition probabilities from multimedia item κ to item λ . Since P is a stochastic transition probability matrix, the future evolution of a state vector $x_{(i)}$ of size $1 \times n$ is given by $x_{(i+1)} = x_{(i)} \cdot P$, having steady state distribution $x_{(\infty)} = \lim_{i \rightarrow \infty} x_{(i)}$ after many transitions (iterations). Only

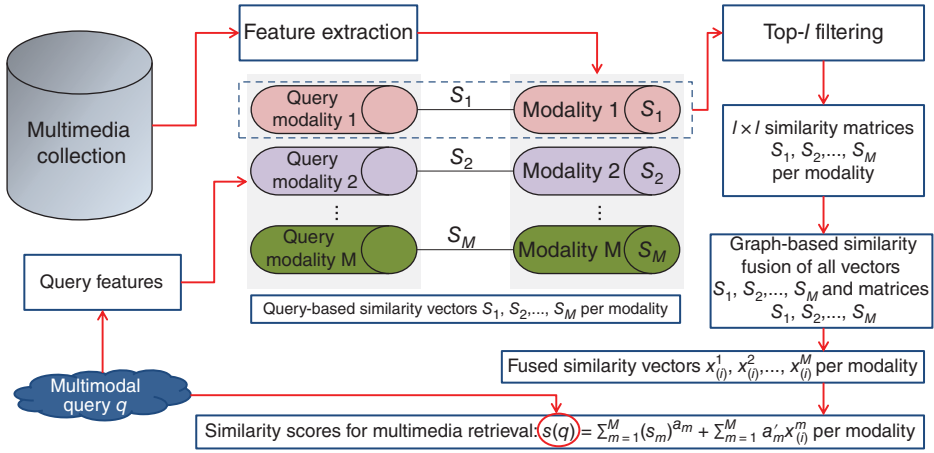


Figure 5.1 A multimodal fusion approach of M similarities, using a top- I filtering stage with respect to a dominant modality. The output of this fusion approach is a fused similarity vector $s(q)$ for all modalities, in response to the multimodal query q .

one transition ($i = 1$) is sufficient for the combination of two modalities [10] and many iterations introduces the noise of one modality to the other.

The graph-based approach has been proposed in [11] in the context of video retrieval. Furthermore, the addition of a “personalization” vector [12], such as the query-based similarity vector s_1 on the textual modality [11], introduces a perturbation towards the results of a text search:

$$x_{(i+1)} = (1 - \gamma)x_{(i)} \cdot P + \gamma s_1 \quad (5.6)$$

where $\gamma \in [0, 1]$ and after many iterations ($i \rightarrow \infty$):

$$x_{(\infty)} = (1 - \gamma)x_{(\infty)} \cdot P + \gamma s_1 \quad (5.7)$$

In addition, an image is also available as a part of a text-image query, so a similar graph-based process with a perturbation s_2 towards the results of an image search is:

$$y_{(i+1)} = (1 - \gamma)y_{(i)} \cdot P + \gamma s_2 \quad (5.8)$$

Therefore, a random walk of i iterations on a graph of multimedia items, linearly combined with the query-based similarity vectors s_1 and s_2 , provides the fused random-walk (similarity) score [10]:

$$s^{rw}(q) = \alpha_1 s_1 + \alpha_2 s_2 + \alpha_3 x_{(i)} + \alpha_4 y_{(i)} \quad (5.9)$$

under the restriction $\sum_{m=1}^4 \alpha_m = 1$.

In a more general context, Figure 5.1 presents the following unifying graph-based model in the case of M modalities, including the top- l filtering stage. The memory complexity is $\mathcal{O}(l^2)$ for the computation of each similarity matrix S_m , $m = 1, 2, \dots, M$, $\mathcal{O}(l)$ for each similarity vector $s_m(q, \cdot)$ and $\mathcal{O}(kl)$ for each x_m^I , $m = 1, 2, \dots, M$, thus the overall memory complexity is quadratic in l : $\mathcal{O}(Ml^2 + Mkl + Ml)$ but not in n (the total number of multimedia items). The optimal number of top- l filtered documents is selected

Table 5.2 Some special cases of the unifying unsupervised fusion model of Eq. 5.11

Fusion model	Equation	Conditions
Linear [5]	5.1	$\alpha_3 = \alpha_4 = 0$
Cross-modal [8]	5.4	$x_{(0)} = s_1, y_{(0)} = s_2, \beta_1 = 0, \gamma = 0, k < l, i = 1$
Random walk [11]	5.7	$x_{(0)}, y_{(0)}$ uniform, $k = l, i = \infty, \beta_1 > 0, \gamma > 0$,
Diffusion [10]	5.13	$x_{(0)} = s_1, y_{(0)} = s_2, \beta_1 > 0, \gamma > 0, k < l, i = \infty$

according to the following formula [13] to keep memory complexity at the level of two modalities:

$$l' = \sqrt{\frac{(k+1)^2}{4} + \frac{2l^2 + 2kl + 2l}{M}} - \frac{k+1}{2} \quad (5.10)$$

where $l = 1000$, $k = 10$, and M is the number of modalities. For example, in the case of highly heterogeneous big multimedia data with four, five, six or seven modalities (e.g., visual features, visual concepts, textual concepts, time, location, audio, user activity) the proposed number of l in the top- l filtered items is set to 704, 630, 575 or 531, respectively, to ensure that the memory cost of the graph-based fusion approach (Figure 5.1) is equivalent to the fusion of two modalities.

In the following, we present a unifying graph-based model for fusing two modalities and Table 5.2 presents the mapping of the unifying graph-based model to the linear, non-linear, cross-modal, and random-walk-based fusion, as well as the general diffusion process as special cases.

5.1.1.4 A Unifying Graph-based Model

A unifying graph-based model has been proposed [10], combining the aforementioned approaches in the case of two modalities:

$$\begin{aligned} x_{(i)} &\propto \mathbf{K}(x_{(i-1)}, k) \cdot [(1-\gamma)D \cdot (\beta_1 S_1 + \beta_2 S_2) + \gamma e \cdot s_1] \\ y_{(i)} &\propto \mathbf{K}(y_{(i-1)}, k) \cdot [(1-\gamma)D \cdot (\beta_1 S_2 + \beta_2 S_1) + \gamma e \cdot s_2] \end{aligned} \quad (5.11)$$

where e is the $l \times 1$ vector of ones, i is the number of iterations and the model sets: $x_{(0)} = s_1$ and $y_{(0)} = s_2$. The number $l < n$ is fixed, usually set to $l = 1000$, and is defined as the number of initially filtered multimedia items, with respect to the dominant modality (usually textual metadata). After the initial filtering stage, l items are left, so the similarity matrices S_1 and S_2 are $l \times l$. The matrices $\beta_1 S_1 + \beta_2 S_2$ and $\beta_1 S_2 + \beta_2 S_1$ are diagonalized when multiplied by a matrix D , which is diagonal and has diagonal elements the inverse of the row sums of $\beta_1 S_1 + \beta_2 S_2$ or $\beta_1 S_2 + \beta_2 S_1$, respectively. The final relevance score vector is given by:

$$s^{graph}(q) = \alpha_1 s_1 + \alpha_2 s_2 + \alpha_3 x_{(i)} + \alpha_4 y_{(i)} \quad (5.12)$$

under the restriction $\sum_{m=1}^4 \alpha_m = 1$ and $x_{(i)}, y_{(i)}$ are given by Eq. 5.11.

In the case of a large number of iterations ($i \rightarrow \infty$), the graph-based model of Eq. 5.12 becomes a (generalized) diffusion process [10]:

$$s^{dif}(q) = \alpha_1 s_1 + \alpha_2 s_2 + \alpha_3 x_{(\infty)} + \alpha_4 y_{(\infty)} \quad (5.13)$$

In Table 5.2 we present the linear fusion, the random walk fusion, the cross-modal similarities fusion, and a general diffusion process as special cases of the unifying framework of Eq. 5.11. We write $x_{(0)}, y_{(0)}$ uniform to state that the initial distribution is uniform, i.e., all elements are equal and they sum to one, for both $x_{(0)}$ and $y_{(0)}$.

Graph-based methods [10] incorporate cross-modal similarities and random-walk based scores. Specifically, the random-walk approach for multimodal fusion was introduced in [11], where the fusion of textual and visual information leads to improved performance in the video search task. The framework in [10] includes as special cases all well-known fusion models (e.g., early, late, diffusion-based, etc.) and does not require users' relevance feedback. In the context of multimedia retrieval it has been shown [13] that the addition of multiple modalities (more than two) can increase the performance of a multimedia search task, without necessarily increasing complexity through the exploitation of Eq. 5.10. The model of Eq. 5.11 has been extended to multiple modalities with a non-linear fusion of all similarities [14], as shown in Figure 5.1.

5.1.2 Partial Least Squares Regression

Similarity matrices can be fused using partial least squares (PLS) approaches that map two modalities to a common latent space. The aim of PLS is to maximize the covariance between T_1 and T_2 below. PLS regression leads to models that are able to fit the response variable with fewer components than principal components regression (PCR) and moreover takes into account the response variable, contrary to the PCR model. Without the use of a top- l filtering stage, as shown in graph-based models (Figure 5.1), the PLS approach is not a scalable solution and therefore is not applicable to big multimedia data. In the following, we assume that a dominant modality has been used to filter out non-relevant items and the similarity matrices S_1 and S_2 are reduced to the dimension of $l \times l$.

The PLS model has been used in the fusion of multimedia data [15] to efficiently combine two modalities. Given two similarity matrices S_1, S_2 , PLS decomposes S_1 and S_2 as follows:

$$\begin{aligned} S_1 &= T_1 \cdot Q_1^T + E_1 \\ S_2 &= T_2 \cdot Q_2^T + E_2 \end{aligned} \quad (5.14)$$

where T_1 and T_2 are projections of S_1 and S_2 , respectively, to latent spaces containing the extracted latent vectors. Q_1 and Q_2 are orthogonal "loading" matrices and, finally, E_1 and E_2 are error matrices. The NIPALS² algorithm initially chooses u_1 as one column of S_2 and iteratively constructs (normalized) projection vectors:

$$w_1 = \frac{S_1^T u_1}{\|S_1^T u_1\|} \rightarrow t_1 = S_1 w_1 \rightarrow q_1 = \frac{u_1^T t_1}{\|u_1^T t_1\|} \rightarrow u_1 = S_2 q_1 \rightarrow \pi_1 = \frac{S_1^T t_1}{\|t_1^T t_1\|} \quad (5.15)$$

The regression coefficient for the first stage is defined as $b_1 = u_1^T t_1 (t_1^T t_1)^{-1}$. After calculating scores and loadings for the first latent variable, the S_1 and S_2 block residuals are calculated:

$$\begin{aligned} E_1 &= S_1 - t_1 \pi_1^T \\ E_2 &= S_2 - u_1 q_1^T \end{aligned} \quad (5.16)$$

2 http://www.eigenvector.com/Docs/Wise_pls_properties.pdf.

The entire procedure is repeated by replacing S_1 and S_2 with their residuals to get the fused similarity S_{fused} defined as:

$$S_{fused} = \sum_i t_i b_i, \quad b_i = u_i^T t_i (t_i^T t_i)^{-1} \quad (5.17)$$

The fused similarity matrix S_{fused} is one unique similarity matrix for both modalities and can be used for retrieval, clustering or classification purposes to compare any two multimodal objects.

PLS and correlation matching have been widely used as multimodal fusion techniques for multimedia data. A PLS-based framework, which maps queries from multiple modalities to points in a common linear subspace, has been introduced in [15]. Correlation matching [16] is utilized between the textual and visual modalities of multimedia documents in the task of cross-modal document retrieval. Some similar multimedia and cross-modal retrieval studies have focused on latent Dirichlet allocation (LDA). In [17], a supervised multimodal mutual topic reinforce modeling approach for cross-modal retrieval, called M3R, has been proposed. As an extension of [14], PLS has also been used to consider multiple views per modality (SIFT, DCNN features) by fusing them using PLS regression [18].

5.1.3 Experimental Comparison

The unsupervised multimedia retrieval approaches are compared in this section.

5.1.3.1 Dataset Description

The experimental comparison among the presented fusion approaches is done in three datasets, namely the WIKI11 [19], the IAPR-TC12 [20], and the MediaEval dataset from the diversity task of 2015 [21]. A title and a short description are provided for each image of both WIKI11 and IAPR-TC12 datasets, thus formulating the textual modality. For each topic, the datasets provide up to five exemplary images and in our experiments we select only one image as a query example and in particular the image which has the most detected visual concepts. Low- and high-level information has been extracted from all datasets:

- *Visual descriptors*: The scale-invariant local descriptors RGB-SIFT [22] are extracted and are then locally aggregated into one vector representation (4000-dimensional) using VLAD encoding [23]. In addition, deep convolutional neural networks (DCNNs) are utilized for the extraction of DCNN descriptors [24]. More specifically, these DCNN descriptors are based on the 16-layer pretrained DCNN [25], which has been trained on ImageNet data. The output (a 4096-element vector) of the last hidden layer of this network provides the DCNN descriptors that are used in this work.
- *Visual concepts*: The images of the multimedia objects are indexed by 346 high-level concepts (TRECVID), which are detected by multiple independent concept detectors. The locally aggregated features (VLAD encoding for RGB-SIFT descriptors) serve as input to logistic regression classifiers and their output is averaged and further refined. Similarly, the extracted DCNN visual descriptors [24] provide DCNN-based visual concepts, in addition to the ones provided by RGB-SIFT.

- *Textual concepts*: The textual concepts used in the evaluation of the multimedia retrieval task are extracted using the DBpedia Spotlight annotation tool, which is an open source project for automatic annotation of DBpedia entities in natural language text [26].

5.1.3.2 Settings

Term frequency-inverse document frequency (TF-IDF) scores are compared using the cosine similarity and the similarities of the visual features are calculated as [27]:

$$S_{\kappa\lambda} = 1 - \frac{d_{\kappa\lambda}}{\max_{\lambda} d_{\kappa\lambda}} \quad (5.18)$$

where $d_{\kappa\lambda}$ is the Euclidean distance between item κ and item λ . The initial filtering stage involves Lucene's³ text search, where we keep the top-1000 similar-to-the-query documents, i.e., $l = 1000$. One iteration ($i = 1$) is used as in [10], since more than one ($i > 1$) iterations adds the noise from one modality to the others. The parameter k in the operator \mathbf{K} is set to $k = 10$ and we used the Python implementation⁴ of PLS regression. Additional details may be found in [18].

5.1.3.3 Results

We compared the reported methods using mean average precision (MAP) and the precision at the top-20 retrieved results (P@20) as two of the best-established measures in information retrieval tasks. First, the majority vote over all modalities (rule-based fusion) determines the modality with the highest performance [28]. Second, we used the cross-media fusion [8] of three modalities and finally the random-walk approach of [11]. The fourth baseline method is the non-linear fusion [7] of all modalities and we compared our framework with the extension of the unifying fusion framework of [10] in the case of three modalities [13] in two cases: first with the SIFT visual descriptors and second with the state-of-the-art DCNN visual features. Finally, another framework is presented [18] which first combines SIFT with DCNN using PLS regression and then uses non-linear graph-based fusion for all three modalities.

Experiments are performed for parameters $\alpha_m, \alpha'_m, \beta_m, \gamma_m, m = 1, 2, 3$ values ranging from 0.0 to 1.0 at step 0.25. Our goal was to gauge the effectiveness of the proposed multimedia retrieval framework across a wide range of parameter values rather than finding the optimal ones. First, the parameters $\alpha_m, \alpha'_m, \beta_m, m = 1, 2, 3$ were kept constant (all equal to $\frac{1}{3}$) while we examined the effect of the parameters $\gamma_1, \gamma_2, \gamma_3$ ($\gamma_3 = 1 - \gamma_1 - \gamma_2$) in the retrieval effectiveness. Afterwards, the results of this investigation indicated that the role of each modality is very important, while the best results are obtained for the following parameters values: $\gamma_1, \gamma_2, \gamma_3$ is (1,0,0) for the WIKI dataset, (0.25,0,0.75) for the IAPR-TC12 dataset, and (0.5,0,0.5) for the MediaEval dataset. Modifying the parameters $\beta_m, m = 1, 2, 3$, while the other parameters are kept constant did not lead to significant changes in the performance. Varying values of the parameters α_m, α'_m results in many well-known fusion techniques appearing as special cases. For example, if $\alpha_1 = 1, \alpha_2 = 0, \alpha_3 = 0, \alpha'_m = 0$, only one modality is involved (text). Similarly, if $\alpha_1 = \alpha_2 = \alpha_3 = 0$ and all other $\alpha'_m, m = 1, 2, 3$ are tuned, then we get a random-walk

³ <https://lucene.apache.org/>.

⁴ http://scikit-learn.org/stable/modules/generated/sklearn.cross_decomposition.PLSRegression.html.

Table 5.3 Evaluation results under MAP and P@20 [18].

Method	WIKI11		IAPR-TC12		MediaEval	
	MAP	P@20	MAP	P@20	MAP	P@20
Majority vote (best modality)	0.3325	0.1630	0.2385	0.2050	0.3154	0.3760
Cross-modal fusion	0.3325	0.1630	0.2392	0.2084	0.3144	0.3764
Random walk based fusion	0.3344	0.1640	0.2401	0.2083	0.3218	0.4020
Non-linear fusion	0.3341	0.1730	0.2418	0.2200	0.3220	0.4020
Graph-based fusion (SIFT)	0.3341	0.1730	0.2418	0.2200	0.3214	0.4028
Graph-based fusion (DCNN)	0.3958	0.2003	0.2716	0.2400	0.3646	0.4439
Graph-based fusion with PLS	0.4013	0.2040	0.2771	0.2417	0.3667	0.4581

based fusion. The variation of all parameters has shown that all three modalities need to be fused to get the highest possible performance.

The best results across all methods are presented in Table 5.3. The best performance appears for two components in the PLS regression for the WIKI11 and the MediaEval dataset, while a peak appears in the performance for three components in the IAPR-TC12 dataset. The best parameter selections are $\alpha_1 = 0.5, \alpha_2 = 0.0, \alpha'_1 = 0, \alpha'_2 = 0.25$ for the WIKI11 dataset, $\alpha_1 = 0.5, \alpha_2 = 0.25, \alpha'_1 = 0.5, \alpha'_2 = 0.0$ for the IAPR-TC12 dataset, and $\alpha_1 = \alpha_2 = \alpha'_1 = \alpha'_2 = 0.0$ for the MediaEval dataset.

PLS regression is used to map SIFT and DCNN features into a common latent space, building a joint block of features (visual descriptors and visual concepts), which is fused with textual concepts. The performance is better when PLS regression is added as an initial fusion stage, as shown in the experiments (Table 5.3).

The fusion of all similarities per modality has at least quadratic complexity and, therefore, it is not a scalable solution. To that end the top- l filtering stage filters out non-relevant items with respect to the textual modality or any dominant modality in general.

One limitation of the proposed method is the number of parameters that need to be tuned when switching from one collection to another. Several approaches towards the model simplification will be considered as one main future challenge. Moreover, the presented multimedia retrieval framework does not use canonical correlation analysis, but PLS regression to map two different visual descriptors into one common latent space, a process where competitive swarm optimization (CSO) would also serve as a scalable but effective solution [29].

5.1.4 Late Fusion of Multiple Multimedia Rankings

Multimedia retrieval using multiple image queries has also been tackled using several alternative methods in [30] and has been qualitatively evaluated in the TRECVID 2011 Known Item Search task, comparing the average query method (Joint-Avg), where the bag-of-words (BoW) of all images (tf-idf scores) are averaged together, against the Joint-SVM to retrieve shots from compound queries, where a linear support vector machine (SVM) is used to learn a weight vector for visual words online at the feature

level. The query set has the positive instances and a random set from the collection provides the negative instances. Visual words are weighted using an SVM-based supervisory layer, but one limitation is that the random choice of the negative class may affect the final results, since positive instances may also be included.

The fusion of multiple multimedia ranked lists of items has been a challenging problem [31]. Earlier works in the combination of multiple retrieval tasks involve late fusion techniques on the multiple rankings. CombSUM is a method for combining several ranked lists of retrieved results that originally appeared in [32], where the sum of all similarity values outputs one unique list of similarities. Alternative to the sum of individual similarities, the minimum, the maximum, the median, and other combinations of retrieved lists have been proposed, namely CombMIN, CombMAX, CombMED, CombANZ, and CombMNZ [32]. Some of these methods have appeared later as MQ-Max and MQ-Avg [30], where the maximum (or average) of individual scores is obtained from each visual query in image and video retrieval. Another late fusion approach is the Condorcet fuse [33], which has been outperformed by reciprocal rank fusion [34]. Furthermore, Borda-count fuses rankings in a simple and scalable way [35] by assigning points to each item based on its order per lists and points are summed for all rankings. Variations of these late fusion methods have been used in the context of video shot retrieval [36], where the authors conclude that features should first be fused separately for the visual examples and then these features' scores should be fused by adding the normalized relevance scores of the top searched results.

In this section we compare late fusion approaches in the context of video retrieval, where compound queries are used, which may also have diverse content. Temporal information is also exploited to re-rank the final list of retrieved results. Late fusion of multiple rankings can be at the score level or the rank level, taking into account only the order of each retrieved video shot.

5.1.4.1 Score Fusion

An initial attempt to combine multiple rankings has been made at the score level. CombSUM combines the relevance scores $s(q_u)$, $u = 1, 2, \dots, U$, summing up for all sub-queries q_u , $u = 1, 2, \dots, U$:

$$CombSUM(shot) = \sum_{u=1}^U s(q_u) \quad (5.19)$$

CombMNZ uses the individual result obtained by CombSUM multiplied by the number of results which have non-zero scores $\zeta(shot) = |\{u : s(q_u, doc) > 0\}|$. CombMAX uses the maximum and scores among all rankings:

$$CombMAX(shot) = \max_u \{s(q_u, shot)\} \quad (5.20)$$

Normalization on the aforementioned score-based approaches, using only the scores of the top- L retrieved results, have shown better performance [36] when compared with CombSUM and CombMAX. The normalization is:

$$norm_{score}(shot) = \frac{s(q_u, shot) - \min_{rank(shot) < L} s(q_u, shot)}{\max_{rank(shot) < L} s(q_u, shot) - \min_{rank(shot) < L} s(q_u, shot)} \quad (5.21)$$

The normalization of Eq. 5.21 provides scores in $[0,1]$ and apart from summation or taking their maximum, multiplication is also possible to get the joint probabilities.

Recently a weighted non-linear score-based fusion method [37] combined visual features (SIFT, HSC, CNN, GIST), which are l_2 -normalized, to improve the performance of image retrieval:

$$\text{sim}(\text{shot}) = \prod_{u=1}^U s(q_u)^{\eta_u} \quad (5.22)$$

where $\eta_u, u = 1, 2, \dots, U$, is the weight of the u th list of results, such as:

$$\sum_{u=1}^U \eta_u = 1 \quad (5.23)$$

5.1.4.2 Rank Fusion

The fusion of multiple retrieval results may also be done using the rankings (orders) of the retrieved results. Three representative methods are Borda count fusion, reciprocal rank fusion, and Condorcet fusion. In addition, rank diffusion processes have been proposed [38, 39] but they require an initial filtering stage, such as the one presented in Eq. 5.1.

5.1.4.2.1 Borda Count Fusion

In Borda count fusion [35] the top- L elements of each ranking are kept. The first item takes L points, the second takes $L - 1$, and the same process is repeated for each ranked list. The sum of all points for each item provides the final score for the fused ranked list.

5.1.4.2.2 Reciprocal Rank Fusion

Given a set of n video shots and U ranked lists, the reciprocal rank fusion score of each *shot* is [34]:

$$\text{RRF}(\text{shot}) = \sum_{u=1}^U \frac{1}{\kappa + r^{\text{nl-graph}}(q_u, \text{shot})} \quad (5.24)$$

where $r^{\text{nl-graph}}(q_u, \text{doc})$ is the order of *shot* at ranking u and κ is fixed with default value $\kappa = 60$ [34]. Video shots with small order $r^{\text{nl-graph}}(q_u, \text{shot})$ for many lists tend to increase the fraction of Eq. 5.24 and get a high reciprocal rank fusion score.

5.1.4.2.3 Condorcet Fusion

This combines rankings by sorting the documents according to the pairwise relation $r^{\text{nl-graph}}(q_u, \text{shot}_1) < r^{\text{nl-graph}}(q_u, \text{shot}_2)$, which is determined for each pair of documents $(\text{shot}_1, \text{shot}_2)$ by majority vote among all rankings.

Apart from the score-based and rank-based deterministic approaches for fusing multiple modalities, another probabilistic late fusion approach has been used in the context of multimedia classification. In the following, we present the active learning state-of-the-art approach and its comparison with other fusion approaches.

5.2 Multimodal Fusion in Multimedia Classification

It is commonly accepted that classification models become more robust when generated by high volumes of noise-free and diverse training data. However, the requirement

of manual annotation hinders their application in large-scale classification problems. In an effort to minimize the labeling effort, active learning [40] trains an initial classifier with a very small set of labeled data and expands the training set by selectively sampling new examples from a much larger set of unlabeled data (also known as the pool of candidates). The additional data samples are selected based on their *informativeness*, i.e., how much they are expected to improve the classifier's performance if their label was known. They are found in the areas in the feature-space with high uncertainty for the classifier and, in the typical case, are annotated upon request by an errorless oracle (i.e., human annotator).

However, in the domain of image classification, large amounts of user tagged images have been made available at almost no cost due to the widespread use of online social networks. These images, besides their mere visual content, offer additional textual information, popularly appearing in the form of tags. If we could leverage these tags to pinpoint the images' actual content, we could potentially remove the need for a human annotator and automate the whole active learning process. However, in this case, where the labels are extracted from the freely available user tags, actively selecting new samples might seem superfluous, since there is no labeling cost to minimize. Indeed, we could simply just add all the images in the pool instead of actively selecting new ones. The question we should pose though is, *how many images will we need to reach the same performance?* The computational overload (i.e., storage, memory, and processing time) of dealing with training sets several times larger might not seem important in the scale of a few concepts and a few thousand images. However, as we have witnessed in the past years, large-scale image benchmarks gain an order of scale almost every year (from the ~10k images of the first Pascal-VOC competition at 2007 [41], to the ~200k images of NUS-WIDE [42], the 1 million images of MIRFLICKR [43], to the 14 million images of ImageNet [44] and the 100 million images of Yahoo [45] nowadays). The situation is similar for the number of concepts, although the growth is not as steep as in the case of images (approximately from 20 concepts at 2007 to 20000 nowadays). By incorporating active learning in the selection process, a smaller part of the pool dataset, which is deemed as informative, is required to be included in the training set. In this way, we only need to add samples that are expected to provide new information to the classifier and boost significantly its performance. This scales down the computational complexity of the proposed method, since it drastically reduces the size of the training set. Besides the gain in computational complexity, active learning reduces the required *annotations* (using the tags) only for the small *informative* part of the pool dataset, which saves us from possible mistakes in the annotation process due to the noisy nature of tagging information.

However, even though tags can be obtained freely, they cannot be considered as accurate labels. While simple string matching could be used to annotate the images, it has been proven that the accuracy of the tags is rather low [46]. Thus, a more sophisticated way is required to define a measurement that extracts the true contents of the images based on their tagging information. Essentially, if we consider the web users as oracles, this measurement can be considered as the *oracle's confidence*. This confidence indicates the probability that an image depicts a specific concept/object. The novelty of the proposed approach is a sample selection strategy that maximizes not only the informativeness of the selected samples (based on visual information), which is the typical process in active learning, but also the oracle's confidence about their label (based on

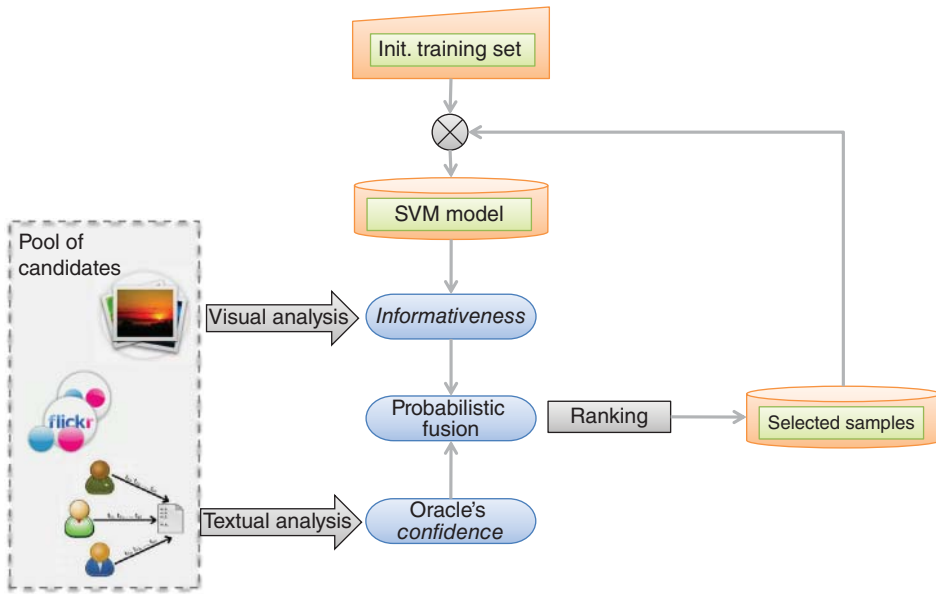


Figure 5.2 Social Active Learning for Image Classification (SALIC) schematic presentation. An initial classifier is trained on a small set of labeled images. This classifier is used to identify informative images from a user tagged image database. The users are the oracles tagging the images and their tags are used to extract a confidence about the true image contents. The system ranks the images based on jointly how informative they are and how confident we are about their true contents. Finally the top-ranked images are included in the training set. This procedure is repeated iteratively.

textual information), by fusing the aforementioned modalities. In order to achieve this goal we formulate the fusion process as a joint optimization problem that maximizes a function conditioned on the samples' informativeness and the oracle's confidence. In order to estimate this joint function, we propose to approximate it with a joint conditional probabilistic function. Towards quantifying this probability, we approximate the samples' informativeness based on their distance to the separating hyperplane of the classification model, which is known to be an effective method for finding informative samples [47]. In order to measure the oracle's confidence, we propose the utilization of the popular BoW approach [48] due to its ability to capture important contextual information and, in this way, reduce the effect of erroneously provided, ambiguous, and misleading tags. This probability indicates the benefit that our system is expected to gain if the examined sample is selected and added to the training set. Finally, the samples are ranked based on the joint probability. The top ranked samples are eventually selected to be added to the training set (see Figure 5.2).

5.2.1 Related Literature

During the past decade, various active learning approaches have been developed using different sample selection strategies [47, 49–52]. The common underlying idea behind the proposed strategies was how to best approximate the gain achieved by a classifier if a sample is labeled and added to the training set. This work considered an error-free

human oracle that would accurately annotate the selected samples. Furthermore, there have been quite a few studies that have investigated active learning with noisy oracles [53–55]. The objective of the work dealing with noisy oracles is to model the expertise of each oracle, so that the most reliable for a specific instance can be selected. However, all these works consider that there is at least one oracle that holds the truth, compared to the proposed approach where the oracle may never be absolutely confident for its decision. In addition, the sample and oracle selection is performed in two steps (i.e., first select the most informative sample and then the most reliable oracle for the selected sample). This is in contrast to the presented work, where we *jointly* select the sample that is both informative and which our oracle can reliably annotate.

The idea of combining the benefits of active learning and the knowledge generated from the crowds has recently become the focus of research [56, 57]. In this line of work, the oracle is substituted by workers in popular crowdsourcing platforms (e.g., MTurk). However, although annotations originating from crowdsourcing platforms are closer to an expert's annotation in terms of labelling accuracy [58], they cannot be considered either fully automated or free. In contrast, SALIC relies on data originating from on-line social networks (Flickr images and tags) that, although noisier can be used to support a fully automatic learning framework. To utilize the free web content, another popular direction is weakly supervised learning techniques [46, 59]. In [59], the authors try to optimize the data-gathering method relying on the results of image search engines, based on the assumption that search engines tend to return very relevant images in the first results of a query. More specifically, the proposed method creates a set of queries, which is given to image search engines and the top images returned by each engine for each query are kept as positive examples. The set of queries is formulated by translating the original concept to 15 different languages, using hyponyms, hypernyms, and synonyms from WordNet and finding related terms within the results of the Google text search engine. Utilizing user tagged images in a deep learning scenario, the authors of [46] propose a noise-resistant version of logistic regression that can learn model parameters for any tag. The proposed method, based on stochastic EM, can learn from large volumes of user tagged images. In principle, SALIC falls under the weakly supervised learning algorithms umbrella, since it only uses tagged examples and no human annotator. However, the novelty of this work comes from the joint consideration of the samples' informativeness (active learning) and the oracle's confidence (tag relevance). In this way, the visual content of the user tagged images, apart from being used for training the classifiers as in the typical weakly supervised learning case, additionally defines which samples should be added in the training set in the first place. The advantage of integrating active learning in the selection process is that we can weed out the non-useful part of the dataset. This counters the inclusion of unhelpful and possibly falsely labeled data to the model, which would increase the computational complexity and diminish the performance of the model.

More closely related to the proposed approach in this section is the work presented in [60], which is examined in the same context as this work (i.e., active learning in the multimedia domain with user tagged images). The approach presented in [60] is based on the assumption that tags can reliably determine if an image does not include a concept, thus making social sites a reliable pool of negative examples. The selected negative samples are further sampled by a two-stage sampling strategy. First, a subset is randomly selected and then the initial classifier is applied to the remaining negative

samples. The examples that are most misclassified (i.e., the examples that received the highest confidence scores from the classification model) are considered as the most informative negatives and are selected to boost the classifier. Compared to this work, which also incorporates active learning in its selection process, the novelty of SALIC lies in the proposed probabilistic fusion strategy, unlike the approach in [60], which relies on a two-step methodology (i.e., using one modality to filter part of the pool dataset and then applying the criterion of the other modality to perform sample selection).

5.2.2 Problem Formulation

In order to facilitate the description of probabilistic fusion, we use the notation in Table 5.4.

Let us consider the typical active learning scenario where we have an initial training set of manually labeled instances $\mathfrak{L} = \{l_1, l_2, \dots, l_{N_{\mathfrak{L}}}\}$, $N_{\mathfrak{L}} = ||\mathfrak{L}||$ accompanied by their corresponding labels $Y = \{y_1, y_2, \dots, y_{N_{\mathfrak{L}}}\}$, $y_i \in \mathbb{D}$, where \mathbb{D} is the label space. In our case, we consider the binary classification problem (i.e., $\mathbb{D} = \{+1, -1\}$), thus the probable labels are positive (i.e., $d_1 = +1$) and negative (i.e., $d_2 = -1$), with respect to the concept that we are trying to learn. In addition to the labeled set, we have a large set of unlabeled instances $\mathfrak{U} = \{u_1, u_2, \dots, u_{N_{\mathfrak{U}}}\}$, $N_{\mathfrak{U}} = ||\mathfrak{U}||$. Moreover, there is an oracle \mathcal{O} providing labels accurately for the unlabeled set \mathfrak{U} on demand (i.e., $y_i = \mathcal{O}(u_i)$). Initially, a baseline classifier \mathcal{H}_0 is trained on the labeled set \mathfrak{L} . The objective of active learning is to establish a function $\mathfrak{E}(\mathcal{H}_0, \mathfrak{U})$, which defines the informativeness of each instance in the unlabeled

Table 5.4 Notation table.

Symbol	Definition
$\mathfrak{L} = \{l_1, l_2, \dots, l_{N_{\mathfrak{L}}}\}$	The set of labeled images
$Y = \{y_1, y_2, \dots, y_{N_{\mathfrak{L}}}\}$	The labels of \mathfrak{L}
$\mathbb{D} = \{+1, -1\}$	The label space
$\mathfrak{U} = \{u_1, u_2, \dots, u_{N_{\mathfrak{U}}}\}$	The set of unlabeled images
$\mathcal{T} = \{\mathbf{t}_1, \mathbf{t}_2, \dots, \mathbf{t}_{N_{\mathfrak{U}}}\}$	The tags of \mathfrak{U} , where \mathbf{t}_i is the set of tags that correspond to image u_i
\mathbf{u}_i	The visual descriptor of the image u_i
$\mathbf{u}_i^{\text{text}}$	The textual descriptor of the image u_i
$\mathcal{H}_m = \{\mathbf{w}_m, b_m\}$	The classifier of iteration m (\mathbf{w}_m is the normal vector to the SVM hyperplane and b_m is the bias term). \mathcal{H}_0 is the baseline classifier, trained with the initially labelled data
$\mathcal{E}(\mathcal{H}_m, u)$	The informativeness of sample u based on classifier \mathcal{H}_m (i.e., the classifier at iteration m)
$\mathcal{O}(u, \mathbf{t}, d_k)$	The confidence of the oracle that the sample u belongs to the label d_k based on its tags \mathbf{t}
$S_i \in \{0, 1\}$	The random variable (RV) modeling the event of selecting the image u_i
$V_i \in [0, 1]$	The RV modeling the probability that u_i is informative
$T_i \in [0, 1]$	The RV modeling the probability that u_i belongs to the examined concept $d_k \in \mathbb{D}$

*We use normal letters (e.g. u_i) to indicate individuals of some population and bold face letters (e.g. \mathbf{u}_i) to indicate vectors or sets of individuals of the same population.

set based on the previous classifier \mathcal{H}_0 . The instance $u^* \in \mathcal{U}$ maximizing this function is selected to be added in the labeled set \mathcal{L} along with its label y^* , which is provided by the oracle \mathcal{O} (i.e., $y^* = \mathcal{O}(u^*)$).

In the proposed approach, the oracle is substituted with the web users and the set of images \mathcal{U} is not completely unlabeled but is associated with a set of tags $\mathcal{T} = \{\mathbf{t}_1, \mathbf{t}_2, \dots, \mathbf{t}_{N_{\mathcal{U}}}\}$, where \mathbf{t}_i is the set of tags associated with image u_i by the web user oracle. In this sense, the oracle has already answered for all the instances in \mathcal{U} beforehand (i.e., the web users have already tagged the images). However, given the noisy nature of the user tags, we consider the oracle to be of questionable reliability and instead of providing an accurate label for a specific sample u^* as before, it provides its confidence for each sample u_i and label $d_k \in \{+1, -1\}$ (i.e., its confidence $\mathcal{O}(u_i, \mathbf{t}_i, d_k)$ that the sample u_i is positive if $d_k = +1$ and negative if $d_k = -1$) based on the tags \mathbf{t}_i of the image u_i . The objective in this case is not only to find the most informative sample, but also the sample for which the oracle is most confident for its label, so that falsely annotated samples are not added in the training set. Towards this, the informativeness of a sample $\mathcal{E}(\mathcal{H}_0, \mathcal{U})$ and the confidence of the oracle $\mathcal{O}(\mathcal{U}, \mathcal{T}, d_k)$ in the label d_k must be jointly maximized. Thus a new function \mathcal{P} defining the benefit of selecting a sample u_i given its informativeness and the oracle's confidence has to be defined. Maximizing this function, the optimal sample u^* with the maximum informativeness and the higher confidence of the oracle for its label can be selected:

$$u^* = \underset{u_i \in \mathcal{U}}{\operatorname{argmax}} \mathcal{P}(u_i | \mathcal{E}(\mathcal{H}_0, u_i), \mathcal{O}(u_i, \mathbf{t}_i, d_k)) \quad (5.25)$$

The instance u^* that maximizes this function for the examined label d_k is added to the training set as positive if $d_k = +1$ or negative if $d_k = -1$. The expectation is that the addition of such samples in the training set will allow for the maximum performance gain of the classifier. With this approach, positive and negative instances can be selected independently, by setting $d_k = +1$ for the positive and $d_k = -1$ for the negative in Eq. 5.25.

Considering that active learning is an iterative method, in the next iterations $m \geq 2$ the informativeness is defined as $\mathcal{E}(\mathcal{H}_{m-1}, u)$. For simplicity, in the following, we will show the methodology for the first iteration using the baseline classifier \mathcal{H}_0 , while the full iterative approach can be seen in Algorithm 1.

5.2.3 Probabilistic Fusion in Active Learning

Given that the function $\mathcal{P}(u_i | \mathcal{E}(\mathcal{H}_0, u_i), \mathcal{O}(u_i, \mathbf{t}_i, d_k))$ cannot be analytically estimated (i.e., there is no analytical expression for this function), we approximate it as a probability. In this direction, let us denote the following random variables (RV): S_i , the RV modeling the event of selecting the image u_i ($S_i \in \{0, 1\}$), V_i , the RV modeling the probability that u_i is informative ($V_i \in [0, 1]$), T_i , the RV modeling the probability that u_i belongs to the examined concept d_k ($T_i \in [0, 1]$). Without loss of generality, we can assume that the function $\mathcal{P}(u_i | \mathcal{E}(\mathcal{H}_0, u_i), \mathcal{O}(u_i, \mathbf{t}_i, d_k))$ is proportional to the probability of selecting an instance ($S_i = 1$) given its informativeness (V_i) and the confidence of the oracle (T_i):

$$\mathcal{P}(u_i | \mathcal{E}(\mathcal{H}_0, u_i), \mathcal{O}(u_i, \mathbf{t}_i, d_k)) \sim P(S_i = 1 | V_i, T_i) \quad (5.26)$$

Consequently, in order to find the optimal u^* , instead of the function \mathcal{P} in Eq. 5.25, we can maximize its proportional probability function $P(S_i = 1 | V_i, T_i)$. In order to calculate

this probability, we make the reasonable assumption that the probability of an image being informative is conditionally independent of the probability that this image belongs to the examined concept (i.e., V_i and T_i are conditionally independent). Using Bayes rule and based on the assumption that V_i and T_i are independent, the probability $P(S|V, T)$ can be written as follows (from now on the subscripts of S , V and T will be omitted):

$$P(S|V, T) = \frac{P(V, T|S)P(S)}{P(V, T)} = \frac{P(S|V)P(S|T)P(V)P(T)}{P(V, T)P(S)} \quad (5.27)$$

In calculating $P(S|V, T)$ we may encounter two cases:

5.2.3.1 If $P(S=0|V, T) \neq 0$:

In order to calculate the probability $P(S=1|V, T)$ and eliminate the probabilities $P(V)$, $P(T)$ and $P(V, T)$, we divide the probability of selecting an image with the probability of not selecting it, following the methodology presented in [61].

$$\frac{P(S=1|V, T)}{P(S=0|V, T)} = \frac{\frac{P(S=1|V)P(S=1|T)P(V)P(T)}{P(V, T)P(S=1)}}{\frac{P(S=0|V)P(S=0|T)P(V)P(T)}{P(V, T)P(S=0)}}$$

Then by replacing $P(S=0|V, T)$, $P(S=0|V)$, $P(S=0|T)$ and $P(S=0)$ with their complements ($1 - P(S=1|V, T)$, $1 - P(S=1|V)$, $1 - P(S=1|T)$ and $1 - P(S=1)$, respectively), we get the following equation that computes $P(S=1|V, T)$:

$$P(S=1|V, T) = \frac{P(S=1|V)P(S=1|T)}{P(S=1) - P(S=1)P(S=1|T)} \cdot \dots \quad (5.28)$$

$$\frac{(1 - P(S=1))}{-P(S=1)P(S=1|V) + P(S=1|V)P(S=1|T)}$$

5.2.3.2 If $P(S=0|V, T) = 0$:

Assuming that the probabilities $P(V)$ and $P(T)$ cannot be 0 (i.e., there is always an *a priori* probability that an image is informative and belongs to the examined concept from Eq. 5.27 we have that either $P(S=0|V) = 0$ or $P(S=0|T) = 0$ (i.e., $P(S=1|V) = 1$ or $P(S=1|T) = 1$). Note that, in this case, Eq. 5.28 also produces the same result (i.e., $P(S=1|V, T) = 1 \Rightarrow P(S=0|V, T) = 0$), so from now on we will use Eq. 5.28 in all cases.

Thus, we only need to estimate three probabilities: $P(S=1)$, $P(S=1|V)$, and $P(S=1|T)$. The first one is set to 0.5 as the probability of selecting an image without any knowledge is the same as the probability of not selecting it. For the probability of selecting an image given its informativeness $P(S=1|V)$, we will approximate it using the informativeness criterion \mathcal{E} , which can be any criterion of the active learning theory [47]. In this work, we will be using the popular criterion that considers the most informative samples to be the ones lying on the separating hyperplane, since SVMs will be used to train the classification models. For the probability of selecting an image given the oracle's confidence $P(S=1|T)$, we will approximate it with a textual analysis algorithm that takes as input the tags of an image and provides as output the probability of an image being positive or negative with respect to a concept, as explained below. While any textual analysis algorithm can be used, in this work we used the popular BoW scheme due to its ability to additionally incorporate contextual information in deriving this probability.

5.2.3.3 Incorporating Informativeness in the Selection ($P(S|V)$)

The probability $P(S = 1|V)$ can be approximated using the informativeness criterion $\mathcal{E}(\mathcal{H}_0, u)$, where \mathcal{H}_0 is the baseline classifier. As mentioned above, SVMs were chosen as the classification scheme for this work. For the SVMs, a popular informativeness criterion deems the samples closest to the separating hyperplane as the most beneficial to the SVM classification model [49]. While other criteria could be used (e.g., halving the version space by explicitly calculating it when a sample is added), they would require a huge number of SVM models to be trained in order to directly measure the impact of the added sample on the version space. On the other hand, the selected criterion can be calculated very efficiently for linear SVMs since it requires only a dot product to calculate the Euclidean distance of a sample from the hyperplane, making it very attractive for a large-scale pool of candidates.

Initially, the baseline classifier \mathcal{H}_0 is trained using the manually annotated set \mathfrak{A} as a linear SVM classifier (i.e., $\mathcal{H}_0 = \{\mathbf{w}, b\}$, where \mathbf{w} is the normal vector to the hyperplane and b is the bias term). Then, for every candidate image in the pool of candidates $u_i \in \mathfrak{U}$, the distance from the hyperplane $d_i(\mathcal{H}_0, u_i)$ is extracted by applying the SVM classifier (\mathbf{u}_i here denotes the feature vector of the image u_i):

$$dv_i(\mathcal{H}_0, u_i) = \langle \mathbf{w}, \mathbf{u}_i \rangle + b \quad (5.29)$$

Based on [49], the samples with the minimum distance to the hyperplane are considered to be the most informative ones while the samples that lie outside the margin area of the SVM model are not expected to have any impact on the classifier. Thus the function $\mathcal{E}(\mathcal{H}_0, u_i)$, and consequently the probability $P(S|V)$, should be maximized (i.e., $P(S|V) = 1$) for the samples lying at the hyperplane and minimized (i.e., $P(S|V) = 0$) for the samples that are outside the margin area. Based on the above observation, we approximate the probability $P(S|V)$ with the following equation, which is also visualized in Figure 5.3a:

$$P(S|V) \sim \begin{cases} 1 - |dv_i| & \text{if } -1 < dv_i < 1 \\ 0 & \text{else} \end{cases} \quad (5.30)$$

where $|V|$ is the absolute value of the quantity in Eq. 5.29.

5.2.3.4 Measuring Oracle's Confidence ($P(S|T)$)

In order to measure the oracle's confidence $\mathcal{O}(u_i, \mathbf{t}_i, d_k)$ that the image $u_i \in \mathfrak{U}$ belongs to class d_k (i.e., is positive if $d_k = +1$ or negative if $d_k = -1$), we incorporate the associated

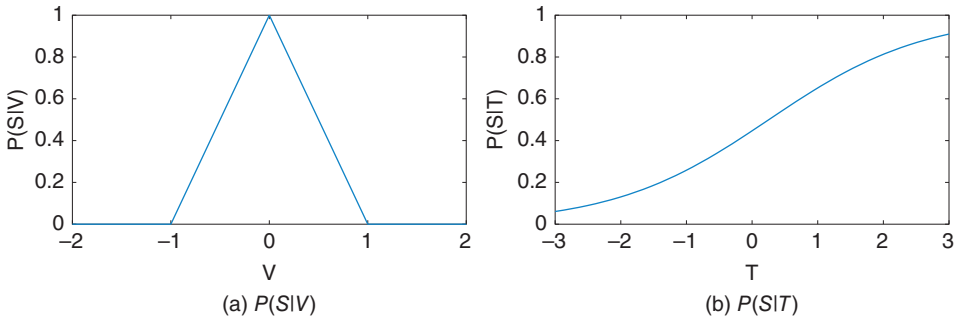


Figure 5.3 Probability of selecting a sample based on (a) visual and (b) textual information.

textual information that is provided in the form of tags. Opting to overcome the noisy nature of social tagging (i.e., lack of structure, ambiguity, redundancy, emotional tagging, etc.), we propose the utilization of the popular BoW scheme [48], due to its ability to capture the context of the whole set of tags instead of only the meaning of each tag independently (e.g., as in the case of tag-to-tag similarity based on WordNet [62]).

The vocabulary is extracted from a large image dataset crawled from Flickr (1 million images along with their tags). Initially the distinct tags of all images are gathered ($\sim 800k$ tags). The tags that are not included in WordNet are removed and the remaining tags compose the vocabulary ($\sim 47k$ tags). Then, following the BoW methodology, a tag histogram for each image is calculated by assigning the value 1 to the bins of the image tags in the vocabulary and 0 to the rest. This tag histogram is binary since no tag appears twice for the same image. PCA is applied to reduce the high dimensionality of the initial vocabulary ($\sim 47k$ distinct tags) to 7k, which was chosen so that 95% of the variance is kept. It is worth noting that, based on internal experiments, PCA has no effect on the performance of the textual analysis algorithm, while reducing significantly the dimensionality and the memory requirements for this step. More specifically, experimenting with tag-based classification on the MIRFLICKR-25k database, the performance of the original feature space and the PCA-reduced one were almost identical (27.34% vs. 27.35% in terms of mAP).

Afterwards, a linear SVM model ($\mathbf{w}^{\text{text}}, b^{\text{text}}$) is trained using the tag histograms as the feature vectors. In order to do this, a training set of images that contain both tags and manual annotations is utilized. The tags are required in order to calculate the feature vectors and the manual annotations to provide the class labels for training the model. In the testing procedure, for every tagged image $u_i \in \mathcal{U}$ the feature vector $\mathbf{u}_i^{\text{text}}$ is calculated as above and the SVM model is applied. This results in a value for each tagged image $dt_i(u_i)$, which corresponds to the distance of $\mathbf{u}_i^{\text{text}}$ from the textual hyperplane:

$$dt_i(u_i) = \langle \mathbf{w}^{\text{text}}, \mathbf{u}_i^{\text{text}} \rangle + b^{\text{text}} \quad (5.31)$$

Thus, the oracle's confidence $\mathcal{O}(u_i, \mathbf{t}_i, d_k = +1)$ that the image u_i is positive, and consequently the probability $P(S|T)$ when selective positive examples, can be calculated using Platt's sigmoid [63] as shown in Figure 5.3b:

$$P(S|T) = \mathcal{O}(u_i, \mathbf{t}_i, d_k = +1) = \frac{1}{1 + \exp(Adt_i + B)} \quad (5.32)$$

The parameters A and B are learnt on the training set using cross-validation. The probability of Eq. 5.32 is for selecting a positive image. In the case where we want to select negative images, the oracle's confidence $\mathcal{O}(u_i, \mathbf{t}_i, d_k = -1)$ that the image u_i is negative, and consequently the probability $P(S|T)$ when selective negative examples, is set to be the complement of Eq. 5.32:

$$P(S|T) = \mathcal{O}(u_i, \mathbf{t}_i, d_k = -1) = 1 - \mathcal{O}(u_i, \mathbf{t}_i, d_k = +1) \quad (5.33)$$

5.2.3.5 Re-training

In each iteration, the images are based on the probability $P(S = 1|V, T)$ and the top b_+ positive and b_- negative examples are selected to enhance the training set. In the case where both positive and negative samples are selected (as in SALIC), a new classifier is trained using the union of the old data and the newly selected examples as the training set. However, in the case where either only positive or only negative examples

are selected (as in some baselines we compare with), the classes become imbalanced, which is a typical problem in the machine learning field. In order to cope with the class imbalance problem we also apply a model aggregation method [60]. For the aggregation method, in each iteration m a new *weak* classifier $\mathcal{H}_m^{weak} = \{\mathbf{w}_m^{weak}, b_m^{weak}\}$ is trained on the newly selected examples of class $d_k = +1$ (or $d_k = -1$) and the original examples from the initial training set of the other class $d_k = -1$ (or $d_k = +1$). This classifier is called *weak* since it is not trained on the whole training set, but only on a portion of it. Then, in order to compute the classifier $\mathcal{H}_m = \{\mathbf{w}_m, b_m\}$ for iteration m , all the *weak* classifiers $[\mathcal{H}_1^{weak}, \mathcal{H}_2^{weak}, \dots, \mathcal{H}_m^{weak}]$ are aggregated by averaging their parameters:

$$\begin{aligned} \mathcal{H}_m &= \frac{1}{m} \sum_{i=1}^m \mathcal{H}_i^{weak} = \frac{m-1}{m} \mathcal{H}_{m-1} + \frac{1}{m} \mathcal{H}_m^{weak} \Rightarrow \\ \begin{cases} \mathbf{w}_m &= \frac{m-1}{m} \mathbf{w}_{m-1} + \frac{1}{m} \mathbf{w}_m^{weak} \\ b_m &= \frac{m-1}{m} b_{m-1} + \frac{1}{m} b_m^{weak} \end{cases} \end{aligned} \quad (5.34)$$

By incorporating this method, in each iteration the weak classifier is trained on equally sized sets of positive and negative instances and the final model is extracted by averaging the balanced weak classifiers of all iterations.

The complete algorithmic procedure of SALIC can be seen in Algorithm 5.1.

5.2.4 Experimental Comparison

Our objective in this section is to show the benefits of the proposed probabilistic fusion compared with baseline configurations incorporating either only visual or textual information, known late fusion techniques, as well as state-of-the-art methods in image classification with noisy labeled training data. Furthermore, we provide a parametric analysis of the proposed approach, demonstrating how each parameter can change SALIC's performance.

5.2.4.1 Datasets

Three datasets were employed for the purpose of our experiments. The imageCLEF dataset [64] (*Imageclef*) consists of 25,000 manually labeled images and was split into two parts, *Imageclef_{train}* and *Imageclef_{test}*, consisting of 15k train and 10k test images respectively. The dataset was annotated by a vocabulary of 94 concepts. The images of this dataset originate from Flickr and the tags were also provided. From this dataset we obtained the manually labelled dataset \mathfrak{Z} and the evaluation set (*Imageclef_{test}*)

The MIRFLICKR-1M dataset (*Mirflickr*) [43] consists of one million user tagged images harvested from Flickr. The images of *Mirflickr* were tagged with 862,115 distinct tags of which 46,937 were meaningful (included in WordNet). The tags that were not included in WordNet were removed and the images ending up with no tags were removed from this dataset (out of the 1 million, 131,302 images had no tags). In addition, given that the *Imageclef* dataset is a subset of *Mirflickr*, the images that are included in both sets were removed from *Mirflickr*. In our experiments, this dataset constitutes the pool of loosely tagged images.

The third dataset (*Imagenet*) includes images from the manually labeled ImageNET database [44]. The vocabulary of *Imagenet* consists of the 34 concepts that were included

Algorithm 5.1 1. SALIC

Input: Labeled data \mathfrak{L} with their labels Y , unlabeled data \mathfrak{U} , an oracle \mathcal{O} , the number of iterations N , the number of positive and negative instances selected in each iteration b_+, b_- .

Output: A classifier \mathcal{H}_N .

```

1:  $\mathcal{H}_0 = \{\mathbf{w}_0, b_0\} = \text{train}(\mathfrak{L}, Y)$ 
2: for  $m = 1 : N$  do
3:   // Get Positive Instances
4:    $d_k = +1$ ;
5:   for  $j = 1 : b_+$  do
6:     Choose  $u^* = \operatorname{argmax}_{u \in \mathfrak{U}} \mathcal{P}(u | \mathcal{E}(\mathcal{H}_0, u), \mathcal{O}(u, \mathbf{t}, d_k))$ ,
      $\mathcal{P}$  is calculated using Eqs. 5.26, 5.28, 5.30 and 5.32
7:      $\{\mathfrak{L}, Y\} \leftarrow \{\mathfrak{L}, Y\} \cup \{u^*, y^* = d_k\}$ 
8:      $\mathfrak{U} \leftarrow \mathfrak{U} \setminus u^*$ 
9:   end for
10:  // Get Negative Instances
11:   $d_k = -1$ ;
12:  for  $j = 1 : b_-$  do
13:    Choose  $u^* = \operatorname{argmax}_{u \in \mathfrak{U}} \mathcal{P}(u | \mathcal{E}(\mathcal{H}_0, u), \mathcal{O}(u, \mathbf{t}, d_k))$ ,
     $\mathcal{P}$  is calculated using Eqs. 5.26, 5.28, 5.30 and 5.33
14:     $\{\mathfrak{L}, Y\} \leftarrow \{\mathfrak{L}, Y\} \cup \{u^*, y^* = d_k\}$ 
15:     $\mathfrak{U} \leftarrow \mathfrak{U} \setminus u^*$ 
16:  end for
17:   $\mathcal{H}_m = \{\mathbf{w}_m, b_m\} = \text{train}(\mathfrak{L}, Y)$ 
18: end for

```

in both the synsets of ImageNet and the vocabulary of *Imageclef*. For every concept in the vocabulary of *Imagenet*, the images in the corresponding ImageNet synset were downloaded. This dataset consists of approximately 500k images.

5.2.4.2 Settings

For the visual representation of the images, we have used two popular approaches in order to verify the effectiveness of the proposed probabilistic fusion in different conditions: one that results in very high dimensional features that are of medium performance and one for low dimensional features that have shown remarkable performance. Our objective is to examine whether this difference in the discrimination ability of the two feature spaces will lead to particular requirements with respect to the sample selection process.

First, for the high dimensional features we have used the approach that was shown to perform best in [65]. More specifically gray SIFT features were extracted at densely selected key points at four scales, using the vl-feat library [66]. Principal component analysis was applied on the SIFT features, decreasing their dimensionality from 128 to 80. Then, Fisher vector encoding (256 GMM components) and spatial pyramids ($1 \times 1, 3 \times 1, 2 \times 2$ regions) were applied, resulting in a 327,680-dimensional feature vector per image.

Second, the low dimensional features are extracted from convolutional neural networks (CNNs), which have shown remarkable performance in the past years in both image annotation and object detection [67]. The implementation and the pretrained CNN models of [68] were used. More specifically, the models for extracting the lowest dimensional feature vectors were utilized (model CNN M 128 for the utilized implementation [68] or vgg-m-128 for the MatConvNet implementation [66]), resulting in only 128 dimensions.

Linear SVM models were trained for both feature spaces using the LIBSVM library [69] and were evaluated by mean average precision (mAP) (measured at all data points). The code for the implementation of the presented approach and the data to reproduce the results can be downloaded from.⁵

Considering that the utilized datasets consist of images labeled for multiple concepts, to conform with the binary classification requirements of the proposed active learning technique we transformed the multi label scheme to binary using the one-vs-all approach. More specifically, for every concept, all images including this concept in their list of annotated labels were considered as positive samples, leaving all the rest of the images as negative samples. Thus, the same image could serve as a positive sample for more than one concept. In our experimental study, 100 positive and 100 negative images from the 15k training images of $Imageclef_{train}$ were randomly selected to train the base-line classifiers (from now on called the $Imageclef_{init}$ dataset). The same initial examples were used for all experiments to allow a fair comparison. Then, in each iteration, 100 images were selected from the pool of candidates to enhance the training set. Based on the parametric analysis experiments presented in section 5.2.4.3, a good choice for the size of the initial training set is 100 positive and 100 negative manually labeled examples and for the size of the batch 100 examples (i.e., in each iteration we add (i) 100 positive or (ii) 100 negative or (iii) 50 positive and 50 negative depending on the training set expansion strategy). Finally, 50 iterations were conducted in all cases, i.e., in total, 5k images were added for each independent binary classification problem (i.e., concept).

5.2.4.3 Results

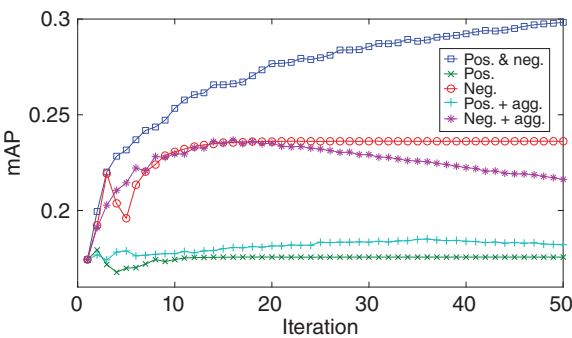
5.2.4.3.1 Expanding with Positive, Negative or Both

Our motivation to search whether positive or negative examples should be selected is based on the following intuitive and experimental observations. Intuitively, negative examples are expected to be more informative than positive ones, since they tend to cover a larger part of the concept space and as a consequence of the feature space (i.e., they depict a larger variety of objects compared to the positive examples which depict just one). In order to verify our intuitive observation experimentally, we developed the following experimental set up. Initially, by training a classifier on the $Imageclef_{init}$ and applying it on $Imageclef_{test}$ we achieved the performance of 17.43% in terms of mAP by averaging the performance of the 34 binary classification problems (i.e., one for each concept). Adding all the remaining positive examples of the $Imageclef_{init}$ dataset, so essentially using all positive vs 100 negative examples for each concept, the performance only increased to 18.38%. Adding all negative examples instead to the training set, i.e., using 100 positive vs all negative examples for each concept, the performance increased

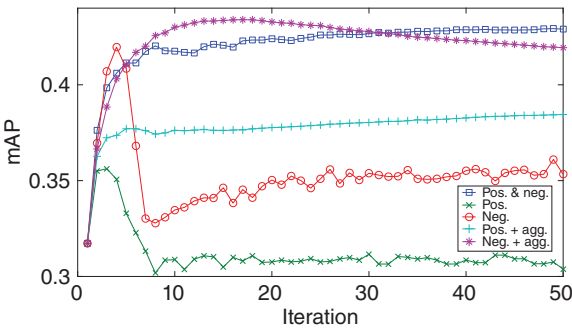
⁵ <http://mklab.itl.gr/research/salic>.

to 23.62%. On the other hand, if we use the full training set, the performance of the classifiers reaches the value of 29.67%. We can see that although the addition of negative examples is indeed more beneficial than the positive examples, it is far from achieving the performance obtained by using both positive and negative examples. The previous numerical results are for the Fisher-based features. Similar conclusions can be reached for the CNN-based features.

In order to test the contribution of each example type (i.e., positive or negative) in an iterative active learning scenario, we apply the active learning paradigm using the $Imageclef_{init}$ dataset to train the initial models and the *Imagenet* dataset as the pool of candidates. The reason is that *Imagenet* is manually annotated, removing the additional factor of how accurately the new samples are annotated. Only the 34 concepts of the *Imagenet* dataset were tested in this section. Then for every concept, 100 images were selected to enhance the initial training set: positive images were selected from the manually annotated *Imagenet* dataset and negative examples from the remaining negatives of the $Imageclef_{train}$ dataset. In each iteration, a total of 100 images are selected based on their *informativeness*. In Figure 5.4 the following five configurations are compared: (i) 50 positive and 50 negative images were added in the training set, (ii) 100 positive images were added in the training set, (iii) 100 negative images were added in the training set, (iv) 100 positive images were selected and the model aggregation method described in section 5.2.3.3 was used, and (v) 100 negative images were selected and the model aggregation method described in section 5.2.3.3 was used. The aggregation method was used in the cases where only one type of example was added (i.e., either positive or



(a) Fisher-based features



(b) CNN-based features

Figure 5.4 Iteratively adding positive, negative or both examples. Training set $Imageclef_{init}$ for 34 concepts, pool of candidates *Imagenet* for positive and $Imageclef_{train} \setminus Imageclef_{init}$ for negative, test set $Imageclef_{test}$ for 34 concepts. (a) Fisher-based features; (b) CNN-based features

negative) in order to tackle the class imbalance problem. On the contrary, this was not required when both positive and negative examples were added since in this case the two classes are balanced.

The results are shown in Figure 5.3a for the Fisher-based features and in Figure 5.3b for the CNN-based features. We can see that, in both cases, the addition of negative examples is more beneficial than that of positive ones, which confirms our previous findings as well as our intuitive explanation. Moreover, tackling the imbalance issue with the aggregation method increases the performance for the low dimensional CNN-based features while it does not improve the performance for the Fisher-based features. On the other hand, we can see that adding negative examples without aggregation (circles) increases the classifier's performance at the initial iterations (i.e., ~ up to 5) when the class imbalance problem is mild, but deteriorates significantly when the classes get severely imbalanced as more iterations are applied. The addition of both positive and negative examples (squares) outperforms greatly all the other variations for the Fisher-based features, as shown in Figure 5.3a, demonstrating the importance of adding examples from both classes. For the CNN-based features, the addition of negative examples with aggregation performs similarly to adding both positive and negative examples (Figure 5.3b). At the initial iterations, adding only negative examples with aggregation provides a higher boost in the classifiers' performance. However, after the 15th iteration their performance starts to deteriorate, while adding both positive and negative provides a more stable increase in the performance of the classifiers.

This difference in performance between the two feature spaces can be attributed to their discrimination ability. The CNN-based features are expected to be much more descriptive than the Fisher-based features, judging by their relative performance. However, a better performing classifier is expected to be confident for more samples from the pool of candidates and thus leave a smaller area in the sample space for which it is uncertain. This evidently means that there will be fewer informative samples in this case. Thus for more discriminant and low dimensional feature spaces, it is sufficient to just add negative images with aggregation, while for the less discriminant and high dimensional feature spaces adding both positive and negative images provides a significant performance increase. However, as is shown in the following experiments, adding data from both classes is more beneficial in real applications, where the pool of candidates is significantly larger and includes more positive images capturing a larger area in the feature space. For the next experiments the first configuration was selected (i.e., both positive and negative examples are added in the initial training set), since a more robust performance is achieved with this configuration across the two feature spaces.

5.2.4.3.2 Comparing with Sample Selection Approaches

In this experiment, we want to compare the performance boost that is achieved by SALIC, which selects new samples based on the joint consideration of visual and textual information, with two selective sampling baselines that rely on one modality alone: (i) self learning [70], where the images that maximize the certainty of the SVM model (Eq. 5.29) trained on visual information are selected to expand the training set and (ii) a text-based approach, where the images that maximize the oracle's *confidence* (Eq. 5.32) are selected. In our case, where the pool of candidates consists of user tagged images instead of the typical manually labeled images, the text-based approach is equivalent to the *random* baseline that every active learning approach compares with, since it neglects

completely the samples' informativeness. It was favored over a completely random approach (i.e., adding samples completely randomly without taking into account if they are actually positive or negative) in order to avoid adding false positives/negatives. The results can be seen in Figure 5.5. We can see that self learning does not provide any benefit to the models; this can be explained by the fact that adding examples far away from the hyperplane does not add any information to the model. Moreover, the fact that SALIC outperforms both the text-based and self-training approaches indicates the importance of incorporating the informativeness of new images in the selection strategy.

Comparing the two feature spaces, we can see that they exhibit similar behavior, with the only difference being self learning. In the case of the CNN features, adding new samples does not improve the classifier's performance. This can be attributed to the high discrimination ability of these features that forces self learning to select non-informative images far away from the hyperplane and thus not produce additional support vectors (i.e., the hyperplane is the same in all iterations). This is also in accordance to our previous observations. Furthermore, in order to demonstrate the effectiveness of SALIC, it is worth noting that the fully supervised result (i.e., using all 15,000 images of $Imageclef_{train}$ to train the classification models) achieves a performance of 27.74% in terms of mAP, while SALIC yields a maximum performance of 31.5%. On the contrary,

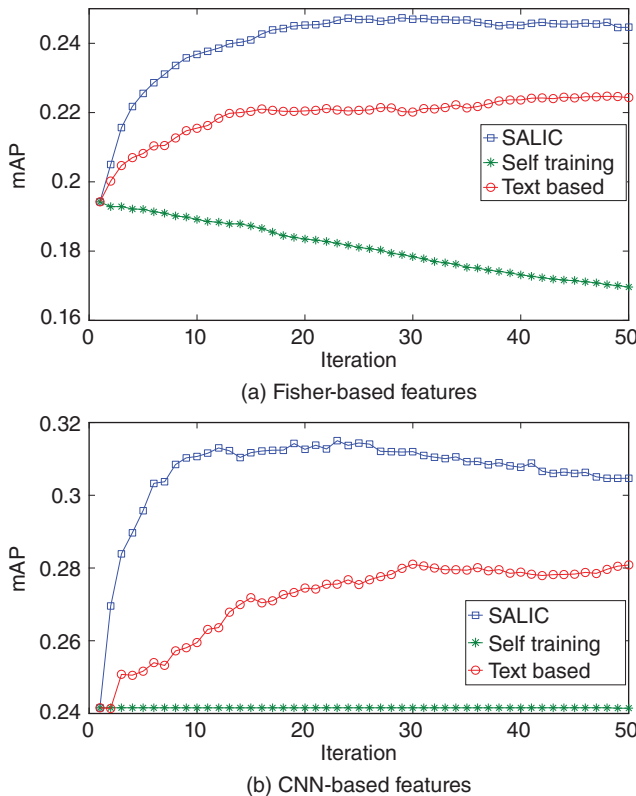


Figure 5.5 Comparing with sample selection baselines. Training set $Imageclef_{init}$, pool of candidates $Mirflickr$, test set $Imageclef_{test}$.

in the case of the Fisher features, although the hyperplane of the self-learning approach changes through the iterations, showing that the selected images are not necessarily far from the hyperplane, we can see that instead of increasing its performance it deteriorates. This can be attributed to the fact that the confidence of the oracle is not taken into account and thus the algorithm selects false positives/negatives to add to the training set (also due to Fisher features having lower discrimination ability). Finally, in this case the fully supervised result achieves a performance of 28.06% in terms of mAP, while SALIC yields a maximum performance of 25%, requiring only a small portion of annotated samples ($\sim 1.3\%$ of the total training set).

5.2.4.3.3 Comparing with Fusion Approaches

In this experiment, our objective is to show the benefit of utilizing the proposed probabilistic fusion approach for combining the informativeness of the samples ($P(S|V)$) and the oracle's confidence ($P(S|T)$). Towards this goal, we compare it with three baseline fusion strategies: (i) the arithmetic mean, where the arithmetic mean is used to combine the two probabilities $P(S|V)$ and $P(S|T)$ into the selection probability $P(S|V, T)$, (ii) the geometric mean, where the geometric mean is used to combine the two probabilities $P(S|V)$ and $P(S|T)$ into the selection probability $P(S|V, T)$, and (iii) a two-step approach that simulates the typical way that active learning is performed, while also keeping the classes balanced. More specifically, in the first step, the tagged images in the pool of candidates are annotated as positive or negative based on the oracle's confidence ($P(S|T)$). In the second step, the most informative of the positive and negative images are selected (i.e., the images that maximize the probability $P(S|V)$). The results are shown in Figure 5.6 for both Fisher-based (Figure 5.6a) and CNN-based (Figure 5.6b) features (the probabilistic fusion is noted as SALIC in the figure). It is obvious that using the simplistic arithmetic and geometric mean approaches the initial models do not improve significantly, showing that the selected samples are either not informative or false positives/negatives. On the contrary, the two-step approach yields a higher performance boost, which is probably due to the strict filtering it applies to the tagged images so that false positives/negatives are not selected, while keeping the notion of informativeness in the selection process. On the contrary, the proposed probabilistic fusions approach greatly outperforms all the baseline fusion strategies by selecting the samples that will have a higher impact when added in the training set, showcasing the importance of maximizing the joint probability $P(S|V, T)$.

5.2.4.3.4 Parameter Sensitivity Investigation

The objective of this experiment is to investigate the sensitivity of the proposed approach to various parameters. More specifically, we want to examine how the size of the initial training set and the number of images added in each iteration affects the performance of the resulting models. In order to investigate this, we ran SALIC using 20, 50, 100, 200 and 400 manually labeled images in the first iteration (half positive and half negative) and added 20, 50, 100, 200 and 400 images in each iteration, respectively (half positive and half negative). In order to visualize the results, we plotted the performance of each run with respect to the total number of samples used in the training set. The results can be seen in Figure 5.7. It is obvious that, as it would be expected, selecting more manually labeled images in the first iteration the performance of the initial models is significantly higher. For example, in the case of CNN-based features (Figure 5.7b), the

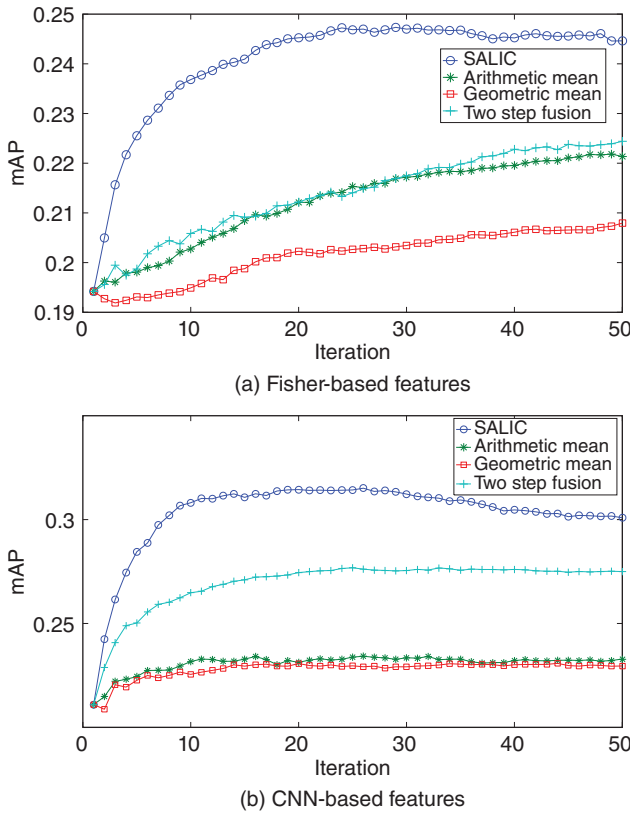


Figure 5.6 Comparing with fusion baselines. Training set $Imageclef_{init}$, pool of candidates *Mirflickr*, test set: $Imageclef_{test}$.

models created with 20 labeled examples (squares) have a performance of 18% mAP, while when we have 400 labeled examples (crosses) the performance rises to 26%. However, by adding training data with the help of SALIC, all the lines tend to converge to a similar performance (i.e. the black line grows from 18% mAP to 30% mAP, while the magenta line grows from 26% mAP to 32% mAP). Moreover, as expected, each classifier trained with a larger number of initially manually labeled data converges to a slightly higher performance and is much faster since only a few iterations are required. However, we also have to consider that these models require much more effort in order to acquire manually the increased number of labels that are needed to train the initial classifier. We can see that selecting 100 samples for the initial and the batch size seems to be a good compromise between training time, performance and annotation cost. Similar conclusions can be drawn for the Fisher-based features (Figure 5.7a).

5.2.4.3.5 Comparing with Existing Methods

In the following, we compare SALIC with existing methods in the literature. More specifically, we compare two types of research: the first is based on active learning [60] while the second approaches the problem of training set generation as weakly supervised learning [46, 59].

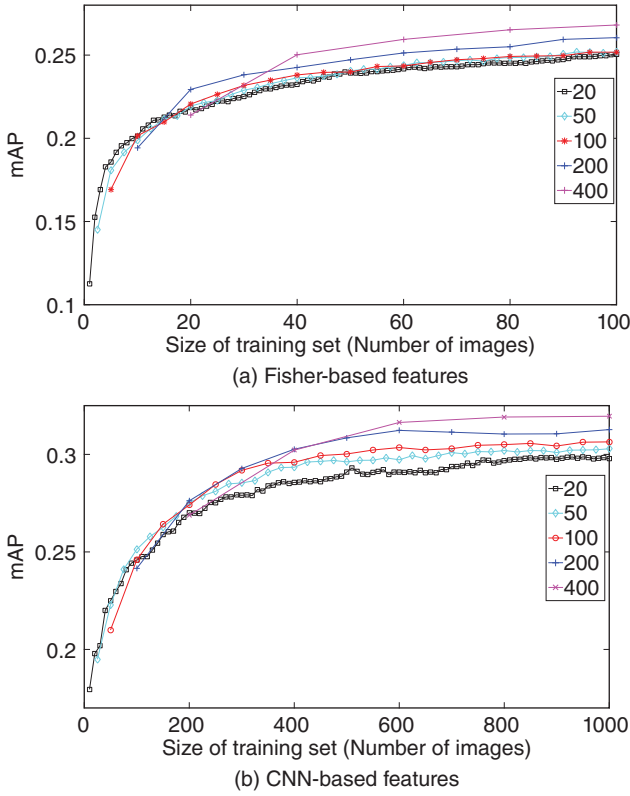


Figure 5.7 Parameter sensitivity. Training set: $Imageclef_{init}$, pool of candidates $Mirflickr$, test set $Imageclef_{test}$.

Comparing with active learning, we selected the most closely related work, named negative bootstrapping, which was presented in [60] (i.e., selecting in each iteration the 100 negative images that are most misclassified and utilizing the model aggregation approach presented in section 5.2.3.3 to cope with the class imbalance problem). The results can be seen in Figure 5.8. We can see that SALIC continuously increases the performance of the classifiers and constantly outperforms the other approach in both cases. It is also worth viewing in parallel the comparison between SALIC and [60] along with the comparison in Figure 5.3b between adding positive+negative (squares) and negative with aggregation (asterisks), which is also used by [60]. While in Figure 5.3b using positive and negative produces similar performance to only using negative with aggregation, SALIC performs much better compared to [60] (Figure 5.8b). This can be attributed to the proposed probabilistic fusion strategy between the two modalities, i.e., the samples' informativeness and the oracle's confidence, compared to [60], which is closer to a two-step fusion.

Comparing with weakly supervised learning, we selected a method that focused on creating a clean dataset from scratch by querying search engines [59] (SE) and another that aims at alleviating the noisy nature of Flickr images and tags by proposing a deep learning weakly supervised approach (DL). The first approach is based on the

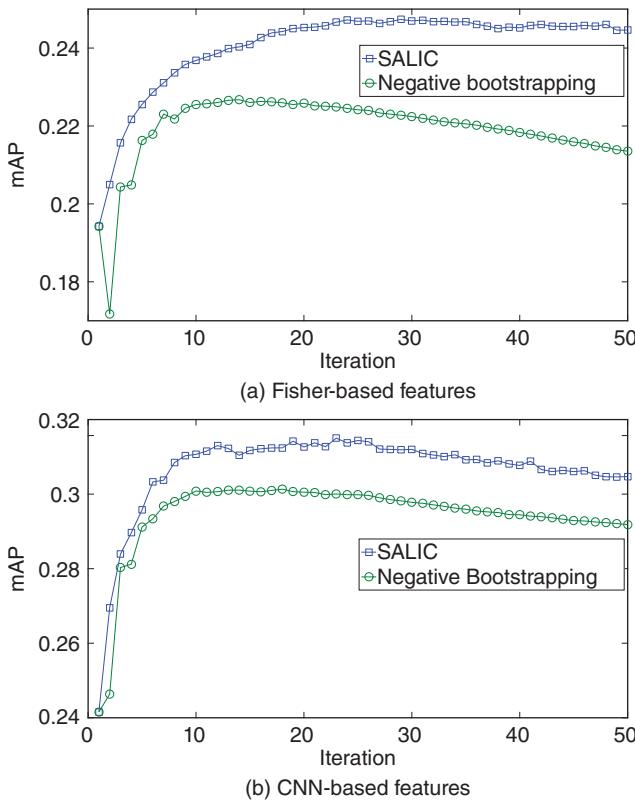


Figure 5.8 Comparing with active learning. Training set $Imageclef_{init}$, pool of candidates *Mirflickr*, test set $Imageclef_{test}$.

observation that search engines tend to provide accurate results for the top retrieved images. The proposed method constructs a set of queries by (i) translating the original query to 15 different languages, (ii) using hyponyms, hypernyms and synonyms from WordNet, and (iii) finding related terms within the results of the Google text search engine. They then query image search engines (i.e. Google, Bing and Flickr) with each term in the previously constructed set and retrieve the top 24 images of each query (24 was found to be optimal in [59]). The second method [46] proposes a noise-resistant version of logistic regression that can learn model parameters for any tag, called robust logistic regression (RLR). This is accomplished by introducing a parameter that models the probability that a user will supply a tag, conditioned on the tag being true for the image. This layer of RLR is added after the last fully connected layer of a deep neural network. In order to learn the parameters of the proposed network the model weights are initialized from AlexNet, a network trained previously on the manually labeled images from ImageNet, and are then refined using the user tagged images.

In order to compare SALIC with [46, 59], we applied our method to the dataset used in [59], which consists of 40 ImageNet concepts. As there was no manually labeled set for these concepts, we selected the initial 100+100 images to train H_0 from the *Mirflickr* dataset using the text-based approach. The results comparing SALIC, SE and

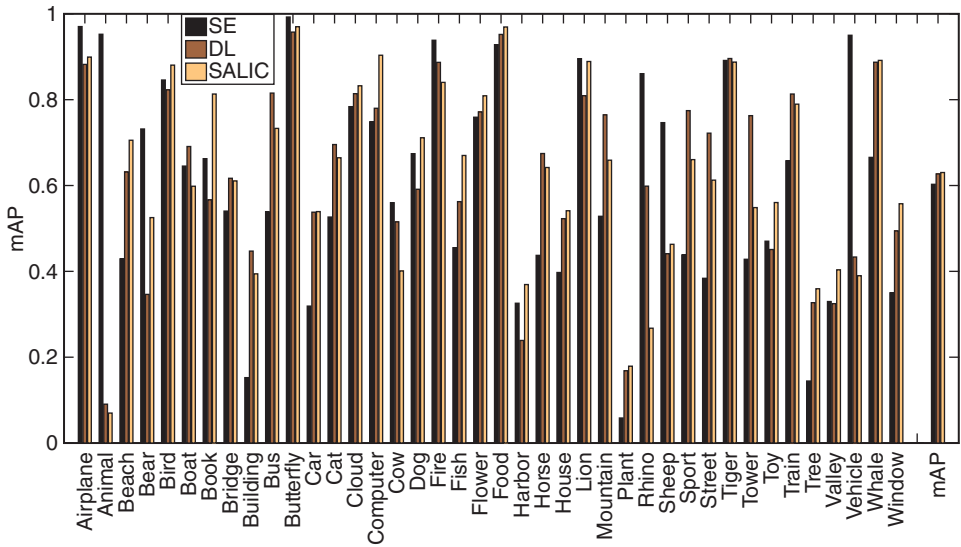


Figure 5.9 Comparing with weakly supervised learning. Training set (a) *Mirflickr* for SALIC and DL and (b) search engine images for SE, pool of candidates *Mirflickr* (only for SALIC), test set dataset from [59].

DL can be seen in Figure 5.9. In order to obtain comparable results, all methods used deep learning-based features with a fully sized penultimate layer (i.e., 4096). In addition, the number of iterations in the DL method was set to 8000 so that all four approaches required similar computational time (i.e., ~ 6 hours for the whole training procedure, including feature extraction for SALIC and SE). With respect to SE, we can see that SALIC outperforms this approach for 28 out of 40 concepts, yielding a mAP of 63.03%, compared to 60.3% for SE. With respect to DL, we can see that SALIC compares slightly favorably to DL (63.03% for SALIC versus 62.7% for DL), outperforming it for 23 concepts out of 40. It is also worth noting that there are three concepts, namely animal, rhino and vehicle, where SALIC fails to gather good quality training samples. For the concepts animal and vehicle, this can be partly attributed to their generic nature, since they include many diversiform sub-concepts (e.g., dogs, cats, insects, fish, etc. for animals, and boats, buses, cars, airplanes for vehicles). In addition, *Mirflickr* seems to be unfit for these concepts. This can be verified if we note the SE outperforms significantly both SALIC and DL for these concepts, which is the only method of the three based on search engines rather than *Mirflickr*.

5.3 Conclusions

In this chapter we presented early, late, and probabilistic multimodal fusion approaches and their application in multimedia retrieval and classification. Both multimedia retrieval and classification tasks need to fuse multiple modalities (textual, visual or high-level concepts) and often visual or textual information appears in multiple views by different visual descriptors. The heterogeneous character of multimedia information is handled by graph-based fusion approaches, as general approaches that

combine classic linear, non-linear, random-walk-based, and diffusion-based fusion of similarities. Additionally, we presented PLS regression, which maps different features into a common feature space and has been used for fusing multiple views and multiple modalities. Scalability issues has also been discussed through the exploitation of an initial filtering stage that keeps memory complexity at the level of two modalities. In the case of supervised techniques, an automatic active learning approach has been used to fuse textual and visual information, using informativeness scores and the oracle's confidence. This probabilistic late fusion approach has been compared with other baseline methods, taking into account scalability issues for the enrichment of the training set in big multimedia classification.

References

- 1 Magalhães, J. and Rüger, S. (2010) An information-theoretic framework for semantic-multimedia retrieval. *ACM Transactions on Information Systems*, 28 (4), 19.
- 2 Caicedo, J.C., BenAbdallah, J., González, F.A., and Nasraoui, O. (2012) Multimodal representation, indexing, automated annotation and retrieval of image collections via non-negative matrix factorization. *Neurocomputing*, 76 (1), 50–60.
- 3 Younessian, E., Mitamura, T., and Hauptmann, A. (2012) Multimodal knowledge-based analysis in multimedia event detection, in *Proceedings of the 2nd ACM International Conference on Multimedia Retrieval*, ACM, p. 51.
- 4 Kitanovski, I., Trojancanec, K., Dimitrovski, I., and Loskovska, S. (2013) Multimodal medical image retrieval, in *ICT Innovations 2012*, Springer, pp. 81–89.
- 5 Atrey, P.K., Hossain, M.A., El Saddik, A., and Kankanhalli, M.S. (2010) Multimodal fusion for multimedia analysis: a survey. *Multimedia Systems*, 16 (6), 345–379.
- 6 Lan, Z.z., Bao, L., Yu, S.I., Liu, W., and Hauptmann, A.G. (2014) Multimedia classification and event detection using double fusion. *Multimedia Tools and Applications*, 71 (1), 333–347.
- 7 Safadi, B., Sahuguet, M., and Huet, B. (2014) When textual and visual information join forces for multimedia retrieval, in *Proceedings of the International Conference on Multimedia Retrieval*, ACM, p. 265.
- 8 Clinchant, S., Csurka, G., Perronnin, F., and Renders, J.M. (2007) XRCE participation to imageval, in *ImageEval workshop at CVIR*.
- 9 Ah-Pine, J., Bressan, M., Clinchant, S., Csurka, G., Hoppenot, Y., and Renders, J.M. (2009) Crossing textual and visual content in different application scenarios. *Multimedia Tools and Applications*, 42 (1), 31–56.
- 10 Ah-Pine, J., Csurka, G., and Clinchant, S. (2015) Unsupervised visual and textual information fusion in cbmir using graph-based methods. *ACM Transactions on Information Systems*, 33 (2), 9.
- 11 Hsu, W.H., Kennedy, L.S., and Chang, S.F. (2007) Video search reranking through random walk over document-level context graph, in *Proceedings of the 15th International Conference on Multimedia*, ACM, pp. 971–980.
- 12 Langville, A.N. and Meyer, C.D. (2005) A survey of eigenvector methods for web information retrieval. *SIAM Review*, 47 (1), 135–161.

- 13 Gialampoukidis, I., Moumtzidou, A., Tsikrika, T., Vrochidis, S., and Kompatsiaris, I. (2016) Retrieval of multimedia objects by fusing multiple modalities, in *Proceedings of the 2016 ACM on International Conference on Multimedia Retrieval*, ACM, pp. 359–362.
- 14 Gialampoukidis, I., Moumtzidou, A., Liparas, D., Vrochidis, S., and Kompatsiaris, I. (2016) A hybrid graph-based and non-linear late fusion approach for multimedia retrieval, in *2016 14th International Workshop on Content-Based Multimedia Indexing (CBMI)*, IEEE, pp. 1–6.
- 15 Siddiquie, B., White, B., Sharma, A., and Davis, L.S. (2014) Multi-modal image retrieval for complex queries using small codes, in *Proceedings of the International Conference on Multimedia Retrieval*, ACM, p. 321.
- 16 Rasiwasia, N., Costa Pereira, J., Coviello, E., Doyle, G., Lanckriet, G.R., Levy, R., and Vasconcelos, N. (2010) A new approach to cross-modal multimedia retrieval, in *Proceedings of the International Conference on Multimedia*, ACM, pp. 251–260.
- 17 Wang, Y., Wu, F., Song, J., Li, X., and Zhuang, Y. (2014) Multi-modal mutual topic reinforce modeling for cross-media retrieval, in *Proceedings of the ACM International Conference on Multimedia*, ACM, pp. 307–316.
- 18 Gialampoukidis, I., Moumtzidou, A., Liparas, D., Tsikrika, T., Vrochidis, S., and Kompatsiaris, I. (2017) Multimedia retrieval based on non-linear graph-based fusion and partial least squares regression. *Multimedia Tools and Applications*, 76 (21), 22383–22403.
- 19 Tsikrika, T. and Kludas, J. (2010) The wikipedia image retrieval task, in *ImageCLEF*, Springer, pp. 163–183.
- 20 Grubinger, M., Clough, P., Müller, H., and Deselaers, T. (2006) The IAPR-TC 12 benchmark: A new evaluation resource for visual information systems, in *International Workshop OntoImage*, pp. 13–23.
- 21 Ionescu, B., Popescu, A., Lupu, M., Ginsca, A.L., and Müller, H. (2014) Retrieving diverse social images at mediaeval 2014: Challenge, dataset and evaluation, in *MediaEval*, CEUR-WS, Volume 1263.
- 22 Van De Sande, K.E., Gevers, T., and Snoek, C.G. (2010) Evaluating color descriptors for object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32 (9), 1582–1596.
- 23 Jégou, H., Douze, M., Schmid, C., and Pérez, P. (2010) Aggregating local descriptors into a compact image representation, in *2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, pp. 3304–3311.
- 24 Markatopoulou, F., Mezaris, V., and Patras, I. (2015) Cascade of classifiers based on binary, non-binary and deep convolutional network descriptors for video concept detection, in *2015 IEEE International Conference on Image Processing (ICIP)*, IEEE, pp. 1786–1790.
- 25 Simonyan, K. and Zisserman, A. (2014) Very deep convolutional networks for large-scale image recognition. *Computing Research Repository*, abs/1409.1556.
- 26 Daiber, J., Jakob, M., Hokamp, C., and Mendes, P.N. (2013) Improving efficiency and accuracy in multilingual entity extraction, in *Proceedings of the 9th International Conference on Semantic Systems*, ACM, pp. 121–124.
- 27 Hafner, J., Sawhney, H.S., Equitz, W., Flickner, M., and Niblack, W. (1995) Efficient color histogram indexing for quadratic form distance functions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17 (7), 729–736.

- 28 Sanderson, C. and Paliwal, K.K. (2004) Identity verification using speech and face information. *Digital Signal Processing*, 14 (5), 449–480.
- 29 Cheng, R. and Jin, Y. (2015) A competitive swarm optimizer for large scale optimization. *IEEE Transactions on Cybernetics*, 45 (2), 191–204.
- 30 Arandjelovic, R. and Zisserman, A. (2012) Multiple queries for large scale specific object retrieval, in *British Machine Vision Conference (BMVC)*, September 3–7, Surrey, UK pp. 1–11.
- 31 Wei, S., Zhao, Y., Zhu, Z., and Liu, N. (2010) Multimodal fusion for video search reranking. *IEEE Transactions on Knowledge and Data Engineering*, 22 (8), 1191–1199.
- 32 Fox, E.A. and Shaw, J.A. (1994) Combination of multiple searches. *Proceedings of the 2nd Text REtrieval Conference (TREC-2)*, National Institute of Standards and Technology Special Publication, pp. 243–243.
- 33 Montague, M. and Aslam, J.A. (2002) Condorcet fusion for improved retrieval, in *Proceedings of the 11th International Conference on Information and Knowledge Management*, ACM, pp. 538–548.
- 34 Cormack, G.V., Clarke, C.L., and Buettcher, S. (2009) Reciprocal rank fusion outperforms condorcet and individual rank learning methods, in *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, ACM, pp. 758–759.
- 35 Aslam, J.A. and Montague, M. (2001) Models for metasearch, in *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, ACM, pp. 276–284.
- 36 Mc Donald, K. and Smeaton, A.F. (2005) A comparison of score, rank and probability-based fusion methods for video shot retrieval, in *International Conference on Image and Video Retrieval*, Springer, pp. 61–70.
- 37 Zheng, L., Wang, S., Tian, L., He, F., Liu, Z., and Tian, Q. (2015) Query-adaptive late fusion for image search and person re-identification, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1741–1750.
- 38 Pedronette, D.C.G. and Torres, R.d.S. (2016) Rank diffusion for context-based image retrieval, in *Proceedings of the 2016 ACM on International Conference on Multimedia Retrieval*, ACM, pp. 321–325.
- 39 Pedronette, D.C.G. and Torres, R.d.S. (2016) A correlation graph approach for unsupervised manifold learning in image retrieval tasks. *Neurocomputing*, 208, 66–79.
- 40 Cohn, D., Atlas, L., and Ladner, R. (1994) Improving generalization with active learning. *Machine Learning*, 15 (2), 201–221.
- 41 Everingham, M., Gool, L., Williams, C.K., Winn, J., and Zisserman, A., The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results, <http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html>.
- 42 Chua, T.S., Tang, J., Hong, R., Li, H., Luo, Z., and Zheng, Y. (2009) Nus-wide: A real-world web image database from National University of Singapore, in *Proceedings of the ACM International Conference on Image and Video Retrieval (CIVR '09)*, ACM, New York, pp. 48:1–48:9, doi: 10.1145/1646396.1646452. URL <http://doi.acm.org/10.1145/1646396.1646452>.
- 43 Huiskes, M.J., Thomee, B., and Lew, M.S. (2010) New trends and ideas in visual concept detection: The MIR Flickr retrieval evaluation initiative, in *Proceedings of the International Conference on Multimedia Information Retrieval (MIR '10)*, ACM, New

- York, pp. 527–536, doi: 10.1145/1743384.1743475. URL <http://doi.acm.org/10.1145/1743384.1743475>.
- 44 Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., and Fei-Fei, L. (2009) Imagenet: A large-scale hierarchical image database, in *2009 IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2009)*, pp. 248–255, doi: 10.1109/CVPR.2009.5206848.
 - 45 Thomee, B., Elizalde, B., Shamma, D.A., Ni, K., Friedland, G., Poland, D., Borth, D., and Li, L.J. (2016) YFCC100m: The new data in multimedia research. *Communications of the ACM*, 59 (2), 64–73, doi: 10.1145/2812802. URL <http://doi.acm.org/10.1145/2812802>.
 - 46 Izadinia, H., Russell, B.C., Farhadi, A., Hoffman, M.D., and Hertzmann, A. (2015) Deep classifiers from image tags in the wild, in *Proceedings of the 2015 Workshop on Community-Organized Multimodal Mining: Opportunities for Novel Solutions (MMCommons '15)*, ACM, New York, pp. 13–18.
 - 47 Settles, B. (2009) Active learning literature survey, *Computer Sciences Technical Report 1648*, University of Wisconsin–Madison.
 - 48 Joachims, T. (1998) Text categorization with support vector machines: Learning with many relevant features, in *Machine Learning: ECML-98, Lecture Notes in Computer Science*, vol. 1398 (eds C. N'dellec and C. Rouveirol), Springer, Berlin, Heidelberg, pp. 137–142, doi: 10.1007/BFb0026683. URL <http://dx.doi.org/10.1007/BFb0026683>.
 - 49 Chang, E., Tong, S., Goh, K., and Chang, C.W. (2005) Support vector machine concept-dependent active learning for image retrieval. *IEEE Transactions on Multimedia*, 12, 3–13.
 - 50 Hoi, S.C.H. and Lyu, M.R. (2005) A semi-supervised active learning framework for image retrieval, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, IEEE Computer Society, Washington, DC, USA, pp. 302–309, doi: 10.1109/CVPR.2005.44. URL <http://dx.doi.org/10.1109/CVPR.2005.44>.
 - 51 Ebert, S., Fritz, M., and Schiele, B. (2012) RALF: A reinforced active learning formulation for object class recognition, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 3626–3633.
 - 52 Li, X. and Guo, Y. (2013) Adaptive active learning for image classification, in *2013 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 859–866, doi: 10.1109/CVPR.2013.116.
 - 53 Yan, Y., Rosales, R., Fung, G., and Dy, J. (2011) Active learning from crowds, in *Proceedings of the 28th International Conference on Machine Learning (ICML-11)* (eds L. Getoor and T. Scheffer), ACM, New York, pp. 1161–1168.
 - 54 Fang, M., Yin, J., and Tao, D. (2014) Active learning for crowdsourcing using knowledge transfer, in *Proceedings of the 28th AAAI Conference on Artificial Intelligence*, July 27–31, 2014, Québec City, Québec, Canada, pp. 1809–1815.
 - 55 Rodrigues, F., Pereira, F.C., and Ribeiro, B. (2014) Gaussian process classification and active learning with multiple annotators, in *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, pp. 433–441.
 - 56 Zhang, L., Ma, J., Cui, C., and Li, P. (2011) Active learning through notes data in Flickr: an effortless training data acquisition approach for object localization, in *Proceedings of the 1st ACM International Conference on Multimedia Retrieval (ICMR '11)*, ACM, New York, pp. 46:1–46:8.

- 57 Vijayanarasimhan, S. and Grauman, K. (2014) Large-scale live active learning: Training object detectors with crawled data and crowds. *International Journal of Computer Vision*, 108 (1-2), 97–114, doi: 10.1007/s11263-014-0721-9. URL <http://dx.doi.org/10.1007/s11263-014-0721-9>.
- 58 Nowak, S. and Rüger, S. (2010) How reliable are annotations via crowdsourcing: a study about inter-annotator agreement for multi-label image annotation, in *Proceedings of the International Conference on Multimedia Information Retrieval (MIR '10)*, ACM, New York, pp. 557–566.
- 59 Papadopolou, O. and Mezaris, V. (2015) Exploiting multiple web resources towards collecting positive training samples for visual concept learning, in *Proceedings of the 5th ACM International Conference on Multimedia Retrieval (ICMR '15)*, ACM, pp. 531–534.
- 60 Li, X., Snoek, C.G.M., Worring, M., Koelma, D., and Smeulders, A.W.M. (2013) Bootstrapping visual categorization with relevant negatives. *IEEE Transactions on Multimedia*, 15 (4), 933–945. URL <http://www.science.uva.nl/research/publications/2013/LiITM2013>.
- 61 Kordumova, S., Li, X., and Snoek, C.G. (2014) Best practices for learning video concept detectors from social media examples. *Multimedia Tools and Applications*, 74 (4), 1291–1315.
- 62 Fellbaum, C. (ed.) (1998) *WordNet: An Electronic Lexical Database (Language, Speech, and Communication)*, MIT Press. URL <http://www.amazon.ca/exec/obidos/redirect?tag=citeulike09-20&path=ASIN/026206197X>.
- 63 Platt, J.C. (1999) Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods, in *Advances in Large Margin Classifiers*, MIT Press, pp. 61–74.
- 64 Bart, T. and Adrian, P. (2012) Overview of the CLEF 2012 Flickr photo annotation and retrieval task, in *the Working Notes for the CLEF 2012 Lab and Workshop*, Rome, Italy.
- 65 Chatfield, K., Lempitsky, V., Vedaldi, A., and Zisserman, A. (2011) The devil is in the details: an evaluation of recent feature encoding methods, in *British Machine Vision Conference*, BMVA Press.
- 66 Vedaldi, A. and Fulkerson, B. (2008), VLFeat: An open and portable library of computer vision algorithms, <http://www.vlfeat.org/>.
- 67 Krizhevsky, A., Sutskever, I., and Hinton, G.E. (2012) Imagenet classification with deep convolutional neural networks, in *Advances in Neural Information Processing Systems 25*, Curran Associates, Inc., pp. 1097–1105. URL <http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf>.
- 68 Chatfield, K., Simonyan, K., Vedaldi, A., and Zisserman, A. (2014) Return of the devil in the details: Delving deep into convolutional nets, in *British Machine Vision Conference*, Computing Research Repository, 1405.3531.
- 69 Chang, C.C. and Lin, C.J. (2011) LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2, 27:1–27:27.
- 70 Ng, V. and Cardie, C. (2003) Bootstrapping coreference classifiers with multiple machine learning algorithms, in *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing (EMNLP '03)*, pp. 113–120.

6

Large-Scale Social Multimedia Analysis

Benjamin Bischke, Damian Borth and Andreas Dengel

The Internet is abundant with opinions, sentiments, and reflections of the society about products, brands, and institutions hidden under tons of irrelevant and unstructured data. This work addresses the contextual augmentation of events in social media streams in order to fully leverage the knowledge present in social multimedia by making three major contributions. First, a global study of the Twitter *Firehose* is presented. To our knowledge this is the first study of this kind and comprehension providing valuable insights about variability of tweets with respect to multimedia content. The results for more than one billion tweets show the great potential of the stream for many application domains. As a second key contribution, a fully automated system was developed for the augmentation of social multimedia with contextual information on a large scale. The system crawls/collects multimedia content from Twitter and performs a multi-modal analysis on it. The analysis considers temporal, visual, textual, geographical, and user-specific dimensions. Third, we present a near-duplicate detection approach based on deep learning to detect the most frequent images being propagated through Twitter during events.

6.1 Social Multimedia in Social Media Streams

This section deals with the importance of *social multimedia* and provides an overview of available social multimedia streams. A comprehensive analysis of social multimedia is performed on the micro-blogging platform Twitter with respect to different facets (e.g. textual, visual, geographical-information). The insights of this study are important when performing a detailed analysis of social multimedia on the micro-blogging platform for a specific use-case (e.g. large-scale multimedia opinion mining).

6.1.1 Social Multimedia

Social media has experienced extraordinary growth as a category of online discourse where people create, share, bookmark content and network at prodigious rate [1]. Along with the enormous user participation in social media, a vast amount of multimedia content is generated within these networks. Users in the social network Facebook produce

about 500 TB of multimedia content every day, including 300 million new images [2], while about 500 million new tweet messages are uploaded daily on the micro-blogging platform Twitter [3].

This user-generated content covers various multimedia formats such as textual, visual-, auditory-, and video-content, and is in many cases annotated with additional meta-data. Short text messages from Twitter, for example, contain in addition to the textual content also meta-data such as geographical-, timestamp-, user-profile and user-communication information as well as links to web resources such as images, videos, and news articles. In the following, such content found on social media streams is referred to as social multimedia. We define social multimedia as a special type of multimedia content which is enriched with extrinsic aspects of the underlying social media stream. In the simplest case this can be a link to a user profile. However, it can also cover more complex dimensions such as aspects of communication, and the interactions and behavior of the content with other users. Unlike raw multimedia content, this allows the intrinsic aspects of the content as well as contextual dimensions, which are important for numerous problems, to be considered. Gelli et al. [4] showed, for example, that the number of followers of a user on the social network Flickr is a better indicator for the prediction of image popularity compared to visual properties extracted from the image content.

6.1.2 Social Multimedia Streams

A wide range of social media service platforms has evolved within the last decade. A consequence of this is the focus on specific user intentions. For example, social network applications such as Facebook, Google+, and LinkedIn elaborate social interactions with friends and professional contacts, while other platforms, like Flickr and YouTube, focus on content sharing in the form of photos and videos, respectively.

The first aspect of social media addresses the type of content which is shared on the platform, for example as text-, image-, video- and meta-data. The social dimension is determined by the second aspect. Social interactions on these media applications can be motivated either by interest (i.e. listening and participating in political debates) or social networking (keep in touch with friends, maintaining and connecting to business contacts). A third aspect categorizes the visible content on the media stream more deeply. Published content can be either fact-based and objective (such as news articles and Wikipedia pages) or in form of user-generated content, including subjective opinions and discourses. The last dimension captures the main purpose and intention of the social multimedia stream.

Table 6.1 characterizes the most popular social media streams according to the categorization scheme described above. In many cases, the boundaries between the categories are blurry and not easy to categorize. For example, the main type of content created on Twitter is text messages. However, users can also embed images and videos into tweets. Twitter and Facebook are popular streams from a social media analysis point of view. The table shows that both streams cover a rich set of multimedia content originated by users and official institutes as well as containing the social networking aspects of users. Unlike most other social media streams, Twitter also provides simple and free (rate-limited) access to tweets. In the following we investigate Twitter’s characteristics as a social multimedia stream more deeply.

Table 6.1 This table shows the major social media streams characterized by the four dimensions of (1) multimedia content, (2) social networking, (3) content origin, and (4) purpose Twitter and Facebook cover the most aspects, showing multimedia content and social interactions as well as subjective and objective statements about events.

Social Media Stream	Multimedia content				Social Interactions		Content Origin		Purpose Category
	Photo	Text	Video	Meta-data	Interest motivated	Networking motivated	User gen. (opinions)	Fact based (news)	
Twitter	(x)	x	(x)	x	x	(x)	x	x	Microblogging
Sina Weibo	x	x	x	x	x	(x)	x	x	Microblogging
Facebook	(x)	x	(x)	x		x	x	x	Social Network
Renren	(x)	x	(x)	(x)		x	x	x	Social Network
Mixi	(x)	x	(x)	(x)		x	x	x	Social Network
Flickr	x		(x)	x	x		x		Photo Sharing
Instagram	x		(x)	x	x		x		Photo Sharing
Tumblr	(x)	x		(x)	x		x		Blogging
Wordpress	(x)	x	(x)		x		x	x	Blogging
Youtube			x	x	x		x	x	Video Sharing
LexisNexis	(x)	x		(x)	(x)			x	News
NewsCreed	(x)	x		(x)	(x)			x	News
Reddit	(x)	x		(x)	x		x		News Sharing
Wikipedia		x		x	(x)		(x)	x	Knowledge Sharing
Linkedin	(x)	x		x	(x)	x	x		Social Network
Google+	x	x	(x)	x		x	x		Social Network
Pintrest	x			x	x		x		Social Bookmarking
Daily Motion			(x)		x		x		Video Sharing
Bitly	x	x	(x)	(x)	x		x		Link Sharing

6.1.3 Analysis of the Twitter Firehose

In March 2016, Twitter had more than 100 million monthly active users [5] and nowadays it is one of the largest social multimedia streams worldwide. Besides information sharing, people use the service for online discourses and to express their opinions and thoughts about events, companies, people and current trending topics. Since the service can be also viewed as a social network, the service offers interesting starting points for a wide range of social multimedia analysis applications. Given this high popularity, we provide a comprehensive study of the social multimedia content found on this micro-blogging platform. The goal of this analysis is to get high-level insights into social multimedia content retrieved from Twitter according to different dimensions. The results are helpful for addressing questions with respect to the analysis and storage of the social multimedia content as well as its suitability from an application point of view.

6.1.3.1 Dataset: Overview

For the analysis of social multimedia content, online activities on the microblogging platform Twitter were collected from the *Twitter-API* in a distributed manner. The gathered dataset approximates a snapshot of a 10% random sample over all activities on Twitter during a timespan of three weeks in January 2016. Within the crawling period, about one billion tweet activities were collected. About 47.16% (480.9 Mio) of these activities are related to new posts, 30.23% (308.4 Mio) of the activities correspond to shared content that got retweeted while the remaining 22.60% (230.4 Mio) of activities are related to the deletion of tweets.

The collected activities contain in addition to text messages further meta-information, such as geographic, user-related information and links to web resources which have been analyzed within this context. The aim of this analysis is to assess the usage of the social media stream Twitter for (1) the usage of social multimedia analysis and (2) as an initial entry point for collecting data and linking it against other social media streams. Since deleted activities contain only the ID of the corresponding tweet, but not the original tweet or any of its meta-data, the following analytical investigations are based on shared and posted tweets only.

6.1.3.2 Linked Resource Analysis

The collected dataset of tweet activities was analyzed with respect to references to additional web resources. Such an analysis is vital for various applications of social multimedia analysis as it helps to get first insights into the usage of Twitter as a primary seed to crawl further multimedia content. Incorporating iconic images and influential web articles of an event from other data sources is an essential step in multimedia opinion mining. In the course of this link analysis the following aspects were considered: (1) the number of tweets containing any URL, (2) the number of tweets containing any URL to image resources and (3) the number of tweets containing any URL to web pages. The extracted numbers were additionally aggregated by the activity type and are shown in Table 6.2.

Besides analyzing the overall number of tweets which have links to multimedia content, it is also important to consider how often the same links occur. For example, popular images and controversially discussed news articles are often mentioned within multiple tweets. The extracted URLs were additionally aggregated by their unique

Table 6.2 The first row shows the total number of collected tweets aggregated per type (post, share, delete). The second and third rows show the number (and relative percentage) of tweets with any link to either images or web pages per activity type. The last two rows contain the unique number of links to images and web pages per activity type. Note that the reported percentages were computed from the number of tweets having any link (second and third rows).

	Posted Tweets	Shared Tweets	Deleted Tweets
#Tweets	480.9 Mio.	308.4 Mio.	230.4 Mio.
#Tweets with image urls	54.4 Mio. (11.3%)	143.2 Mio. (46.4%)	-
#Tweets with page urls	139.5 Mio. (28.9%)	65.4 Mio. (21.2%)	-
#unique image urls	49.4 Mio. (90.8%)	33.3 Mio. (23.2%)	-
#unique page urls	83.7 Mio. (60.0%)	18.9 Mio. (29.0%)	-

resource identifier (URI). Thereby every linked resource (image, video or web page) is treated as unique entity based on the corresponding URI. It is worth mentioning that this assumption is a simplification which is in reality not always true. Popular resources can get uploaded with minor or no modifications on different web servers and thereby get a new URL. Despite containing the same, similar or near duplicate content, these media resources are treated as different unique entities. The extracted results of the link analysis are summarized in Table 6.2. The following observations can be made:

- When comparing the percentage of tweets with links to resources (each per post and share activity), different outcomes depending on the concrete resource type are visible. Considering the percentage of tweets with links to web pages only, no significant difference between posts and shares is observable: about 28.9% of the posts contain URLs to web pages, while this number is slightly lower for retweets at 21.2%. The results for images reveal the following: about 46.4% of all retweets include references to image resources, while only 11.3% of the posts contain image references. This outcome indicates that images posted on social media streams convey vital elements of tweets which are important for users so they share them with others.
- Considering the unique number of URLs among the two activity types provides the following insights. For web-pages, about 71.0% of links in retweets reference to the same resources while the corresponding percentage is lower for posts at only 40.0%. This difference is even more marked for tweets with links to images: only 9.2% of posted tweets with an image URL reference to the same image resources. For the corresponding retweets this percentage is significantly higher at 76.8%. In a more abstract manner, this distribution of seen and unseen links in retweets with images follows the Pareto-Principle: 20% of the retweets contain links to unique images, while the remaining 80% of retweets reference to already seen images.

In order to get further insights into the referencing appearance of web resources, the distribution of link occurrences is taken into consideration. Figure 6.1 shows a log-log plot for the number of link occurrences in retweets with respect to their frequency in the dataset. Image and web page resources are regarded separately. The two plots in Figure 6.1 illustrate that the majority of links occurs in fewer than three tweets while a few references are covered by a tremendous number of tweets. It is also worth noticing that points in both of the plots approximate a straight line. This suggests

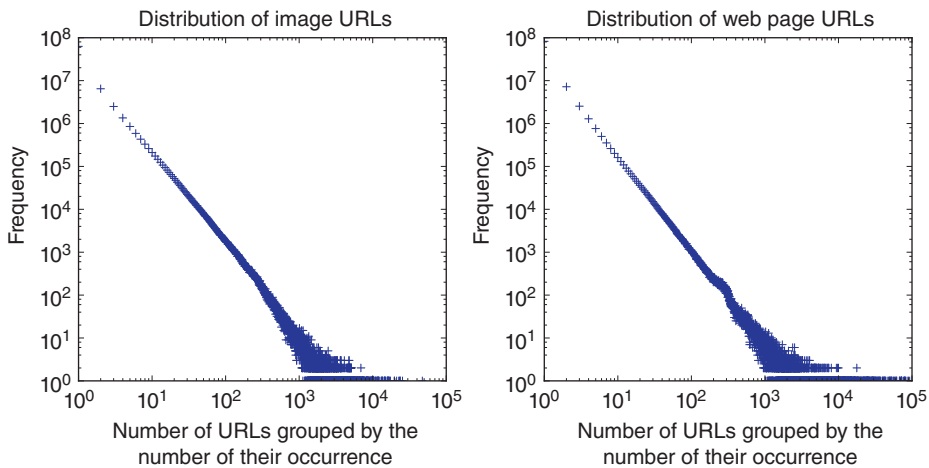


Figure 6.1 The occurrences of links and their frequency in retweets. The distribution is shown on double logarithmic axes (log-log plot), in which the power law follows a straight line.

that the distribution of link occurrences can be best approximated by the power law distribution.

Many real world systems, such as complex networks which represent the link structure of the World Wide Web or model social interactions between people, follow such a power law distribution [6]. One important characteristic of these systems is the *scale free property*. This means that, regardless of which scale the system is analyzed by, there are always so-called ‘hubs’ (high degree nodes) which grow polynomially with the system size. The reason for emphasizing this property within this context is its high relevancy for social multimedia analysis. For example, when performing opinion mining and extracting sentiments from referenced images or news articles, the correct aggregation of the results is important. Treating every image as an independent entity (based on the same URL) and averaging the analysis results over all images might not reflect the real opinions of the society. Instead, the power law distribution shows that the incorporation and weighting of more frequent links into the analysis is essential.

6.1.3.3 Image Content Analysis

Based on the previous link analysis, the most common images were downloaded and analyzed with respect to their content. In order to avoid analyzing images which are related to spam, only image resources with an URL occurrence greater than ten were considered. After combining the corresponding links from shares and posts with a count above that threshold, 1.0 Mio. unique image URLs were extracted. Since some images were deleted between the date of crawling and the date of the analysis, only 0.8 Mio images (about 84% of the identified images) could be successfully downloaded. All analytical experiments in this section built upon this image dataset.

Classification of Image Types. Incorporating visual content into social multimedia analyses, such as opinion mining, is often a crucial step, since images convey several concepts which are often not expressed in short text messages. Before performing a deeper analysis of the image content, such as sentiment extraction, it is vital to categorize images by their type. For example, synthetic images such as maps, charts, and logos

should be suppressed for sentiment analysis since they often do not convey meaningful sentiments. In order to assess the percentage of image which are suitable for sentiment analysis the following two image classes were introduced: (1) synthetic images and (2) natural images. For the classification, a convolutional neural network (CNN) with the architecture of AlexNet was fine-tuned on a custom dataset containing 2500 syntectic and natural images. The classifier achieves an accuracy of 97.0% on the test set and is described more detailed in section 6.3.2.2. From the overall 0.87 Mio. images, 0.24 Mio. images (27.5%) were classified as synthetic while the remainder were assigned to the natural image class. This relative high percentage of almost one third shows the importance of an additional filtering step in social multimedia analysis when working with images from social networks.

Top Concepts in Images. In addition to the classification of the image type, the most common concepts which are visible in the images were extracted and the concept classifier tool DeepSentiBank [7] was applied on every naturally classified image. The top five adjective–noun pairs (ANPs) determined by means of DeepSentiBank were aggregated by the nouns over all images. Table 6.3 lists the top 20 most frequently classified objects contained in the images together with the top five most frequently extracted adjectives. The nouns illustrated in the table show that the majority of nouns address

Table 6.3 The top 20 most frequently extracted concepts of all analyzed images. The last column lists combinations of the top five co-occurring adjectives for each noun.

Nouns	Coverage	Top five co-occurring adjectives
Girls	16.7%	Hot, sexy, stupid, cute, pretty
Face	13.6%	Ugly, silly, chubby, handsome, stupid
Hair	11.6%	Natural, great, healthy, pretty, bad
Guy	10.8%	Hot, nice, bad, handsome, creepy
Food	7.2%	Fancy, healthy, natural, great, weird
Dog	6.8%	Funny, mad, crazy, lost, tiny
Smile	6.7%	Adorable, great, lovely, charming, innocent
Baby	6.5%	Laughing, chubby, cute, funny, sleepy
Text	6.4%	Horizontal
Cat	6.1%	Funny, lost, crazy, poor, fat
Eyes	5.8%	Bright, pretty, crazy, beautiful, dark
Night	5.1%	Great, awesome, dark, clear, rainy
Sign	5.0%	Bad, stupid, funny, strange
kids	4.8%	Funny, hungry, super, excited, cute
student	4.5%	Talented, young, excited, helping, proud
Dress	4.5%	Fancy, sexy, traditional, elegant, stunning
Skin	4.4%	Smooth, clear, healthy, dark, fresh
Artist	3.9%	Young, Christian, favorite, talented, successful
Wedding	3.9%	Happy, elegant, Christian, traditional, outdoor
Body	3.8%	Healthy, hot, sexy, dead, strong

concepts related to *people* (girls, guy, face, hair, baby, eyes, kids), *animals* (dog, cat) and *food* (food).

6.1.3.4 Geographic Analysis

Geographical information is an important dimension of social multimedia. The direct enrichment of social multimedia with geographic information allows a link to be established to other data sources such as satellite imagery or content from the same locations from other social media streams. Additionally, the geographic context provides a complementary view of social multimedia which can be beneficial for more detailed analyses (e.g. how are opinions extracted from text content distributed across the world).

Precise GPS Annotations of Tweets. Twitter users have the option to annotate their own tweets with their current GPS position. From all 789.3 Mio. posts and shares shown in Table 6.4, only 16.8 Mio. (2.1%) tweets contained the precise GPS information. This low percentage % is slightly higher than the findings (of 0.7%) reported by other researchers in [8, 9].

Inferred Geo-Locations from User Profiles. Apart from user-given GPS tags, geographical information can be inferred indirectly by leveraging the profile data of users. To evaluate how many locations can be inferred with this approach, all user profiles were extracted from the dataset. In total 65.1 Mio. unique user profiles were gathered and approximately 43.1% (28.0 Mio.) of them contained a non-empty value in the profile location field. After taking only the user profiles with a given location into consideration, about half of the locations could be successfully mapped with the geographical database GeoNames¹ to a real location (i.e. *BigApple, US* to *New York, US*). Mapping the other half of the user profiles to real locations is more difficult because these profiles contain (1) non-existent locations such as “at the end of the world” or “in the blue sky” or (2) ambiguous location names that cannot be mapped to a unique address without additional knowledge of the state or country name. As a consequence, only 20.5% (13.3 Mio.) of all user profiles can be used for the inference of further geolocations. It is worth mentioning that this indirect inference approach makes a strong assumption about the trustworthiness of the majority of users when providing their real residential address in their profiles.

Distribution of Extracted Geo-Locations In this analysis, the distribution of geo-locations is examined more deeply. A demographic overview of active users on

Table 6.4 Tweets with GPS tags aggregated by country name. A significant number of these tweets (almost one third) is originated by users from United States.

No.	Country	Absolute	Per.	No.	Country	Absolute	Per.
1	United States	5.0 Mio.	29.8%	6	Turkey	0.6 Mio.	3.5%
2	Brasil	2.4 Mio.	14.5%	7	Philippines	0.5 Mio.	3.1%
3	Argentina	1.1 Mio.	6.6%	8	Indonesia	0.4 Mio.	2.5%
4	Japan	0.8 Mio.	5.2%	9	Malaysia	0.4 Mio.	2.5%
5	United Kingdom	0.7 Mio.	4.6%	10	Spain	0.3 Mio.	2.2%

1 <http://www.geonames.org/>

Twitter was obtained with regard to the countries the users belong to. GPS tags as well as inferred locations from user profiles were used for mapping to countries. Table 6.4 shows the top ten countries extracted from GPS annotations. The table shows that almost a third of all annotated tweets originated from users in the United States. This clearly shows that, even though the service is used worldwide, there is a strong bias in usage by American citizens. Countries with a relative high percentage of annotated tweets are Brazil (14.5%) and Argentina (6.6%). All other countries in the top ten list are covered by less than 5% of geo-tagged tweets, while the percentage drops below 2% for countries beyond the top ten list. The same findings can be observed from Figure 6.2,

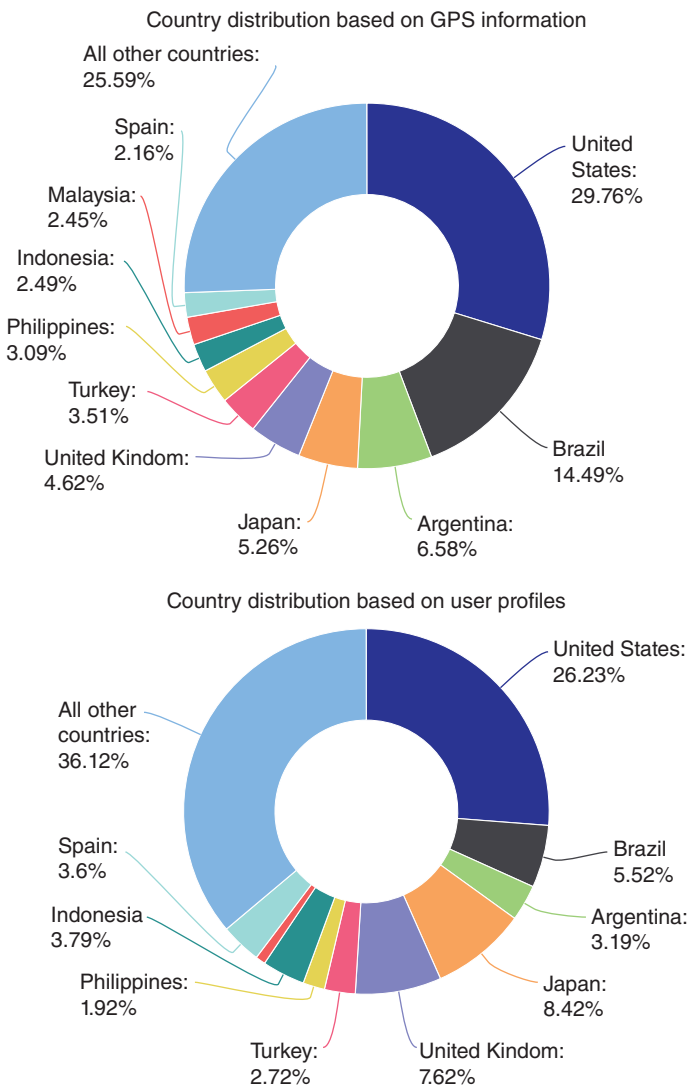


Figure 6.2 This plot shows the top ten countries determined by GPS versus profile information. Both approaches give a similar distribution of countries.

which shows a similar distribution of countries extracted from tweets based on GPS versus those that have been extracted from the profile information of the tweet. These insights have important consequences for the applicability of the social media stream for geographic-specific solutions when focusing on certain regions, such as countries in Europe. Since only 2% of all tweets contain GPS annotations and less than 2% of those originate from countries like Germany, France or Italy it is challenging to develop situational aware solutions for those regions by only building upon geo-tagged tweets. Further investigations are necessary to assess how representative these tweets are with respect to the corresponding population of a region and how many locations can be additionally inferred from textual and visual content.

6.1.3.5 Textual Analysis

The textual dimension of a tweet is most commonly used by users. However, text messages retrieved from the micro-blogging platform exhibit strong differences compared to textual content considered by classical natural language processing (NLP) problems (such as documents and web pages). The main differences are the limited length of the text message and the different vocabulary of abbreviations and slang compared to normal written text.

Since existing NLP approaches are not able to cope with such short and unstructured text messages, dedicated solutions for tasks such as named entity extraction [10], topic modeling [11, 12], topic clustering [13, 14], and topic detection [15] have been proposed. Authors of tweets can additionally include hashtags and user names in the text. Hashtags can be understood as a form of label or marker to link the content to current events, entities or sentiments. The mentioning of user names establishes the link from text message to another user profile in the form of a mention or reply.

Distribution of Languages. Closely related to geographical experiments is the examination of the languages used on Twitter. In this analysis, the language detector NLP<GO> was applied to the short text messages of posts and shares. After aggregating the languages and normalizing the distribution, the top ten most frequently used languages were extracted and are shown in Table 6.5. English is the dominating language on the service, being present in 36% of all tweets, while Japanese and Spanish also have relatively high coverage with 19% and 10%, respectively. Furthermore, the extracted results show that the language distribution is consistent with the outcome of geographical distribution. The languages of the top ten countries from Table 6.4 are

Table 6.5 The language distribution among all shares and posts. English is the most dominating language on the platform, being covered by almost one third of all tweets. Note that all languages in this table are spoken by the top ten countries in Table 6.4.

No.	Language	Absolute	Per.	No.	Language	Absolute	Per.
1	English	287.9 Mio.	36.5%	6	Korean	25.0 Mio.	3.2%
2	Japanese	155.9 Mio.	19.8%	7	Indonesian	24.1 Mio.	3.1%
3	Spanish	82.3 Mio.	10.4%	8	French	20.6 Mio.	2.6%
4	Portuguese	52.2 Mio.	6.6%	9	Turkish	17.2 Mio.	2.2%
5	Arabic	49.6 Mio.	6.3%	10	Thai	16.7 Mio.	2.1%

all covered by the top ten extracted languages in Table 6.5. The languages belonging to those countries which are not shown in Table 6.4 are covered by less than 2% of the tweets. However, this observation might be also due to the fact that users publish their messages in English and not in the mother tongue of their residential country in order to reach a wider audience.

6.2 Large-Scale Analysis of Social Multimedia

In this section we will first present techniques for processing social multimedia on a large scale. We will then have a look at state-of-the-art methods in machine learning for the analysis and information extraction of social multimedia.

6.2.1 Large-Scale Processing of Social Multimedia Analysis

Large-scale social multimedia can be characterized by the high volume, high velocity, and high variety of content and therefore be considered as an instance of the big data problem. Processing and handling such content requires special algorithms and a dedicated infrastructure. In the following, an overview of frameworks for large-scale processing is given. The frameworks are grouped into batch processing and stream processing. Batch-processing frameworks operate over a finite dataset and are typically used when access to the complete dataset is required. In stream-processing frameworks, the engine processes the data continuously while data is produced, assuming that the size of the dataset is infinite.

6.2.1.1 Batch-Processing Frameworks

Apache Hadoop² is one of the first batch-processing frameworks made large-scale batch processing of accessible. The framework consists of a stack of components and is based on the Map-Reduce [16] programming model in order to solve the problem of processing large datasets. The core idea of Map-Reduce is to apply a combination of map and reduce functions on the data. Instead of relying on expensive machines, the main aim of these frameworks is to make use of computer clusters with commodity machines. This is achieved by splitting the data into smaller chunks across a distributed file system. Data chunks are then processed in parallel on those machines where the data resides.

Apache Spark³ can be seen as an extension of the Map-Reduce engine in Apache Hadoop with additional performance optimizations. One significant performance gain is achieved by performing only in-memory batch computations. This is achieved through the introduction of a new memory model called resilient distributed datasets (RDDs), which allows the engine to operate on the memory model without the need to flush each operation back to disk for maintaining fault tolerance. Another optimization of Spark is achieved by constructing a directed acyclic graph (DAG) which represents the operations that are applied on the dataset. By analyzing the graph before the task scheduling, the engine can leverage relationships in the graph to coordinate the

² <http://hadoop.apache.org/>

³ <http://spark.apache.org/>

computation more effectively. Spark was originally designed for batch processing only and was later extended to also cope with stream processing. The engine introduces the concept of micro batches, a sequence of small batches which are processed by the batch engine. However, this approach does not provide real stream processing as it lacks low latency and real item-by-item processing as provided by other frameworks such as Apache Flink and Apache Storm.

6.2.1.2 Stream-Processing Frameworks

Apache Flink⁴ is a distributed dataflow engine which is primary designed for distributed computations over data streams but can be also used for traditional batch tasks. Batch processing is considered by Flink as a subset of stream processing with finite boundaries. The framework provides a high-throughput, low-latency streaming engine and supports event-time processing, which guarantees ordering and grouping of events even in the case of out-of-order or late-arriving data. Fault-tolerance and exactly-once semantics is achieved by taking snapshots during the computation, for recovery. Similar to Apache Spark, Flink analyzes and optimizes the computation of tasks before they are executed in a cluster or cloud environment.

Apache Storm⁵ is another stream-processing framework for real-time processing with very low latency. Storm describes small discrete operations and transformations of data into a *topology*. A topology can be understood in this context as directed acyclic graph, where the graph describes the data transformation pipeline of incoming data. The framework guarantees that each message is processed at least once, but does not guarantee an exactly-once semantics and that messages are processed in the correct order. However, the Storm framework can be extended by an additional abstraction layer called Trident which provides stateful processing ordering between batches and exactly-once guarantee of data. This is achieved through the introduction of a micro-batching model instead of a pure streaming system. As a consequence of this modified execution model, the low latency and fast processing provided by the Core-Storm engine are lost.

Apache Samza⁶ is a distributed stream-processing framework which heavily builds upon the messaging system Apache Kafka. Kafka is a massively scalable pub/sub message queue architected as a distributed transaction log developed by LinkedIn which provides low-latency access and replicated storage of real-time data feeds. By relying on Kafka, Samza can make direct use of the features provided by Kafka and guarantee properties such as data buffering, fault-tolerance and state storage. This state storage allows Samza to guarantee an at-least-once delivery but does not provide exactly-once semantics in the event of a failure. Compared to Apache Storm, the framework simplifies parts of stream processing by providing high-level abstractions but requires the additional deployment of Apache Kafka.

6.2.1.3 Distributed Processing Frameworks

In addition to frameworks dedicated to batch and stream processing, big data problems can also be handled by processing the workload in a distributed manner. This allows

⁴ <http://flink.apache.org/>

⁵ <http://storm.apache.org/>

⁶ <http://samza.apache.org/>

customized and optimized solutions to be developed, but is accompanied by the loss of features provided by the previously described frameworks. Depending on the complexity and abstraction, solutions can rely on low-level concepts such as sockets to higher level middleware such as ZeroMQ, RabbitMQ and Kafka.

Distributed Task Queue. *Celery*⁷ is a distributed task queue which uses message-oriented middleware systems to accomplish the architectural publish-subscribe design pattern. In this design pattern of message passing, publishers and consumers of messages are loosely coupled. The key idea when applying a distributed task queue is to restructure the processing work for a large dataset into the processing work for multiple smaller independent execution units. In the context of distributed task queues these units are called tasks and are represented as messages published on queues of a middleware broker. Worker processes can then consume specific messages and execute the corresponding computation concurrently. The choice of the underlying broker is a trade-off of performance and assured guarantees such as data buffering and fault tolerance.

6.2.2 Analysis of Social Multimedia

This section provides an overview of machine learning methods used for the analysis of social multimedia. These methods are illustrated by considering visual, textual, geographical and user-related facets of social multimedia.

6.2.2.1 Analysis of Visual Content

The visual analysis of social multimedia includes a wide range of research problems such as image classification, object detection, scene understanding, information extraction, and visual sentiment analysis. The common way to address these problems is to define a set of concepts to be recognized, extract features and train a model for the mapping from feature space to concepts. Thereby, the extraction of representative and invariant visual features is for many visual problems one of the key challenges. In the past CNNs (see chapter 1, ref. [11]) achieved significant success for visual recognition tasks such as image classification (see chapter 1, ref. [11, 24] and chapter 3, ref. [60]), semantic segmentation [17], object detection [18] and information extraction (see chapter 1, ref. [34]). Zeiler et al. showed that this is due to the fact that CNNs are able to learn a hierarchy of visual features for thousands of objects in million of images at different abstraction layers. In the context of visual multimedia analysis, features of pre-trained CNNs can often be reused (e.g. for visual clustering, visual concept detection, and visual information retrieval) or transferred to a new domain by transfer learning. We show and evaluate in the next section the usage of state-of-the-art CNN-based features and the concept of transfer learning via network fine-tuning in more detail.

6.2.2.2 Analysis of Textual Content

In textual analysis, methods from machine learning, NLP, and data mining are applied to address problems such as clustering, categorization, summarization of text, sentiment analysis and information extraction of concepts or entities. For the extraction of information, text is often processed with respect to a linguistic viewpoint.

⁷ <http://www.celeryproject.org/>

Thereby, tasks such as sentence parsing, stemming, named entity extraction (NER), and part-of-speech tagging (POS) are addressed. This is usually achieved through a lexical analysis and by leveraging the structure of the language. For predictive and regressive tasks, the most important step is to convert text into another representation which can be then used by existing machine learning models such as neural networks, support vector machines or random forests. With recent advances in deep learning, new representations of language learned by neural networks have been shown to be useful for numerous tasks in NLP. One such representation is word2vec [19] which uses a two-layered neural network to learn the representation of words in a lower dimensional vector space. Mikolov et al. showed that word2vec embedding captures semantic and syntactic relationships among words. Using such a representation for the training of standard machine learning classifiers outperformed traditional approaches in sentiment analysis. Current state-of-the-art approaches in this research area are moving further away from word representation towards the character level [20]. In particular, generative models such as Long Short-Term Memories (LSTMs) have shown the ability to learn powerful feature representations [20, 21].

6.2.2.3 Analysis of Geographical Content

Geographic annotations are an important property of social multimedia as they provide the base for numerous research directions such as situational awareness and behavioral analysis (e.g., crime prediction, health monitoring). Social multimedia can directly be annotated with precise GPS Information by users or contain indirect geographical locations in the textual and visual facets. The problem of estimating locations from where multimedia items (photos or videos) were captured, solely by inspecting the content and metadata of these items, has been addressed by the MediaEval *Placing Task* [22].

6.2.2.4 Analysis of User Content

Users in social media can be analyzed by using one of two approaches. The first approach is based on meta-information from the underlying social media stream and represents a feature-centric approach. Examples of such user features are gender, age, profession or political attitudes. Within the scope of the micro-blogging platform Twitter, extrinsic features of the social network such as the *follower counts* of user profiles or *retweet* and *favourite counts* of tweets can be leveraged to model the importance of users. The second approach is based on social network analysis. User interactions are modeled as a graph in which users are represented as nodes and relationships between users as edges between two nodes. In the context of Twitter, such relationships could be modeled by user replies such as retweets and mentions. Another relationship can be established by considering the followers of a user. The structure of these graphs is analyzed with quantitative and statistical metrics in order to capture important properties of the graph. A detailed overview of network analytic approaches focusing on social networks can be found in [23, 24].

6.3 Large-Scale Multimedia Opinion Mining System

Building upon the previous sections, we demonstrate how large-scale analysis of social multimedia can be applied to the specific case of multimedia opinion mining. In particular, a system is shown which collects social multimedia for a given topic from Twitter

and performs a multi-modal analysis of the content. The goal is to demonstrate the challenges which arise when crawling and analyzing social multimedia.

6.3.1 System Overview

The proposed system for multimedia opinion mining consists of three main components: (i) a social media data crawler, (ii) a social multimedia analysis engine, and (iii) a browser with interactive visualizations. The conceptual overview of the system is depicted in Figure 6.3.

The social media data crawler uses a set of initial keywords to collect multimedia content for an event. The crawler requests either Twitter's public search API to gather tweets from the past seven days or Twitter's historical powertrack (HPT)-API for older tweets. Retrieved tweets are passed to the social multimedia analysis engine. The goal of the analysis engine is the extraction and enrichment of contextual information for opinion mining from tweets. In this work, the main focus of the contextual augmentation is determined by textual, visual, temporal, geographical and social facets of opinion mining. Technically, for the retrieved tweets, multiple analysis tasks are instantiated and passed into the distributed task queue Celery. Worker processes, distributed among multiple compute nodes of a cluster, consume these tuples (of tasks and tweets) from the task queue, perform the corresponding analysis and store the results in the document-oriented database MongoDB. The contextualized content is accessed by a web application, which provides a suite of visualizations for interacting and exploring the extracted information. The first two components are described in the following sections in more detail.

6.3.2 Implementation Details

6.3.2.1 Social Media Data Crawler

When requesting Twitter's service endpoints a rule-based filtering mechanism is applied to select and filter relevant tweets. These rules can be, for example, matches of keywords and hashtags in text messages, geographic specifications of bounding boxes and locations where tweets originate from, or tweets filtered from specific user profiles.

In performing keyword-based crawling on social media streams, two main challenges arise which have to be considered. The first one addresses the semantic gap of keywords [25]. This means that same linguistic terms can refer to different concepts depending on the corresponding context. For example, the term "wildfire" is ambiguous within the scope of different scenarios. The term is often used in text messages to represent the concepts of (1) an instance of a fire in the forest or bush, (2) a smartphone model (by HTC), (3) songs with the same name (e.g. from Michael Murphy) and (4) as part of speech such as "spread like wildfire". Another challenge when relying on keywords for retrieval of social multimedia is the proper incorporation of relevant keywords into the crawling strategy. In many cases the exact vocabulary necessary to cover all relevant tweets for a given event is not known beforehand. These problems are solved by applying an iterative query expansion of related terms along with a relevance feedback measure.

The approach of using a query expansion is often used in the context of information retrieval [26]. The main purpose in those systems is to induce a better relevance ordering for retrieved media objects. The main goal of the keyword expansion in this work is the

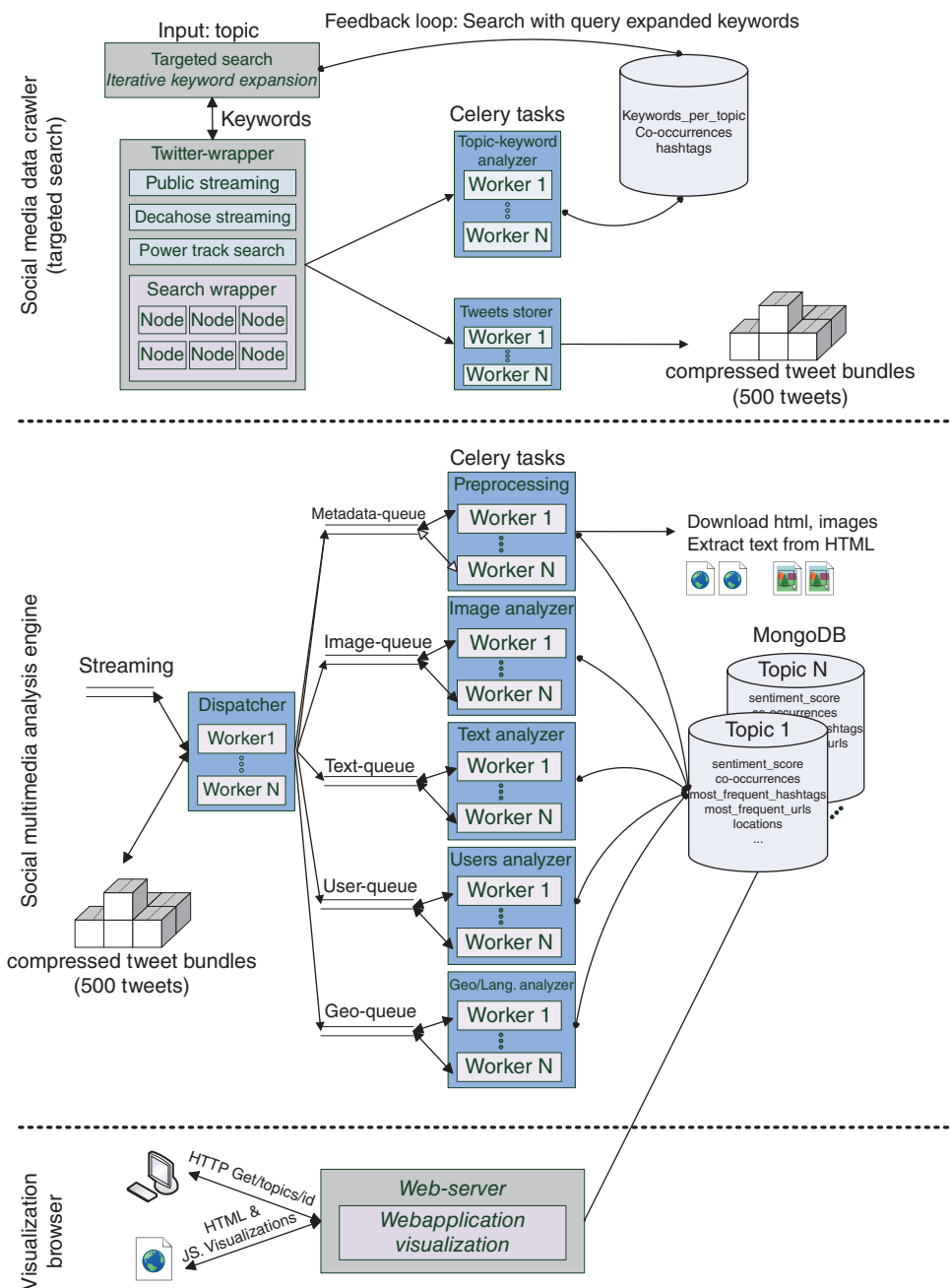


Figure 6.3 Architectural overview of the developed system. There are three main components: a social media data crawler, a social multimedia analysis engine and a visualization browser.

identification of representative term descriptors for the social multimedia corpus. Using query expansion for social media crawling works as follows. The system first collects social multimedia content based on user-given keywords. The gathered result set is then used to infer further potential keywords which can be used to refine the initial search and to collect further tweets. These acquired tweets usually do not contain the original keywords but are highly relevant for the searched event. Empirical evaluations showed that hashtag terms and entity based on co-occurrences yield the best retrieval results for the expansion of topic terms. The same features showed good results for the retrieval of social multimedia in the context of emergency situation in [27, 28].

Since expanded terms can still refer to very general and ambiguous concepts (e.g. wild-fire) an additional relevance feedback measure is used. For each expanded query term, the crawler collects 2000 tweets from the search API and computes a distribution of the hashtags from those tweets as a bag of words (BoW). The cosine similarity between the $\text{BoW}_{\text{retrieved}}$ of the new retrieved tweets and the $\text{BoW}_{\text{initial}}$ of the initially collected tweets is then computed. If the similarity is below an empirically determined threshold (70%), the expanded query term is considered as too general and is rejected for the query expansion. After each successful query expansion, the initial hashtag distribution is updated with the newly collected tweets and the process is repeated.

6.3.2.2 Social Multimedia Analysis

The goal of the analysis engine is the extraction and enrichment of contextual information for opinion mining from tweets. In this work, the main focus of the contextual augmentation is determined by textual, visual, temporal, geographical and social dimensions for opinion mining. The analysis is separated into five main processing units: (i) resource fetching, (ii) text analysis, (iii) image analysis, (iv) user analysis, and (v) geo and language analysis. By separating the computation into different tasks, the processing work is grouped logically. This allows the system to be easily extended and replaced with new features as well as executing the computation in a concurrent and distributed manner.

A detailed description of all five processing units of the system would go beyond the scope of this work, hence we will briefly summarize the main analysis tasks in each unit. The analysis on visual content is explained in more detail since from a research point of view many challenging problems arise. We introduce in this context a novel approach for near-duplicate detection of images and discuss concepts of transfer learning.

The system identifies in the first unit (1) URLs in tweets which refer to images, videos and web pages, follows the links and downloads the corresponding resources. In unit (2), text filtering, textual sentiment analysis, named entity extraction, identification of influential Tweet texts and the extraction of top hashtags is performed. Unit (3) filters synthetic images, groups near-duplicate images and extracts visual sentiments. A user target classification and the identification of influential users is achieved in unit (4). The last unit (5) aggregates tweets by geographical location (by country and as a heat map) and aggregates detected languages. It is worth mentioning that we also linked the geographic information to other sources, such as satellite imagery from NASA's Landsat8-Satellite. Our work in [29] shows how extracted information from social multimedia can complement remote sensed data to raise situational awareness and provide a comprehensive overview in the context of an emergency response for natural disasters.

6.3.2.3 Analysis of Visual Content

Images are a popular media format that are often used in social multimedia (see analysis in Section 6.1.2). Images can be more expressive than text, especially with regard to very short text messages like tweets. In addition to the representation of scenes and objects, they often convey important concepts, ideas and emotions which are vital for opinion mining.

Filtering of Synthetic Images. In social media, users uploaded, in addition to natural images, synthetic images like maps, logos or charts [30]. For many applications, such as visual sentiment analysis, this type of image is not of interest and should be suppressed in the analysis by a visual classifier. We developed a deep neural network classifier which can discriminate between the two image type classes and explain in the following the principle of transfer learning for it. The idea behind transfer learning is to reuse the learned knowledge of a source problem and apply it to a similar target problem. A pre-trained CNN, CaffeNet, builds the foundation of the classifier and was fine-tuned for the particular task of image type classification. CaffeNet is a replication of the popular AlexNet architecture defined by Krizhevsky et al. [31] and has five convolutional layers, three fully connected layers and produces a softmax distribution over 1000 object classes. The network was trained on ImageNet (see chapter 5, ref. [44]) and has been shown in the past to produce features that generalize well for other problems. We kept the architecture of CaffeNet and replaced only the last fully connected layer (fc8 layer) by a new layer with only two output classes. For the model training, the network was initialized and fixed with the weights of the pretrained model, while the weights in the last layer were initialized randomly and learned on a custom dataset containing about 5400 synthetic and natural images. By freezing weights in first layers, already learned filters of the model are transferred to the new domain and can be reused for the image type classification task. On our test set, the classifier achieved an accuracy of 98% and outperformed existing approaches [30, 32] based on handcrafted features.

Near-Duplicate Detection. Due to the sharing culture present on microblogging websites, there is a large amount of duplicate content online. The same image is often hosted on many different servers and retweeted in many different social circles [30]. This can be seen in Figure 6.4, where multiple images that are different pixelwise but show the same scene were posted during one event. In order to avoid analyzing duplicated images multiple times and to identify iconic images which stay in the memory of society, image duplicate detection beyond the unification of URLs is necessary. Additional image manipulations such as cropping, scaling and storing of different JPEG quality levels applied to the images make duplicate detection purely based on image pixels infeasible. It is therefore necessary to identify near duplicates based on the content of the image. This problem is solved by utilizing a high-level representation of the image content and searching every image for its nearest neighbors, as performed in information retrieval systems. We feed every image into a CNN with the architecture of AlexNet [31] and extract the activations of the fc6 layer as 4098 dimensional feature vectors. Images are ranked to a given query image by computing the cosine similarity between the corresponding feature vectors. Images above an empirically determined threshold (0.8) are considered as near duplicates and together with the searched image are removed from the image set. The approach is applied iteratively on the remaining images until the image set is empty. The result of the near-duplicate detection approach is shown in



Figure 6.4 These images illustrate two detected clusters of duplicate images on Twitter for the trending topic “car explosion in Berlin” in March 2016. Both groups show the same car from a different perspective. Even though all images depict the same content, the images are different pixelwise.

Figure 6.4. In the next section we evaluate features from multiple layers (fc6, fc7, pool5) of state-of-the-art CNNs.

Image Sentiment Analysis. In order to perform visual sentiment analysis on the filtered images, the concept classifier tool DeepSentiBank [7] is applied. DeepSentiBank is a fine-tuned CNN that is based on visual sentiment ontology (VSO) [33]. VSO consists of 3244 ANPs, but a subset of VSO including 2089 ANPs was used to train the classifier. DeepSentiBank is applied to every image and the top 10 ANPs with the highest probability are taken from the 2098 dimensional output vector. In addition, the top 20 most frequently computed ANPs are extracted for every day of observation.

6.3.3 Evaluations: Analysis of Visual Content

6.3.3.1 Filtering of Synthetic Images

In order to recognize synthetic images in social media streams, a new classifier was developed. The following subsections provide an overview of the used dataset, a baseline comparison and experiments about the generalization ability of the used features for training a classifier.

Generated Dataset. The dataset used for this experiment consists of 2700 synthetic images crawled from the web and 2700 natural images randomly sampled from the YFCC100M dataset [34]. Synthetic images were collected by querying the search engine Bing⁸ for the following five classes: *logos*, *maps*, *diagrams*, *charts* and *cartoons*. The five image classes are almost equally sized with minor deviations (of ten images across classes). In a post-processing step the downloaded images were converted into an RGB

⁸ <https://bing.com>

image such that successive feature extraction approaches rely for all images on the same image format and can be compared against each other. 70% of synthetic and natural images were randomly selected for the training set, the remaining images were used for the test set.

Network Fine-Tuning vs. Handcrafted Features. The developed classifier is based on the architecture of AlexNet, in particular its implementation of CaffeNet is used. The fc8 layer of this model is replaced by a new layer consisting of only two output classes. The resulting network was fine-tuned on the above described training set and achieved after 800 iterations the best classification accuracy of 97.83% on the test set. For a baseline comparison the approach of [30] was implemented. The authors extract (1) 64-bin color histograms in RGB and HSV color space and (2) edge histograms grouped into vertical, horizontal, 45 and 135 degree categories as described detailed in [32]. The final feature vector is represented by concatenating and normalizing the computed histograms. Using these handcrafted features, the best accuracy of 96.3% was attained by training a support vector machine (SVM) with radial-basis function as kernel. The comparison of the two approaches shows in both cases a high classification performance, where the fine-tuned features are performing slightly better. It is also faster to compute and therefore more suitable for large-scale processing in real time.

Transferability of Features. Since the fine-tuned neural network achieved a high performance, it is interesting to understand the influence of the softmax function and the pretrained features. A high ability of the features to generalize would show that they can be well transferred to similar problems like image type classification. For this purpose, the feature representation of the fc7 layer of the pretrained model was extracted and used to train a linear SVM. This accounts for an accuracy of 98%, which is marginally better than the fine-tuned network model. However, more importantly, these results show that the high accuracy is mostly independent of the classifier like SVM or deep neural network fully connected component and mainly due to the robust representation of the features. Another interesting aspect in this context addresses the transferability of the features extracted from different layers of the network. Instead of relying on the highest level representation of the network using fc7 features, the same experiment was performed on the fc6 layer:

- 1) fc8 and fc7 layers were removed from AlexNet, a new output layer was attached and finetuned.
- 2) A linear SVM was trained on extracted fc6 features from the pretrained model.

The classification results are similar to those in the experiment before and even slightly better. Details of the results are listed in Table 6.6. The table shows that fine-tuning the last layers of AlexNet results in an 1% improvement against the baseline, while using the representation of the last layers as features for training a linear SVM yields to a 2% improvement, reaching 98.6% accuracy for the fc6 layer features for a linear SVM. One explanation why this approach produces even better results may be the fact that the fc7 layer is a very high-level representation but already specialized towards the problem task. The fc6 layer might be a less problem specific representation, hence being better suited for transfer learning. In addition to the better classification results, the removal of the last fully connected layer also reduces the model size from 256 MB to 180 MB. This is due to the fact that most weights are contained in the last layers.

Table 6.6 Evaluation results of image type classification approaches.

Approach	Accuracy
Baseline: histogram features + SVM (RBF)[30]	96.10%
Fine-tuning on new fc8 layer (old fc8 layer removed)	97.83%
Fine-tuning on new fc7 layer (old fc8 and fc7 layers removed)	97.13%
Fc7 features from pretrained model + SVM (linear)	98.34%
Fc6 features from pretrained model + SVM (linear)	98.64%

6.3.3.2 Near-Duplicate Detection

The work in [35] shows that image classification using CNNs is invariant to a few image quality distortions such as contrast reduction and distortion by means of a lower JPEG quality. Motivated by this observation, the following work investigates into how accurate near-duplicates of images can be detected by using features extracted from state-of-the-art CNNs. Since these networks exhibit different network architectures and are trained on mixed datasets, a first evaluation aims to find suited features that can be used for a near-duplicate detection system.

Near-Duplicate Dataset. The goal of this experiment is to evaluate features of state-of-the-art deep neural networks among common image transformations. The evaluation is based on the CopyDays dataset [36] from INRIA. This dataset contains 157 original images and includes three types of image transformations:

- 1) 16% resizing of the image surface plus e different levels of JPEG quality compressions (3%, 5%, 8%, 10%, 15%, 20%, 30%, 50%, 75%)
- 2) cropping of the surface between 5% and 80% in nine steps (10%, 15%, 20%, 30%, 40%, 50%, 60%, 70%, 80%)
- 3) strong image transformations such as print and scan, painting, and perspective effects on the image.

While transformations (1) and (2) each generate nine newly transformed images, transformation (3) produces in total only 223 images. Since strong attacks represent an extreme case which is usually not observed in social media streams and because of the imbalanced number of distorted images of the transformations ((1) and (2) each produce 1413 images, while (3) only includes 223 images), the evaluation depends only on the former two transformations.

It is worth mentioning that for a general near-duplication system there are further attacks such as blurring or noise addition which are not included in this dataset. As the focus of this work lies on social media analysis, where most manipulations include cropping, scaling, and using different JPEG qualities, the evaluation is based on the former two transformations only.

Network Model and Feature Selection. Various features from AlexNet [31], GoogLeNet (see chapter 1, ref. [24]), ResNet (see chapter 3, ref. [60]), and VGG-Net (see chapter 1, ref. [23]) have been extracted from the original images and the manipulated ones due to transformations (1) and (2). For every original image, all manipulated images were ranked according to the cosine similarity of the feature vectors. Since

Table 6.7 The performance of features extracted from state-of-the-art CNNs for the use of near-duplicate detection. It can be seen that AlexNet (trained on ImageNet) shows the best mean MAP@9-performance against cropping, scaling, and JPEG compression attacks.

Pretrained model (dataset)	Feature	MAP@9 cropping	MAP@9 JPEG quality	MAP@9 avg. both
AlexNet (ImageNet)	fc7	89.7%	93.9%	91.8%
	fc6	90.0%	97.6%	93.8%
	pool5	87.8%	99.6%	93.7%
VGG_S (ImageNet)	fc7	92.0%	90.4%	91.2%
	fc6	91.9%	94.0%	92.9%
	pool5	86.9%	99.1%	93.0%
GoogLeNet (ImageNet)	pool5/7x7_s1	90.4%	84.5%	87.4%
AlexNet (Places205)	fc7	89.6%	87.5%	88.5%
	fc6	91.0%	92.4%	91.7%
	pool5	88.7%	98.1%	93.4%
VGG_16 (Places205)	fc7	89.0%	80.9%	84.5%
	fc6	90.9%	86.9%	88.9%
	pool5	88.1%	93.9%	91.0%
GoogLeNet (Places205)	pool5/7x7_s1	90.4%	73.9%	82.2%
ResNet	fc1000	92.8%	82.2%	87.5%

there are exactly nine images for the first two transformations, the mean average precision at nine (MAP@9) was computed as the evaluation metric. Table 6.7 shows the results of the features on the two transformations. The last column contains the average of the former two metrics of MAP@9. fc6 features from AlexNet achieved the best performance overall. Considering only cropping or JPEG quality transformations, the best accuracies are observed by fc1000 from ResNet and AlexNet’s pool5 features, respectively.

6.4 Conclusion

In this chapter we first presented the importance of social multimedia and gave an overview of available social multimedia streams. One such important data source is Twitter, on which we performed a comprehensive study. The study showed the potential of this stream and revealed important aspects (e.g., scale-free property, distribution of media URLs in tweets, geographical/language distributions) which have to be considered when performing a deeper analysis of social multimedia. In section 6.2 we

addressed the analysis of social multimedia. We first considered the scalability aspect and provided an overview of state-of-the-art frameworks for large-scale data processing. Different methods of machine learning were presented for the content analysis of social multimedia with respect to different modalities (textual, visual, geographical, user aspects). Building upon the previous sections, we showed in section 6.3 how the concepts of large-scale social multimedia analysis can be applied to a real-world scenario. We considered the use of multimedia opinion mining and discussed solutions for the crawling and analysis of social multimedia. A major focus of this analysis was the visual content, where we explained transfer learning with deep neural networks and introduced a new approach for near-duplicate detection of images based on features from CNNs.

References

- 1 Asur, S. and Huberman, B.A. (2010) Predicting the future with social media, in *2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT)*, vol. 1, IEEE, pp. 492–499.
- 2 Tam, C.D. (2012) *Facebook processes more than 500 TB of data daily*. <https://www.cnet.com/news/facebook-processes-more-than-500-tb-of-data-daily/> (accessed 3 October 2016).
- 3 Twitter (2013) *New Tweets per second record, and how!* <https://blog.twitter.com/2013/new-tweets-per-second-record-/and-how> (accessed 4 October 2016).
- 4 Gelli, F., Uricchio, T., Bertini, M., Del Bimbo, A., and Chang, S.F. (2015) Image popularity prediction in social media using sentiment and context features, in *Proceedings of the 23rd ACM International Conference on Multimedia*, ACM, pp. 907–910.
- 5 Twitter (2012) *Twitter turns six*. <https://blog.twitter.com/2012/twitter-turns-six> (accessed 3 October 2016).
- 6 Barabási, A.L. (2009) Scale-free networks: a decade and beyond. *Science*, 325 (5939), 412–413.
- 7 Chen, T., Borth, D., Darrell, T., and Chang, S.F. (2014) DeepSentibank: Visual sentiment concept classification with deep convolutional neural networks. *arXiv preprint arXiv:1410.8586*.
- 8 Graham, M., Hale, S.A., and Gaffney, D. (2013) Where in the world are you? Geolocation and language identification in Twitter. *ArXiv e-prints*.
- 9 Hecht, B., Hong, L., Suh, B., and Chi, E.H. (2011) Tweets from Justin Bieber's heart: the dynamics of the location field in user profiles, in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ACM, pp. 237–246.
- 10 Li, C., Weng, J., He, Q., Yao, Y., Datta, A., Sun, A., and Lee, B.S. (2012) Twiner: named entity recognition in targeted Twitter stream, in *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*, ACM, pp. 721–730.
- 11 Zuo, Y., Wu, J., Zhang, H., Lin, H., Wang, F., Xu, K., and Xiong, H. (2016) Topic modeling of short texts: A pseudo-document view, in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ACM, pp. 2105–2114.

- 12 Yang, S.H., Kolcz, A., Schlaikjer, A., and Gupta, P. (2014) Large-scale high-precision topic modeling on Twitter, in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ACM, pp. 1907–1916.
- 13 O'Connor, B., Krieger, M., and Ahn, D. (2010) Tweetmotif: Exploratory search and topic summarization for Twitter, in *ICWSM*.
- 14 Rosa, K.D., Shah, R., Lin, B., Gershman, A., and Frederking, R. (2011) Topical clustering of tweets. *Proceedings of the ACM Conference on Research & Development on Information Retrieval (SIGIR), 3rd Workshop on Social Web Search and Mining (SWSM)*, ACM, Beiiing.
- 15 Cataldi, M., Di Caro, L., and Schifanella, C. (2010) Emerging topic detection on Twitter based on temporal and social terms evaluation, in *Proceedings of the Tenth International Workshop on Multimedia Data Mining*, ACM, p. 4.
- 16 Dean, J. and Ghemawat, S. (2008) Mapreduce: simplified data processing on large clusters. *Communications of the ACM*, 51 (1), 107–113.
- 17 Badrinarayanan, V., Kendall, A., and Cipolla, R. (2015) Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *arXiv preprint arXiv:1511.00561*.
- 18 Ren, S., He, K., Girshick, R., and Sun, J. (2015) Faster R-CNN: Towards real-time object detection with region proposal networks, in *Advances in Neural Information Processing Systems*, pp. 91–99, IEEE Computer Society.
- 19 Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013) Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.
- 20 Radford, A., Jozefowicz, R., and Sutskever, I. (2017) Learning to generate reviews and discovering sentiment. *arXiv preprint arXiv:1704.01444*.
- 21 Kim, Y., Jernite, Y., Sontag, D., and Rush, A.M. (2016) Character-aware neural language models, in *Proceedings of the 30th AAAI Conference on Artificial Intelligence (AAAI '16)*, Phoenix, Arizona, USA, pp. 2741–2749.
- 22 Choi, J., Thomee, B., Friedland, G., Cao, L., Ni, K., Borth, D., Elizalde, B., Gottlieb, L., Carrano, C., Pearce, R. et al. (2014) The placing task: A large-scale geo-estimation challenge for social-media videos and images, in *Proceedings of the 3rd ACM Multimedia Workshop on Geotagging and Its Applications in Multimedia*, ACM, pp. 27–31.
- 23 Barabási, A.L. (2003), Linked: The new science of networks *American Journal of Physics*, 71 (4).
- 24 Scott, J. (2012) *Social Network Analysis*, Sage.
- 25 Hu, X., Sun, N., Zhang, C., and Chua, T.S. (2009) Exploiting internal and external semantics for the clustering of short texts using world knowledge, in *Proceedings of the 18th ACM Conference on Information and Knowledge Management*, ACM, pp. 919–928.
- 26 Carpineto, C. and Romano, G. (2012) A survey of automatic query expansion in information retrieval. *ACM Computing Surveys*, 44 (1), 1.
- 27 Abel, F., Hauff, C., Houben, G.J., Stronkman, R., and Tao, K. (2012) Twitcident: fighting fire with information from social web streams, in *Proceedings of the 21st International Conference on the World Wide Web*, ACM, pp. 305–308.
- 28 Abel, F., Hauff, C., Houben, G.J., Stronkman, R., and Tao, K. (2012) Semantics+ filtering+ search= twitcident. Exploring information in social web streams, in

- Proceedings of the 23rd ACM Conference on Hypertext and Social Media*, ACM, pp. 285–294.
- 29 Bischke, B., Borth, D., Schulze, C., and Dengel, A. (2016) Contextual enrichment of remote-sensed events with social media streams, in *Proceedings of the 2016 ACM on Multimedia Conference*, ACM, pp. 1077–1081.
 - 30 McParlane, P.J., McMinn, A.J., and Jose, J.M. (2014) Picture the scene...: Visually summarising social media events, in *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, ACM, pp. 1459–1468.
 - 31 Krizhevsky, A., Sutskever, I., and Hinton, G.E. (2012) Imagenet classification with deep convolutional neural networks, in *Advances in Neural Information Processing Systems 25* (eds F. Pereira, C.J.C. Burges, L. Bottou, and K.Q. Weinberger), Curran Associates, Inc., pp. 1097–1105.
 - 32 Wang, F. and Kan, M.Y. (2006) Npic: Hierarchical synthetic image classification using image search and generic features, in *International Conference on Image and Video Retrieval*, Springer, pp. 473–482.
 - 33 Borth, D., Ji, R., Chen, T., Breuel, T., and Chang, S.F. (2013) Large-scale visual sentiment ontology and detectors using adjective noun pairs, in *Proceedings of the 21st ACM International Conference on Multimedia*, ACM, pp. 223–232.
 - 34 Thomee, B., Shamma, D.A., Friedland, G., Elizalde, B., Ni, K., Poland, D., Borth, D., and Li, L.J. (2016) Yfcc100m: The new data in multimedia research. *Communications of the ACM*, 59 (2), 64–73, doi: 10.1145/2812802. URL <http://doi.acm.org/10.1145/2812802>.
 - 35 Dodge, S. and Karam, L. (2016) Understanding how image quality affects deep neural networks. *arXiv preprint arXiv:1604.04004*.
 - 36 Douze, M., Jégou, H., Sandhawalia, H., Amsaleg, L., and Schmid, C. (2009) Evaluation of GIST descriptors for web-scale image search, in *Proceedings of the ACM International Conference on Image and Video Retrieval*, ACM, p. 19.

7

Privacy and Audiovisual Content: Protecting Users as Big Multimedia Data Grows Bigger

Martha Larson, Jaeyoung Choi, Manel Slokom, Zekeriya Erkin, Gerald Friedland and Arjen P. de Vries

7.1 Introduction

In this chapter we discuss the relationship between privacy and algorithms that make use of large amounts of multimedia data. As users continue to post their audiovisual content online, and as companies continue to collect user profiles and interaction data, concerns about privacy are becoming increasingly urgent. With much of their behavior, users unwittingly share information about themselves that could compromise their privacy. As big multimedia data continues to grow bigger, it is essential to understand not only the risks for users, but also the potential of using multimedia technology to protect users. In this chapter we look at online multimedia sharing and discuss a selection of multimedia algorithms that can help to protect user privacy.

The success of privacy protection algorithms depends on their connection to the human and social aspects of privacy. Effective privacy solutions are situated in the intersection of:

- 1) education: educating users on the danger of privacy and providing them with the information they need to make informed privacy decisions
- 2) legal: developing legal frameworks that protect users
- 3) technical: creating technologies that are able to protect data from unauthorized access or misuse.

This chapter focuses on multimedia algorithms, but looks beyond a purely technical approach to privacy. We connect to the human and social aspects of privacy by focusing on algorithms that put control in the hands of users, i.e., of the people who produced the data in the first place and whose privacy needs to be protected. Solutions that give users control help to free them from dependence on external parties such as service providers. Our focus on user control should not be interpreted as detracting from the responsibility of external parties to protect user privacy. To the contrary, parties that collect, hold, and process user data have an enormous responsibility to protect privacy. It is important to understand that we focus on user control because this perspective on privacy is conventionally underrepresented in previous treatments of privacy and big data from a technical perspective.

The chapter is organized as follows. In the remainder of section 7.1 we motivate and define privacy. Next, in section 7.2, we turn to the discussion of what must be done to protect users' privacy. Section 7.3 discusses the particular privacy challenges raised by multimedia, and specifically by big multimedia data. Section 7.4 presents example techniques and algorithms, and finally section 7.6 provides an outlook for the next steps for multimedia privacy research.

7.1.1 The Dark Side of Big Multimedia Data

Anyone who has learned a new skill by watching a video on YouTube, delighted in exploring a faraway place by browsing collections on Flickr, or reconnected with a long-lost friend by sharing photos on Facebook knows how the rise of online multimedia has added functionality and joy to our daily lives. However, along with the appeal and benefits of sharing videos and photos, sharing multimedia is accompanied by risks. An important risk is *cybercasing*, the use of information available online to mount real-world attacks. This term was introduced in 2010 in [1] in order to help raise awareness of rapidly emerging privacy threats of big multimedia data online. Here, we present cybercasing as a motivating example in order to illustrate the importance of privacy.

The discussion of cybercasing in [2] focused on the inference of users' geo-location by people with criminal intent, e.g., people aiming to rob a house when the owner is away. The danger represented by unwittingly revealed geo-location information can be extended to other sensitive information such as family status and health state, which can be inferred by automatic methods using information shared or otherwise available online. The findings of [1] have serious implications. While users typically realize that sharing locations compromises their privacy, they (i) are unaware of the full scope of the threat they face when doing so and (ii) often do not even realize when they publish such information. The threat is elevated by developments that make systematic search for specific geo-located data and inference from multiple sources easier than ever before. The authors of [1] base these insights on a summary of the state of geo-tagging as of 2010, an estimate of the amount of geo-information available on several major sites, including YouTube, Twitter, and Craigslist, and an examination of its programmatic accessibility through public APIs. The magnitude of the threat of cybercasing is illustrated with set of scenarios demonstrating how easy it is to correlate geo-tagged data with corresponding publicly available information for compromising a victim's privacy. The investigations in [1] were able to find, for example, private addresses of celebrities as well as the origins of otherwise anonymized Craigslist postings. Additionally, further work [3, 4] has shown that it is possible to predict users' home, work and vacation locations using geo-tagged posts on Flickr, Twitter and YouTube.

Protecting users' privacy defends users against cybercasing and against other unintended ill-effects of information misuse. Now that we have highlighted the importance of protecting user privacy with the example of cybercasing, we turn to examine in more detail how privacy is defined.

7.1.2 Defining Multimedia Privacy

Opening a dictionary provides a definition of privacy common in every day usage. Merriam Webster,¹ for example, lists the following two definitions of privacy:

¹ <https://www.merriam-webster.com/dictionary/privacy>.

A: *the quality or state of being apart from company or observation: seclusion*

B: *freedom from unauthorized intrusion one's right to privacy*

The dictionary definition provides a natural connection to our everyday experience of the importance of privacy. As human beings, we need sleep as a physical necessity. In addition to sleep, we also need seclusion. The need for seclusion is reflected in the fact that we seek moments away from the broader circle of other people around us. As humans, we seclude ourselves to relieve ourselves or while engaging in intimacy. Seclusion provides us with time away from observers.

A possible explanation for the physiological and psychological importance of seclusion to humans is the following. Independently of whether observers pose a threat or not, observers require us to maintain a readiness state. Even if the readiness state involves only low-level readiness to comply with social conventions, for example to give a polite response to a greeting, perpetual readiness can wear us down. This view is echoed by the book *Understanding Privacy*, which states, "Privacy protections are responses to problems caused by friction in society" (p. 76). If seclusion is respite from the burden of social interaction, the parallel between sleep and seclusion can be extended further. Like sleep, privacy by way of seclusion is needed for the body and mind to be able to restore themselves. However, the parallel also hints at reasons why privacy is difficult to understand. Like sleep, seclusion is a state that is desirable in the right quantity, but becomes undesirable, or even dangerous, in excessive quantities. Further, each individual has different needs for both sleep and seclusion, and sometimes these needs are only apparent with long-term deprivation. Finally, like sleep, seclusion is a phenomenon that is currently not yet well understood, despite its obvious critical importance for human health and well being.

We are not experts on either sleep or seclusion, and make no claims on the underlying nature of either here. Rather the point that we make is that privacy is an inherent human need, and the fact that we do not understand it completely should not discourage us, but rather provide further motivation to study it. In order to gain insight on how to study privacy related to big multimedia data, it is necessary to go beyond the dictionary definition of privacy and look at more formal characterizations.

The authors of the book *Privacy and Big Data* [5] identify three basic types of privacy, which we use as the basis for the following definitions.

- Physical privacy: the state of being free from intrusion into your personal space, including your possessions and your own body.
- Information privacy: the state of control over information about you that is collected, stored, processed, or shared.
- Organization privacy: the state of secrecy used by companies and governments to hide their activities from competitors and enemies.

When privacy is discussed in the context of big data, the focus is on *information privacy*. For example, the distinction between information privacy and other sorts of privacy is made explicit by [6]. In the context of multimedia, information privacy can also be viewed as a sort of seclusion. Instead of being out of range of observers, information privacy requires withdrawing from the view of the camera. Unlike observers, however, the camera records, allowing a person to be observed and re-observed endlessly. The persistence of multimedia creates a new type of friction: a person needs to maintain a state of readiness to clear up misunderstandings that arise when recordings are viewed out of context or juxtaposed in a way that creates a misleading perspective. Information

privacy must protect users from the myriad of mirrors that is online information. These mirrors confront the user with distorted, disorganized, and contradictory views of themselves. Without information privacy, nothing protects the user from the wear and tear of the effort of maintaining their identity and reputation in the face of these fragmented views.

Information available online can also lead to damage that is more focused than wear and tear. Specifically, information used in the wrong way by the wrong person can lead to dangers such as the cybercasing attacks discussed above. Safeguarding users' information privacy in cyberspace means protecting users from real-world harm.

Here we cite some other helpful definitions of information privacy as a starting point for further discussion of the complexity of privacy. The multimedia privacy tutorial at ACM Multimedia 2016 [7] mentioned [8], who define information privacy as "The claim of individuals, groups, or institutions to determine for themselves when, how, and to what extent information about them is communicated to others." Further, work in the area of privacy for recommender systems [9] follows [10] in defining information privacy as "an individual's claim to control the terms under which personal information—information identifiable to the individual—is acquired, disclosed, and used."

There are two aspects where these definitions fall short. First, they fail to emphasize that there are two types of information that users have claim to control: explicit and implicit information. Explicit information is consciously known by the user. For example, the user must have the ability to control images of his children. Implicit information is information that is not consciously known, but is derivable from multimedia content in some manner, i.e., by an intelligent system (trained on aggregated data) or by a human expert. For example, the user must have the ability to control information on the health state of his children. He himself might not see that an image reveals that his child has curvature of the spine. However, such a condition is noticeable to an expert. A definition of information privacy must encompass the claim to control both explicit and implicit information. Second, these definitions fail to consider the issue of inaccurate information. The danger of inaccurate information is acute with intelligent systems. An intelligent system might analyze the people who a user associates with, as depicted in the photos he posts online, determine that they have committed traffic violations, and conclude that the user represents a risk and should not be offered a job requiring driving. There is no external reference by which to judge if this decision is accurate or fair to the user. The aggregation might have included an error, and it might also be wrong to assume that someone is a bad driver because their friends are. A definition of information privacy must encompass the claim to the control of purported information about the user, independently of whether this information is correct.

In order to address these two aspects, a broader definition of information privacy is needed, such as the one provided in [11], which defines information privacy as "Practically securing the implications of communication". In other words, we can recognize that privacy has been protected when we have protected users from unintended consequences that result from sharing or interacting with multimedia online.

The fact that this definition uses the word "securing" raises the question of the relationship between privacy and security. Wikipedia defines information security as "...the practice of preventing unauthorized access, use, disclosure, disruption, modification, inspection, recording or destruction of information" [12]. From this definition we

can see that the focus of information security is on protecting information. Although protecting privacy is also about protecting information, it does so with a different focus. Protecting privacy is about protecting people, not only information. The authors of [11] point out explicitly that the goal of their privacy research is not providing a secure communication line between communicating parties, but rather ensuring that information that is available to the public conveys only what the person producing that information intended it convey. The focus on people also differentiates privacy from confidentiality. As formulated by [9], confidentiality is related to keeping specific pieces of information secret, whereas information privacy focuses on protecting the individual who the information is about.

Achieving the goal of successfully protecting people requires a collaboration between the people-focused view of privacy and the information-focused view of security. Here, we list several aspects of the people-focused view of privacy that are easy to overlook if too much emphasis is placed on the information focused-view.

- First, protecting a person's information privacy has two facets: it must prevent unintended implications of information sharing by that person or about that person, and it must ensure that the person has the perception that his privacy is protected. For multimedia privacy, the topic of user perceptions of privacy is addressed by [13].
- Second, privacy has a subjective component. For any given piece of information, one user may be comfortable with having that piece of information public and another may not. In the case of multimedia privacy, it has been observed that users differ in the kinds of photos that they would like to keep private [14]. Users' individual judgments may or may not be consistent with more objective assessments of risks. Subjective judgments have a different status depending on whether users are able to understand and reason about risks.
- Third, privacy is contextual. Under one set of circumstances sharing of certain information may be seen as acceptable, but under another it might constitute a privacy violation.
- Fourth, privacy protection must be robust in the face of information accumulation and improvement of information inference technologies (i.e., intelligent systems). A user's privacy may be protected for today, but that protection must extend into arbitrary futures. In the case of cybercasing, this means that information that a user shares today must be prevented from contributing to future prediction of his geo-location by malicious parties.
- Finally, as already mentioned above, information privacy must be protected whether or not the user himself is aware of the information to be protected.

We also mention another important people-focused concept in privacy, namely, *intentional privacy*. Here, we follow the definition used in [15]: "intentional privacy – the right of the individual to prevent or forbid further communication of observed events of exposed features (e.g., publishing photos or video footage)". This definition can be subsumed under the "implications of communication" mentioned by the definition of informational privacy above. However, discussing it separately allows us to correct a frequent misunderstanding. Often privacy is approached as a "barn door" problem. In other words, people assume that once a user has shared or given permission for the use of information "anything goes" and there is nothing to be done, just as it is senseless to close the barn door after the cow has already escaped the barn. However, "intentional privacy"

clearly spells out that it is the intention of the user that is important. The user has no obligation to state intent explicitly ahead of time. Instead, as pointed out by [13], the implicit assumptions of users concerning how their data will be used must be respected. The importance of the scope of sharing is underlined by [9], who state, “Privacy involves keeping a piece of information in its intended scope” (p. 269). This scope comprises the user’s intended audience, the types of usage that the user intended to allow, and the intended lifetime of the information.

This section has introduced a definition of privacy and discussed its intricacies. We have seen that the successful protection of privacy requires protecting many things, in many ways, under many circumstances, and under changing conditions. It is inevitable that some researchers will view privacy as impossible or too complex. However, instead we wish to encourage the view that the complexities of privacy are exactly what makes privacy such an interesting research field, and that they should have the effect of inspiring new and creative solutions. Researchers who feel that there is “no cure” for privacy threats should reflect on the medical world. Medical researchers do not abandon research on a disease because it seems difficult or impossible to cure. Instead, they try to understand it and also to prevent its spread. As multimedia researchers, we need to understand privacy and prevent the spread of technologies that threaten it. In the remainder of the chapter we turn to discussing technical solutions for protecting users.

7.2 Protecting User Privacy

In this section we provide a more concrete description of what we need to protect when we are protecting user privacy.

7.2.1 What to Protect

What we protect about users breaks down into two categories, which we define following [15]:

- 1) Personal identifiable information: personal information that makes it possible to identify someone.
- 2) Personal information: any information relating to a person.

The distinction is also formulated as identity vs. attribute disclosure. Here, we provide definitions following the discussion of [16]. Disclosure takes place when available data is used to accumulate knowledge that can be potentially misused. Identity disclosure happens when a link is made between knowledge and a particular person. Attribute disclosure takes place when new knowledge is accumulated. We mention this distinction for completeness, and do not discuss it further here.

Instead, we take a closer look at personal identifiable information and personal information. Personal identifiable information is described as a set of identifiers. Specifically, [15] lists three types of identifiers:

- 1) biometric identifiers: permanent characteristics that differentiate a person from other people (including voice and gait)
- 2) soft biometric identifiers: differentiating characteristics that are less permanent (e.g., height and weight) and
- 3) non-biometric identifiers (including hairstyle).

A great deal of research has been carried out on eliminating identifiers from multimedia content, a field which is referred to as de-identification. The topic of multimedia de-identification is covered in depth in [15]. Key areas include the de-identification of faces, gaits, and body silhouettes, as well as gender, age, race, and ethnicity. Open challenges include the visual acceptability of the resulting multimedia content after the information has been removed, and also the ability of systems to operate in real time, as is necessary for surveillance systems. For de-identification algorithms it is important to remember the context of the judgment. For example, a person's face could be de-identified, but that person would still be identifiable in a photo to a friend who was present at the moment that the photo was taken. Further, [15] mentions the problem of "pairwise constraint identification", which means that if a person's face is de-identified in a photo, another photo taken in the same context can serve to identify the person due to other matching factors such as hairstyle and dress.

In the rest of the chapter we concentrate on personal information. As discussed in section 7.1.2, information falls into two categories: information of which the user is aware and information that the user is not aware of. We further highlight the difference between information that is inferable by inspection of a single piece of multimedia content, i.e., a person has visited a particular bar, vs. information that requires aggregation of information across multiple pieces of content, i.e., a person visits a particular bar every evening. Further, multimedia content can convey information by virtue of something that it does not show. For example, someone might be shown driving a car without their glasses on, which is illegal if their driver's license require them to wear glasses.

Personal information that must be protected is referred to as *sensitive information*. Although the exact nature of which information is sensitive has a certain dependence on context, certain information is always considered sensitive. From the legal perspective, the General Data Protection Regulation (GDPR) of the European Union lays out a set of rules to protect data. Here, personal data "... can be anything from a name, a photo, an email address, bank details, your posts on social networking websites, your medical information, or your computer's IP address."² In general, deciding what needs to be protected is part of the overall problem of developing privacy protection. Researchers must pro-actively seek to understand what must be protected, rather than expect to be handed a list of sensitive personal information.

7.2.2 How to Protect

We begin our discussion on how to protect users with a broad view of the connection between privacy and multimedia research. Recall that section 7.1 mentioned the importance of developing privacy solutions in the overlap of three areas: (i) education, (ii) legal, and (iii) technical. The broad view reminds us that not only do our solutions exist in this intersection, we ourselves also inhabit it. Concretely, this means that we should understand ourselves not only as researchers developing technology, but also as actors promoting education and upholding legal protection. We now discuss our education and legal roles briefly in turn.

Many researchers active in the area of multimedia research teach, supervise students, mentor junior colleagues, and lead labs. The students and early-career researchers that

² http://europa.eu/rapid/press-release_IP-12-46_en.htm?locale=en.

we work with today are the computer scientists who design the technologies of tomorrow. The success of future privacy protection technology starts with the education of those who will create it. Looking backwards, [17] points out that if privacy had been given serious thought when the Internet was first developed, today we would be facing less of a serious problem. Teaching students about the importance of privacy early will give them the motivation and the skills necessary to ensure that the technological developments that they contribute to incorporate privacy in their design, rather than as an afterthought. Further, we point out that the contribution of computer scientists with technical knowledge of data science, multimedia, and computer networks are needed to further the development of curriculum materials in order to teach students about privacy. A key example is the teaching privacy initiative, described in [18].

Researchers also support the legal system in protecting users' privacy by conscientiously following the procedures of informed consent. When we collect data from users for the purpose of developing a new algorithm, it is important to ensure that users are informed and agree to their data being collected, and to the purpose for which we are using it. Sensitive user data should be stored safely, should not be shared beyond the group designated in the informed consent, and should be destroyed after use. Destruction of data has two purposes. First, it prevents data from accidentally falling into the wrong hands due to neglect or a data breach. Second, it prevents what is known as "function creep" (cf. [15]), the effect that slowly over time, due to forgetfulness or the pressure of producing research results, data are put to uses for which they were not originally intended.

It is important to keep in mind that protecting users' privacy might be the most time-consuming aspect of any particular research project. An exemplary paper is [19], which carries out a study of young people's nighttime habits by studying photos that they take while going out in the evening. The participants who contributed photos are recruited via a recruitment campaign, which was approved by the ethics advisory board of the city. The purpose of the study is explained to the participants, and they provide informed consent. The data collected from the study is used only by the researchers for the particular purpose of the specific research, and it is not shared beyond the research group. When using data collected online, multimedia researchers should keep in mind the concept of "intentional privacy". Users shared their data online with a specific intent, and had no way of anticipating the use that it would be put to by researchers. Researchers should not assume that it is acceptable to carry out any experiment that they can conceive of using data found online. Rather, they should consider the harm that could come to individuals by being singled out by multimedia algorithms that predict information that those individuals could have no way of imagining was possible.

Although purely legal aspects of privacy are out of the scope of this chapter, we do find that it is important for researchers to be aware that privacy holds the status of a fundamental human right. The Universal Declaration of Human Rights (UDHR),³ proclaimed by the the United Nations General Assembly in 1948, includes privacy as a right to be protected along with other fundamental human rights. The right to privacy is encoded in the legal systems of different countries in different ways. However, in general, researchers should realize that their contributions to protecting users' privacy are part

3 <http://www.un.org/en/universal-declaration-human-rights/>.

of a larger picture. By protecting users' rights to privacy, they strengthen the protection of their own rights. Next, we turn to technical solutions for protecting users' privacy.

7.2.3 Threat Models

In order to develop technical solutions, we must first be able to formulate privacy problems. Upon first consideration, many researchers feel that protecting privacy is "mission impossible". The following two questions illustrate the tension between researchers, perceptions of what types of problems can be considered "solvable" and what is necessary in order to protect privacy.

How is it possible to protect users who do not seem willing to stop sharing multimedia online and publicly? Research in privacy requires not laying the blame at the door of the user, but rather providing tools and technologies to protect users as they engage in their natural behaviors and go about their activities. A frequently cited parallel to online sharing is transportation. Asking people to stay home is not a viable solution to protect them from automobile accidents. Likewise, asking people to stop sharing is not a viable solution to protect them from loss of privacy. Just as automobile safety took years to develop, and is also an ongoing target of much research attention (i.e., in the area of self-driving cars), researchers must direct concerted attention to privacy in order to develop creative and effective solutions.

How is it possible to demonstrate progress in protecting privacy, when the privacy problem is never really solved? Research in privacy requires realizing that privacy solutions are not absolute. Instead, privacy problems are like security problems: they can also be solved with respect to a definition of a model. A frequently cited parallel is protecting a house. A house can be protected with locks and bars on the windows and doors. These protections will prevent a person with a standard set of implements from entering. However, someone with a backhoe can easily make a hole in the wall and walk right in. The house is considered secure because of the very small probability that someone with a backhoe will want to enter. When we secure a house, we secure it with respect to a model that does not include the possibility of a backhoe.

These examples illustrate the importance of being able to formulate a concrete privacy problem in order to be able to move forward with developing solutions. The basis for the description of privacy protection problems is a *threat model*. A threat model is a detailed description of the situation in which the solution is intended to provide protection. The model characterizes the source of the threat (adversaries) and the opportunities for threat (vulnerabilities). A classic definition of a threat model from the security literature was presented by [20]. Here, we touch on the main points and highlight some respects in which a model for privacy must extend the standard threat model.

The threat model starts with a characterization of the *adversary*: the person or entity who poses the threat. The adversary has multiple dimensions that must be taken into account. First, the objective: what is the goal or the purpose of the adversary? Second, the degree to which the adversary has or can obtain access to critical data. Third, the resources at the adversary's disposal (e.g., algorithms and computational resources). Fourth, how willing is the adversary to invest those resources given the likelihood of success or failure. Next, the threat model characterizes vulnerabilities. Here, it is critical not to consider only the immediate system and/or data, but the surrounding ecosystem of data sources. Finally, the threat model describes countermeasures. The only countermeasures that must be considered are the ones that correspond to the objectives of the

adversary, and the type of attack the adversary is willing and able to mount. In order to be viable, a countermeasure must fulfill a series of non-functional requirements, including being feasible to implement and easy to use. While outlining these steps in the threat model, [20] also mentions that there are three steps to any attack: (i) plan the type of attack, (ii) gain the necessary access, and (iii) execute the attack. Countermeasures can be effective at any of the three steps.

This threat model was designed for security applications. As previously mentioned, information security is oriented to protecting information and is not focused on protecting users. For this reason, the threat model does not directly transfer to the case of privacy. Here, we discuss some additional factors that must be taken into account to study privacy.

Part of securing a system is determining who to trust. Security conventionally involves one or more trusted parties. Privacy, on the other hand, must be protected in cases where no one can be trusted. The user sharing information might be irresponsible, the friends he shares it with might thoughtlessly share it on, the system that he uses to share might be attempting to target him with unwanted advertising or might be breached. In short, there may not be a single trusted entity in the entire setup, including the user himself. The threat model approach dictates that each of these possible issues be enumerated and their significance estimated. If a privacy solution does not address all of them, it will have still made an informed choice on what not to address, and its limitations will also be explicitly stated, helping to ensure that it is not inappropriately applied.

The threat model states explicitly that countermeasures must be viable, with ease-of-use being a key criterion. Security conventionally involves expert users who are already convinced of the need for security. Privacy protection measures must be easy to use for non-specialist users. Moreover, if the user is not convinced about the importance of protecting privacy, the measures must contribute to educating the user and motivating their use. If a privacy protection algorithm is created, but users do not adopt it, then it might as well not exist. The threat model approach dictates that countermeasures be assessed for the potential for user adoption before they are even developed.

In this section, we have discussed what we must protect when protecting privacy and how we should proceed. Specifically, we discussed the broad view of how multimedia researchers contribute to privacy and also the details of how to formulate a privacy problem. Next, we move on to discussing privacy and privacy threats specifically with respect to big multimedia data.

7.3 Multimedia Privacy

Protecting multimedia privacy involves securing the implications of users' multimedia sharing and consumption behavior. Users should be free to produce, share, and consume media without suffering or fearing unintended consequences.

7.3.1 Privacy and Multimedia Big Data

Tackling the challenges of multimedia privacy involves understanding why not just multimedia data, but, specifically, *big* multimedia data pose a threat to privacy. The

authors of [5] clarify the nature of the debate on privacy by stating that it is "...a debate about the collection and use of our personal information from a commercial and political standpoint" (p. 2). They highlight two particular negative implications of big data for privacy. First, big data is not only big, but it is also drawn from multiple sources. By automatically matching bits of data from different sources, algorithms can infer information about people. This information was not available as long as the sources were scattered and not easy to access. Second, big data has the ability to build a complete picture of an individual. Privacy violations continue to also involve the revelation of individual sensitive facts. However, in the age of big data, it is now possible that a privacy violation reveals a complete picture of an individual's life and activities. In *Data and Goliath: The Hidden Battles to Collect Your Data and Control Your World* [21], it is argued that today's technologies enable nothing short of mass surveillance.

Other negative implications are related to the fact that standard procedures and checks become costly, if not impossible, to carry out. From a legal standpoint, [22] points out that standard procedures of gathering consent become unmanageable for big data. Consent is often reduced to whether or not the users from whom data is being gathered clicked "I agree" rather than whether they genuinely understood the purposes to which they are allowing their data to be put. This situation is different from a standard data collection context, for example during a medical trial, in which informed consent is gathered individually, and opportunities exist for clarifications and questions.

Further, as a data sets grows bigger and bigger, it becomes less and less feasible to check it for accuracy. In fact, most big data applications use data in the state in which it was collected, and make no attempt at quality control. Multimedia collections are particular difficult to check for quality. Researchers regularly work with collections of video so large that it would be humanly impossible to verify them by linear watching. Further, multimedia is often accompanied by metadata. Even if the media content can be verified, the association with the metadata might be wrong (i.e., a YouTube video that has been uploaded with the wrong title).

Big data applications prefer to use data sets that contain every existing, available data point, the so-called $N = \text{all}$ of big data. If every single data point is used, no data sources are left in which the reliability of the data can be checked. It becomes very easy to treat the data set as "reality", since nothing is known of reality beyond the data set.

Independently of the verification of big data, the question of verification of the *output* of big data algorithms arises. The discussion in [6] highlights two important privacy concerns: first, the reliability of correlations discovered in big data, and second, the trustworthiness of the interpretation of inferences.

Finally, it is important to realize that the commercial value of big data gives rise to forces that make it increasingly difficult to protect privacy. Collection and exploitation of user data generates profits. With profit comes the ability to buy the infrastructure needed for large-scale data processing, which in turn generates more profit. Protecting users' privacy does not generate profit. However, research developing privacy-protection technology also requires expensive infrastructure. The picture is different from the "small data" era in which data fit on a standard desk-top hard drive. Unless business models can also be linked to privacy research, the playing field will remain uneven.

In the field of multimedia, this means that entities that can afford infrastructure will continue to process video, images, and audio for profit, with little regard for protecting those people who produced them.

On top of the force created by profit, another force that makes it difficult to protect privacy is industry lock-in: the fact that big data needs big infrastructure locks the advantages in to players that have the resources; those players can, in turn, offer services that lock in users. They have little to no incentive to develop new business models that would embrace privacy since they face no serious competition. The result is a reinforcement of the dominant idea that a commercial company can collect and store users' data indefinitely. There is no burden of proof that the value provided to users in turn is near the value that users would claim, were they fully aware of the privacy risks. The effects of lock-in also make it difficult for researchers interested in studying privacy to be able to access the data that is necessary to develop and test algorithms effective at large scale.

The picture that emerges is that big multimedia data poses unique and pressing challenges for multimedia privacy. At some time in the past, multimedia privacy could mean adding a black bar over the eyes of a person depicted in a picture published in a magazine. Clearly, it is not enough to simply scale-up techniques for privacy protection effective with small-scale data. Instead, new technologies must be developed that make possible the protection of users from new threats that arise from big multimedia data.

7.3.2 Privacy Threats of Multimedia Data

We have seen that the information contained in big data can either be explicit (users are aware of the information) or implicit (users are unaware of the information). The two categories of information hold for general data, as well as multimedia data, in particular. Here, we turn to the question of the particular potential of multimedia data to pose privacy threats from which users must be protected. Previously, multimedia data has been described as having two privacy levels [13]. The primary privacy level encompasses the ostensive information in the multimedia content, and especially the information that is intended to be communicated. The primary privacy level of a lecture video, for example, is the topic of the lecture. The secondary privacy level encompasses the social/psychological characteristics of the data, which are the necessary, but not necessarily intended, by-product of its production. Following the discussion in [13], we list the following examples:

- 1) textual cues: style, vocabulary size, mastery of grammar
- 2) verbal cues: tone, accent, enunciation, projection of confidence
- 3) visual cues: manner, gestures, dress, projection of confidence.

These examples support the argument that multimedia data is more privacy sensitive than other forms of data (documents) since the secondary privacy level can more easily escape the notice of the user. Next we turn to discuss a set of examples of current technologies that represent privacy threats in various sorts of multimedia data.

7.3.2.1 Audio Data

Current open-source technology can be easily repurposed to deanonymize consumer-produced videos just based on their audio tracks. This ability was demonstrated in 2011

by [23]. The authors describe an approach that utilizes audio to link the uploaders of videos based on the audio track of the videos. Using a subset of the MediaEval 2010 Placing Task's⁴ Flickr video set, which is labeled with the uploader's name, they conducted an experiment with a similar setup to a speaker identification experiment. Based on the assumption that the audio might be matched in various ways (speaker, channel, environmental noise, etc.), they trained the open-source speaker ID software on the audio tracks of the Flickr videos. Note that since the selection of videos was essentially random, the audio track can contain any types of sound. They obtain an equal error rate of 36.7% on 312 videos with 11,550 trials. The result is interesting for several reasons. It first shows that even highly tuned systems, like current speaker ID systems, are generic enough to be "abused" for a different task. Second, it shows that random Internet data is not nearly as random as one might think. A speaker ID system can be used to link independent personas. In other words, it is not safe to use different user names to keep sets of videos distinct.

7.3.2.2 Visual Data

With the rapid advancement of visual analysis technology, it has become possible to extract much useful information from images and videos such as objects (what is there) and events (what is happening). Face recognition technology has matured to the point that it is included in commercial software. Multimedia data is often shared online with location information. However, recently people have become more conscious and careful about the level of information released in each image. More people are consciously switching on and off geo-tag labeling when posting to social media and do not turn on the GPS geo-tag function when they are near everyday places such as home or work [4]. However, turning off the GPS will not necessarily solve the problem. What most people are not aware of is that when a group or stream of images are collected even without geo-tag, attackers can still learn about the target much more than expected. Visual geo-location estimation can automatically predict the location at which an image was taken, and recent years have seen it becoming more mature [24]. For instance, a search-based algorithm [25] can geo-locate 7.5% of the MediaEval Placing Task 2016 dataset within 1 km from the ground truth location. With the sheer amount of data shared online that can be easily crawled every minute, this level of accuracy is already good enough to find real-world targets among them. This information reveals the location of the photographer at the time at which the photo was taken, and also, indirectly reveals where the photographer was *not* at that moment. With a stream of geo-data, which gives a history of a user's location, much more can be revealed: habits, interests, hang outs, where the user works, and where the user lives.

7.3.2.3 Multimodal Threats

Although less work has been carried out on multimodal information inference, research in that direction is developing rapidly. In the future, we can also expect privacy threats to be elevated by the combined use of indicators from different media. The potential *and* danger of multimodal location estimation was pointed out by [2]. The ability of multimodal inference to compromise privacy will be magnified many times as machine learning approaches continue to mature. Imagine a future where multimedia query

4 <http://www.multimediaeval.org/mediaeval2010/placing>.

engines *just work*. It would be possible to search by topic, location, person, camera identity, and time, even if the uploader did not explicitly include such information. If someone would like to steal an expensive piece of sound equipment, he could query for photos depicting that piece of equipment. Then, he could query for other photos that were taken by the same person. Finally, he could query for the geo-location of these photos. If these photos reveal a pattern, a time can be pinpointed at which the owner is unlikely to be home. As described earlier, current technology already makes cybercasing attacks possible. The point that we would like to make here is that with a sophisticated multimedia query engine, simple methods of anonymizing posts and suppressing metadata will no longer be enough to protect privacy.

For all types of threats, audio, video, and multimodal, users are most endangered by technologies whose performance is so advanced that it is no longer intuitive. In [26], a series of experiments was carried out on multimodal geo-locations to compare human with machine performance. In the cases in which machines can outperform humans, they do so in a way that would be nearly impossible to guess for a user not trained in multimedia analysis. For example, using a very large number of images, the automatic approach was able to discriminate mountain scenes in Asia, North America, and South America. A human expert might also have this ability, but a general user would have no reason to believe that mountains in different regions are particularly easy to distinguish. Another example is the role of apparently harmless tags such as ‘iphone’ or ‘3g’. Such tags may be associated with photos taken in any region, the distribution of the tags itself is able to discriminate regions. Finally, it was found that videos of the Shinkansen train leaving Tokyo station could be automatically geo-located due to the distinctiveness of the sound produced. A user listening to the Shinkansen might realize that it makes a particular sound, but has no way of knowing it is that distinctive enough to pick out an exact location.

In the next section we turn our attention to multimedia algorithms that provide users with control to protect their privacy in the face of these threats.

7.4 Privacy-Related Multimedia Analysis Research

First, we look at examples of machine learning techniques that are able to infer information about what is depicted in multimedia content. In contrast to conventional multimedia analysis algorithms, which focus on the literally depicted content of images and videos, these approaches focus on less-commonly studied problems with a connection to educating users and supporting the law in protecting them.

7.4.1 Multimedia Analysis Filters

Recent years have seen the development of multimedia analysis algorithms that are able to infer the privacy status of user photos. There are several possible use scenarios for such algorithms. One prominent scenario is that the social networking application would warn a user that a photo might be private before the user makes the decision to upload a photo. This scenario is interesting because it not only helps to protect the user in the moment (avoiding an uploading mistake of this particular photo) but it also educates the user (raises awareness of what would be considered a privacy-sensitive photo).

Other possible use scenarios include smart cameras that warn a user that a photo might be “too private” before clicking the shutter, and automatic sorting applications.

The first work, to our knowledge, that studied the classification of photos by privacy status was [27] and the accompanying demonstration, PicAlert! [28]. This work conceptualized the privacy of a photo as a generic notion: the privacy status of photos was judged independently of the personal view of the user who took the photo. The photos were crawled from Flickr, and the judges were provided with the following description of what constitutes a private photo: “*Private* are photos which have to do with the private sphere (like self portraits, family, friends, your home) or contain objects that you would not share with the entire world (like a private email)”. It is notable that this definition covers both aspects related to the identity of the people pictured in the photo, but also related to the subject material that the user photographs. In other words, it is clear from this definition that a private photo can be a photo other than one that depicts the user himself in a state, location, or with company that he does not wish to make public. At the time that this work was published, there was a surprising reaction of incredulity that judges make stable judgments of privacy, or that it was possible at all to carry out such classification. However, the classifier performed surprisingly well, both using visual features and using text features. A combination of the two achieved an F-measure of 0.8 (data set size 9402 photos evenly split between public and private images). The PicAlert! work appears to have had the function of allowing the multimedia community to accustom itself to the idea that it is possible to build a classifier to make classification decisions about the characteristics of images going above and beyond their literally depicted content.

Work that followed addressed two shortcomings of the PicAlert! work. First, PicAlert! used data collected from Flickr, so the chance was high that the users who originally took the photos did not consider them private (since they were posted to Flickr). Also, PicAlert! did not go beyond the generic notion of privacy. In [29], a study was carried out that investigated private photos of users together with the users’ own privacy status classification. The data set used was tiny (150 photos), but the work is nonetheless important. The work addresses the scenario of classifying the photos on a user’s device to prevent them from inadvertently being viewed. An accuracy of 80% was achieved using easily-available metadata and image features. The best set of features was quite diverse: latitude, longitude, elevation, Unix minutes, calendar week, weekday, day of the month, important hue, ISO-speed, local acutance, number of faces, and resolution. The paper includes a discussion of the underlying trust model and also analyzes the results in terms of their impact on the user. Private photos misclassified as non-private lead to a privacy violation, whereas non-private photos misclassified as private only lead to inconvenience. Understanding the effectiveness of multimedia filters for privacy requires not only reporting standard evaluation metrics, but also analyzing the implications of these metrics for users using the filters in practice.

The most recent work on the classification of images with respect to their privacy status is [14]. This work used photos taken by users (a total of 1511 photos collected from 27 users) and judged by users. The work neatly circumvents the issue of collecting private photos by only collecting features. An important aspect of this study is that it looked at how a generic privacy model can be adapted to cover a user’s individual privacy needs. The findings verified that users have individual preference for privacy. The finding showed that very few photos labeled by users between 5 and 35 were enough to

substantially improve the performance of the generic model. This result shows promise that a personalized privacy filter can be trained for individual users, requiring very little feedback. The findings also showed that it is important to find the appropriate weight to balance the contribution of the generic model and the user. An important aspect of this work is its emphasis of explainability. Although the best visual classifier directly used CNN features, competitive performance was achieved by a classifier using so-called *semantic features*, the output of a concept detector that detected a large number of different classes, such as “child”, “erotic”, “seaside”, and “groom”. The benefit of using semantic features is that they help the user to understand the decision of the classifier. The result is better informed privacy decisions on the part of the user, and, as already mentioned, a potential educational effect about which types of images can lead to privacy violations.

We finish the discussion of multimedia analysis for privacy by mentioning the connection to the research area of multimedia *user intent* [30]. Specifically, we point to work in the area of predicting *uploader intent* [31], defined as “the reasons that motivate users to upload videos to the Internet”. A set of YouTube videos (1677 total) was labeled by mechanical turk workers, who judged whether the video best could be considered “personal” (uploaded for family or friends), “social” (uploaded for people with a common interest, but not everyone), or “public” (uploaded for a broad audience). Using visual features it was possible to build a weak predictor for these classes. Text features derived from the video metadata outperformed the visual classifier, achieving a weighted F-measure across the three classes of 0.53 (for comparison the dominant class baseline achieved 0.32).

In short, multimedia research has shown that it is viable to create classifiers that can estimate the privacy status of multimedia content and support users in making informed choices for sharing. However, before deploying these classifiers in products, it is important to understand how to teach users to use them wisely. A user should never be tempted to suspend their own judgment when making the decision of whether or not to share multimedia content.

7.4.2 Multimedia Content Masking

Masking approaches to privacy protect users by changing data in order to reduce the privacy risk that it represents. In [16], masking methods are defined as including perturbative and non-perturbative methods (p. 62). Masking via perturbation involves distortion of the original data, while masking without perturbation involves replacing the values of the original data with less specific values. We note that [16] points out that perturbative methods can introduce error, while non-perturbative methods just reduce the level of detail of the data. Here, we focus on perturbative masking methods, especially obfuscation techniques. We follow the definition of [32], who state, “Obfuscation is the deliberate addition of ambiguous, confusing, or misleading information to interfere with surveillance and data collection” (p. 1). The key idea of obfuscation is to produce data modeled on existing data to make the collection of data more difficult to analyze and act on. Obfuscation techniques are discussed in detail by [32], who argue that obfuscation provides people who are not in a position to exercise control over their own data with ways to evade or even sabotage the surveillance. Obfuscation is a two-edged sword, and [32] provides concrete examples of cases in

which it has been used to muddy essential communication during elections, as well as protect people working against a repressive government. It is important to differentiate between obfuscation used destructively and obfuscation used as self-defense. Here, we discuss obfuscation in the service of self-defense since we are focused on multimedia algorithms that provide users with greater control to protect themselves from privacy threats.

Recent work demonstrates the potential of obfuscation to protect information about user location by foiling the content-based geo-location estimation methods discussed above. The study in [33] investigates the impact of common user image enhancements (filters and cropping) on the accuracy of geo-location estimation algorithms. Because the enhancements remove some information from images, the expected result is that filtered photos will be less geographically distinctive and more difficult to associate with the location at which they were taken. The paper reveals that filters have a small, but measurable effect. When image enhancements are applied, performance of both the search-based [25] and classification-based approaches [34] degraded.

The benefit of obfuscation-related strategies can be understood by referring back to the discussion of the viability of privacy protection measures taken to counter the privacy threats encoded in threat models. There we mentioned the importance of measures being easy to implement and also maintaining their protective function in the face of the lack of trustworthy parties. Obfuscation is an interesting technique because it does both. It is easy to understand, and, also, obfuscated data minimizes the danger of data falling into the wrong hands, i.e., due to a data breach. Further, obfuscation also provides users with the possibility of plausible deniability: any of the events that occur in their photo streams, for example, are possibly synthetic events added for the sake of allowing users to blend in.

7.5 The Larger Research Picture

The focus of this chapter is on multimedia algorithms that help users to stay in control. It is important, however, to realize that the research presented here is only part of a much larger picture involving many types of research related to privacy and big data. In this section, we point out some of the major areas that multimedia privacy researchers should be familiar with.

7.5.1 Multimedia Security and Trust

Multimedia security, which seeks to protect multimedia data, has been an important topic in the research community for the last two decades. The development of the field of multimedia security has been largely driven by the needs of industry to protect intellectual property (IP) rights. Work on protecting IP has led to research on specific areas of “organization privacy” (mentioned in section 7.1.2). Many of the technologies developed are currently in commercial use.

Specifically, IP protection has been studied by researchers working in the area of fingerprinting (e.g., [35]), watermarking (e.g., [36]), and “secure watermark detection” (e.g., [37]), namely, proving the existence of a fingerprint or watermark without

revealing it. The impact of this line of research was greater than expected; a new research field, secure signal processing, was born.

In a system where security and privacy are needed, research starts with a definition of the capabilities of the attacker, his goals, and the security assumptions. In other words, a threat model, discussed in section 7.2.3, is needed. For example, in cryptography, cryptographic protocols are assumed to be secure in the presence of computationally bounded adversaries, meaning that the attacker has limited polynomial time computational power. Alternatively, we assume that the protocol is perfectly secure; no matter how much computational power the attacker has, the system is not breakable [38] since all possible solutions are equiprobable and the attacker cannot distinguish any of these. The security assumptions must also include a description of the data that should be protected from the adversary.

Security research distinguishes two types of adversary. The first one is an outsider: the traditional attacker who listens to the communication line passively or tries to modify the interaction between the involved parties actively. Traditional security measures like deploying encryption for secure communication, hashes, MACs and digital signatures for integrity and authenticity are straightforward approaches to defend against outsider adversary. The second type of adversary is an insider: an adversary directly in contact with the data. Insider adversaries are associated with cases where the sensitive data on the service provider side are leaked, e.g., due to a malicious or careless employee. This second model is the basis for a growing amount of research since it captures the notion of the untrustworthy service provider: the customers would like to use the services but do not trust the service provider with their sensitive data.

As already discussed in section 7.2.3, privacy must be protected in cases in which no one can be trusted. This concern is shared across the board by researchers oriented towards security and researchers oriented towards privacy. Even if the service provider does not use users' information inappropriately, it is not possible to assume that the information will never fall into the wrong hands, for example due to a systems breach or to a takeover by another company or a untrustworthy government. Research that looks at different types of trust issues is carried out under the umbrella of privacy-enhancing technologies (PETs) and deploys methods such as anonymization [39, 40], differential privacy [41], and computational privacy, which relies on processing encrypted data [42, 43].

7.5.2 Data Privacy

Multimedia privacy can be understood to be part of a much larger research area on data privacy. Data privacy research is described by [16] as encompassing three research communities: statistical disclosure control, privacy-preserving data mining, and PETs.

In turn, issues of data privacy are embedded into larger issues of protecting the entire pipeline of a system that makes use of user data. The pipeline of such a information system requires privacy protection at different stages:

- Ingoing information (queries): Users queries and query patterns must be protected. For example, for the problem of search in encrypted databases scenarios are addressed in which the service provider as the owner of the data should not be able to see which entry is being queried.

- Outgoing information (delivery): Information about the use of data must be protected. For example, in a content delivery system, the data owner should not be able to see who accessed or consumed which content.
- Stored information (data): Data that contains explicit information about users or data from which knowledge about users can be inferred must be protected while it is at rest. Stored information is the main challenge addressed by data privacy. It must be possible to exploit data, yet still maintain the privacy protection of users.

Two areas of research which look at data privacy in the context of systems are private information retrieval (PIR) [44] and privacy for recommender systems [45]. Here, we focus on recommender systems research. We point out that from the beginning, recommender systems research has looked at ways to keep users in control and has also made use of threat models, two aspects that we cover here. Specific examples are [46, 47], which combined secure multi-party computation and homomorphic encryption in order to create privacy-preserving recommender system protocols under the assumption of an untrusted service provider. Users collaborate to compute intermediate values without a central server, making it possible to do away with the need for a trusted service provider.

In the remainder of this section we look at recommender systems research that has yielded interesting examples of the application of masking methods. Applying masking to recommender system data involves introducing perturbations that protect privacy without affecting recommendation effectiveness. In other words, the prediction accuracy is maintained. We take a closer look at examples of two research directions.

The first research direction is interested in masking the rating data, where ratings are the attribute that needs to be protected. As examples of this type of research, we mention [48], where the authors showed that one of the solutions to the privacy issue in collaborative filtering recommender systems is to apply masking, which modifies the ratings stored in user profiles. The evaluation showed that users can mask large parts of the user profiles (ratings) without significantly decreasing the accuracy of the predictions.

The second direction of research defines specific privacy-sensitive attributes (e.g., gender, health state) that should be protected and demonstrates that prediction accuracy can be maintained even when masking is introduced. An example of this type of research is given in [49, 50]. In [49] a framework is proposed for privacy-preserving recommendations using data obfuscation. It is demonstrated that system performance on the obfuscated data does not necessarily impact the accuracy of the prediction. This type of protection would enable, for example, e-commerce vendors to share information about their users without violating their privacy. In [50], the authors proposed a new method called *BlurMe* that adds ratings to a user's profile with the goal of making it hard to infer the user's gender, while causing a minimal impact on recommendation quality.

This brief discussion highlights opportunities for multimedia privacy research to take a cue from data privacy research, which devotes effort to considering privacy at multiple stages of the pipeline, and also has yielded interesting demonstrations of how masking can be used to protect privacy and for which sorts of situations it is appropriate.

7.6 Outlook on Multimedia Privacy Challenges

In this section we look at the research challenges in multimedia privacy that remain to be tackled, as well as ways in which research must reorient itself in order to tackle them effectively.

7.6.1 Research Challenges

We expect that the future will bring further work on a wide range of areas, for example de-identification of multimedia content, which was briefly covered in section 7.2.1. Here, we focus on research challenges related to keeping control in the hands of the users who produce the multimedia content, which has been the focus of this chapter.

7.6.1.1 Multimedia Analysis

Techniques related to multimedia analysis have a bright future. Here, we mention several areas that would be well served by more research attention than they are currently receiving. First, researchers should take a closer look at mitigation. For example, [17] points out the need for approaches that are able to infer for a given photo which information the user intends to convey (foreground information) and which information is conveyed unintentionally (side information). These approaches should focus on meta-data (Does the user know that the photo he is posting also contains information about the identity of his camera?) and also on the semantic content of the photo (Does the user realize that the wallpaper of the hotel is visually unique?) In addition to educating the user about danger of possible information leaks of this sort, and informing the user of potential leaks before they happen, algorithms should be developed that are able to control the damage of leaks once they have occurred.

Techniques that seek to thwart multimedia analysis by blocking inference of sensitive information are nearly as important as the multimedia analysis themselves. Here, we point to the possibility of developing multimedia storytelling techniques that would be capable of enhancing social network feeds. As mentioned in [32], in 2012 a technique was proposed and given the name Bayesian Flooding.⁵ Its purpose is to hide real-life events in a stream of creative fiction, with the aim of shifting the prior probability that a user belongs to a certain demographic, as calculated by advertisers interested in targeting particular market segments. A key point of this technique is that it needs to be fun. The obfuscated feed should be engaging and not confusing for the user's followers. We will comment further on the importance of user engagement below. The user may choose not to obfuscate certain characteristics, such as his gender or his music tastes, but he might choose to obfuscate subtle information such as his psychological state as a reaction to concerns about inference of psychological fragility.⁶

7.6.1.2 Data

Conventionally, researchers assume that big multimedia data will only become bigger, and that the growth will last indefinitely and without control. Here we point out that this assumption may inadvertently be leading to a lack of interest in research topics focused

⁵ http://www.kevinludlow.com/blog/1610/Bayesian_Flooding_and_Facebook_Manipulation_RD/.

⁶ <https://www.theguardian.com/technology/2017/may/01/facebook-advertising-data-insecure-teens>.

on deriving more use from smaller amounts of data. Developing algorithms that need only a small amount of user data is an important step in helping to protect user privacy. Less data can be collected in order to achieve the same aim.

Recently, the area of recommender systems has seen growing interest in data minimization. For example, in [51] data requirements analysis is proposed as best practice. The argument this work advances is straightforward: just as we avoid developing algorithms with unnecessary computational complexity, we should also avoid developing algorithms that need unnecessary data. Evaluation of a new algorithm should include an analysis which demonstrates that the algorithm uses the minimum amount of data necessary in order to achieve its goal. The goal of a recommender system is defined in terms of one or more metrics that capture the effectiveness of the prediction that it generates. Once this goal is met, the use of additional data is no longer justifiable and should be avoided. The future could see the commercial demand rise for data minimization algorithms. First, as users grow more aware of the dangers of sharing, they will be drawn to services that set themselves apart by their responsible handling of data. Second, new laws (cf. the EU General Data Protection Regulation,⁷ which comes into force in 2018) are coming into force which impose strict data requirements. Companies will find themselves in a position that they need to do more with less data. A small nudge might be enough to push industry to working with less data. After all, algorithms that use less data may also require less computation, saving on computational resources, including energy.

Another opening for researchers is the area of synthesizing realistic data. It has been pointed out in the literature, e.g., by [14], that privacy-related data sets are difficult to create due to the nature of the material. Even if it is not possible to synthesize data representing individual users, data synthesis might be helpful at the ecosystem level. Synthetic or semi-synthetic data could make it possible to better understand which types of data are the biggest risk to user privacy and which levels of aggregation are most dangerous. Further, with sufficiently advanced data synthesis, the amount of user information that must be collected could be reduced—algorithms would only need a minimal amount of data to seed the synthesis process, and the multimedia and interaction data needed for training systems could be home grown from there.

7.6.1.3 Users

In this chapter we have emphasized that protecting privacy is related to protecting information, but that its main focus is protecting users. For this reason, future research will necessitate the investment of time, energy, and resources in the detailed user studies needed in order to ensure that privacy-protection algorithms meet user needs. During the presentation of the threat model in section 7.2.3, we mentioned the importance of privacy-protection measures being viable. Critically, viability includes ease of use, and, we argued, effective privacy-protection measures may actually need to educate users, or provide incentives in order to ensure that they are adopted. It is important that users' sense of fun and feel for aesthetics are not disturbed by technology that protects their privacy. Recall that [33] investigated the ability of naturally occurring user practices of photo enhancement to reduce the accuracy of content-based geo-location estimation. Instead of developing a privacy-protection measure and hoping that users would

⁷ <http://ec.europa.eu/justice/data-protection>.

adopt it, this work starts with a behavior that users already engage in and investigates its potential as the basis for a privacy-protecting technology. We hope that future research will be able to also productively adopt such a tactic. How to employ obfuscation techniques for multimedia privacy without compromising users' quality of experience is an important research question that must be addressed in the future.

7.6.2 Research Reorientation

We have seen that multimedia privacy holds significant challenges. It is clear that the research necessary can be measured at the scale of conferences, projects, and products, rather than at the scale of a handful of papers. In order to make real progress in privacy, it is necessary that multimedia research reorients in order to direct more resources to privacy challenges. With our call for reorientation of research we echo the sentiment of [14] that structuring of an active research community in the area of multimedia privacy is an important next step. Here, we discuss three changes that would support such a reorientation.

7.6.2.1 Professional Paranoia

The security world cultivates an attitude of "professional paranoia" towards scientific problems. This attitude predisposes researchers to look at a system or situation from the perspective of what could possibly go wrong. This concept is lacking in today's multimedia research environment, where the focus is on reasonable or probable scenarios that assume cooperative behavior from all users. The threat model is the key tool, adopted from the security discipline, that allows researchers to make headway on developing algorithms that protect users' privacy in face of the complexity and messiness of the topic. Multimedia researchers would be well served by working more closely with information security researchers, adopting both their tools and their attitudes. One hurdle to overcome before we can achieve such collaboration is the traditional reluctance of security researchers to work with detailed user requirements or to carry out user studies (e.g., as pointed out by [13]). However, security researchers are skilled in projecting themselves into the minds of the adversaries interested in compromising systems, and with enough cooperation with user-oriented multimedia researchers they will naturally also become better at projecting themselves into the minds of the users who need their privacy protected. In addition to professional paranoia an attitude of professional empathy must be developed: whatever seemingly illogical behaviors and practices users might pursue, the multimedia privacy researcher must be willing to devote time and energy to the task of protecting that user's privacy.

7.6.2.2 Privacy as a Priority

Protecting user privacy may well be the most challenging problem currently faced by multimedia researchers. Making privacy a priority requires adapting our assumption that the information world exists on a separate plane and does not interact with the real world. As computer scientists, we are used to being able to abstract away from physical reality. In the digital world, people do not suffer from hunger, disease or cold the way they would in the physical world. We feel confident that we can ignore the physical world. However, this confidence can get us into trouble when we study big multimedia data. When we create multimedia systems, we jump directly to the assumption that we can treat online users as entities abstracted away from physical-world needs. In our rush

to abstraction, we forget that privacy concerns information, and this issue follows us from the physical to the digital world in ways that temperature does not. It is easy to assume that privacy violations, occurring as they do in the realm of information, cannot be dangerous. Although information effects clearly have non-physical properties, we should not assume that an information-related phenomenon like privacy cannot result in physical harm.

The assumption that the information world does not impact the physical world is reinforced by the lack of research on the long-term harmful effects of privacy-deprivation on human well-being. Given the lack of such research, it might first seem logical to assume that privacy research can wait until scientists get around to confirming the harmful effects of privacy deprivation. However, studies on such questions may actually be impossible to formulate in an ethical manner. For this reason, it is important to act now on the basis of our natural understanding of the importance of seclusion for the well-being of human beings, and not wait until it is too late.

It is interesting to note that not all researchers will be equally attuned to the importance of privacy. Anyone who has fallen victim to an attack such as cybercasing will obviously feel a more acute sense of urgency for developing algorithms to address the problem. Also, the chance of encountering a problem with privacy violation increases with age: people who are older simply have longer life stories, more that can be revealed, and more time in which the privacy violation could have occurred. Older people may be underrepresented in the research and development workforce (most obviously because older people retire). It is important that multimedia researchers do not rely exclusively on their own experiences to inspire them to carry out privacy research, but also talk to a large range of other people who have first-hand experience of the ill-effects.

7.6.2.3 Privacy in Parallel

A popular position that researchers assume without reflecting particularly carefully is that geo-location prediction is the “main task” and that privacy protection of users’ photos is only an interesting side branch of the research. This position characterizes the status quo and should not be taken as a research vision. Formulating a research vision is difficult since we are limited in our abilities to predict what the future will bring. However, we do know something about the realities that give rise to the big multimedia data collections that we study: the data are created by users and the systems are profitable to the extent that they can attract and retain users. If our multimedia systems fail to protect users, we lose not only the source of our data, but the entire motivation for creating multimedia technology in the first place. If we seek to be prepared for any future, then we need to be prepared for a future in which the problem of multimedia privacy stands on an equal footing with the problem of multimedia inference.

References

- 1 Friedland, G. and Sommer, R. (2010) Cybercasing the joint: On the privacy implications of geo-tagging, in *Proceedings of the 5th USENIX Conference on Hot Topics in Security (HotSec'10)*, HotSec'10, pp. 1–8.
- 2 Friedland, G., Vinyals, O., and Darrell, T. (2010) Multimodal location estimation, in *Proceedings of the 18th ACM International Conference on Multimedia (MM '10)*, pp. 1245–1252.

- 3 Zheng, D., Hu, T., You, Q., Kautz, H.A., and Luo, J. (2015) Towards lifestyle understanding: Predicting home and vacation locations from user's online photo collections, in *Proceedings of the 9th International AAAI Conference on Web and Social Media (ICWSM '15)*, pp. 553–561.
- 4 Tasse, D., Sciuto, A., and Hong, J.I. (2016) Our house, in the middle of our tweets, in *Proceedings of the 10th International AAAI Conference on Web and Social Media (ICWSM '16)*, pp. 691–694.
- 5 Craig, T. and Ludloff, M. (2011) *Privacy and Big Data*, O'Reilly.
- 6 Wacks, R. (2015) *Privacy: A very short introduction*, OUP Oxford.
- 7 Friedland, G., Papadopoulos, S., Bernd, J., and Kompatsiaris, Y. (2016) Multimedia privacy, in *Proceedings of the 2016 ACM on Multimedia Conference (MM '16)*, pp. 1479–1480.
- 8 Westin, A.F. (1968) Privacy and freedom. *Washington and Lee Law Review*, 24 (1).
- 9 Jeckmans, A.J., Beye, M., Erkin, Z., Hartel, P., Lagendijk, R.L., and Tang, Q. (2013) Privacy in recommender systems, in *Social Media Retrieval*, Springer, pp. 263–281.
- 10 Kang, J. (1998) Information privacy in cyberspace transactions. *Stanford Law Review*, pp. 1193–1294.
- 11 Friedland, G., Janin, A., Lei, H., Choi, J., and Sommer, R. (2015) Content-based privacy for consumer-produced multimedia, in *Multimedia Data Mining and Analytics*, Springer, pp. 157–173.
- 12 Information Security, Wikipedia, https://en.wikipedia.org/wiki/Information_security (accessed 1 December 2017).
- 13 Adams, A. (2000) Multimedia information changes the whole privacy ballgame, in *Proceedings of the 10th Conference on Computers, Freedom and Privacy: Challenging the Assumptions (CFP '00)*, pp. 25–32.
- 14 Spyromitros-Xioufis, E., Papadopoulos, S., Popescu, A., and Kompatsiaris, Y. (2016) Personalized privacy-aware image classification, in *Proceedings of the 2016 ACM on International Conference on Multimedia Retrieval (ICMR '16)*, pp. 71–78.
- 15 Ribaric, S., Ariyaeinia, A., and Pavesic, N. (2016) De-identification for privacy protection in multimedia content: A survey. *Signal Processing: Image Communication*, 47, 131–151.
- 16 Torra, V. (2017) *Data Privacy: Foundations, New Developments and the Big Data Challenge*, Springer International Publishing.
- 17 Friedland, G. and Tschantz, M. Privacy Concerns of Multimodal Sensor Systems, book chapter, to appear.
- 18 Bernd, J., Gordo, B., Choi, J., Morgan, B., Henderson, N., Egelman, S., Garcia, D.D., and Friedland, G. (2015) Teaching privacy: Multimedia making a difference. *IEEE MultiMedia*, 22 (1), 12–19.
- 19 Santani, D., Biel, J.I., Labhart, F., Truong, J., Landolt, S., Kuntsche, E., and Gatica-Perez, D. (2016) The night is young: Urban crowdsourcing of nightlife patterns, in *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '16)*, pp. 427–438.
- 20 Salter, C., Saydjari, O.S., Schneier, B., and Wallner, J. (1998) Toward a secure system engineering methodology, in *Proceedings of the 1998 Workshop on New Security Paradigms (NSPW '98)*, pp. 2–10.
- 21 Schneier, B. (2015) *Data and Goliath: The hidden battles to collect your data and control your world*, W.W. Norton & Company.

- 22 Strandburg, K.J. (2014) *Monitoring, Datification, and Consent: Legal Approaches to Privacy in the Big Data Context*, Cambridge University Press, pp. 5–43.
- 23 Lei, H., Choi, J., Janin, A., and Friedland, G. (2011) User verification: Matching the uploaders of videos across accounts, in *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2404–2407.
- 24 Larson, M., Kelm, P., Rae, A., Hauff, C., Thomee, B., Trevisiol, M., Choi, J., Van Laere, O., Schockaert, S., Jones, G.J.F., Serdyukov, P., Murdock, V., and Friedland, G. (2015) The benchmark as a research catalyst: Charting the progress of geo-prediction for social multimedia, in *Multimodal Location Estimation of Videos and Images*, Springer, pp. 5–40.
- 25 Li, X., Larson, M., and Hanjalic, A. (2017) Geo-distinctive visual element matching for location estimation of images. *IEEE Transactions on Multimedia*, 20 (5), 1179–1194.
- 26 Choi, J., Lei, H., Ekambaram, V., Kelm, P., Gottlieb, L., Sikora, T., Ramchandran, K., and Friedland, G. (2013) Human vs machine: Establishing a human baseline for multimodal location estimation, in *Proceedings of the 21st ACM International Conference on Multimedia (MM '13)*, pp. 867–876.
- 27 Zerr, S., Siersdorfer, S., Hare, J., and Demidova, E. (2012) Privacy-aware image classification and search, in *Proceedings of the 35th International ACM Conference on Research and Development in Information Retrieval (SIGIR '12)*, pp. 35–44.
- 28 Zerr, S., Siersdorfer, S., and Hare, J. (2012) PicAlert!: A system for privacy-aware image classification and retrieval, in *Proceedings of the 21st ACM International Conference on Information and Knowledge Management (CIKM '12)*, pp. 2710–2712.
- 29 Buschek, D., Bader, M., von Zezschwitz, E., and De Luca, A. (2015) Automatic privacy classification of personal photos, in *Proceedings of the 15th IFIP TC.13 International Conference on Human-Computer Interaction (INTERACT '15)*, pp. 428–435.
- 30 Kofler, C., Larson, M., and Hanjalic, A. (2016) User intent in multimedia search: A survey of the state of the art and future challenges. *ACM Computing Surveys*, 49 (2), 36:1–36:37.
- 31 Kofler, C., Bhattacharya, S., Larson, M., Chen, T., Hanjalic, A., and Chang, S.F. (2015) Uploader intent for online video: Typology, inference, and applications. *IEEE Transactions on Multimedia*, 17 (8), 1200–1212.
- 32 Brunton, F. and Nissenbaum, H. (2015) *Obfuscation: A user's guide for privacy and protest*, MIT Press.
- 33 Choi, J., Larson, M., Li, X., Li, K., Friedland, G., and Hanjalic, A. (2017) The geo-privacy bonus of popular photo enhancements, in *Proceedings of the 2017 ACM on International Conference on Multimedia Retrieval (ICMR '17)*, pp. 84–92.
- 34 Weyand, T., Kostrikov, I., and Philbin, J. (2016) PlaNet—photo geolocation with convolutional neural networks, in *Proceedings of the European Conference on Computer Vision (ECCV '16)*, pp. 37–55.
- 35 Caldwell, A.E., Choi, H.J., Kahng, A.B., Mantik, S., Potkonjak, M., Qu, G., and Wong, J.L. (1999) Effective iterative techniques for fingerprinting design IP, in *Proceedings of the 36th Annual ACM/IEEE Design Automation Conference*, pp. 843–848.
- 36 Bianchi, T. and Piva, A. (2013) Secure watermarking for multimedia content protection: A review of its benefits and open issues. *IEEE Signal Processing Magazine*, 30 (2), 87–96.

- 37 Prins, J.P., Erkin, Z., and Lagendijk, R.L. (2007) Anonymous fingerprinting with robust QIM watermarking techniques. *EURASIP Journal on Information Security*, 2007, 20.
- 38 Katz, J. and Lindell, Y. (2014) *Introduction to Modern Cryptography*, CRC Press.
- 39 Weber, R.H. and Heinrich, U.I. (2012) *Anonymization*, Springer Science & Business Media.
- 40 Raghunathan, B. (2013) *The Complete Book of Data Anonymization: From planning to implementation*, CRC Press.
- 41 Dwork, C. and Roth, A. (2014) The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science*, 9 (3–4), 211–407.
- 42 Lagendijk, R.L., Erkin, Z., and Barni, M. (2013) Encrypted signal processing for privacy protection: Conveying the utility of homomorphic encryption and multiparty computation. *IEEE Signal Processing Magazine*, 30 (1), 82–105.
- 43 Dara, S. (2013) Cryptography challenges for computational privacy in public clouds, in *2013 IEEE International Conference on Cloud Computing in Emerging Markets (CCEM)*, IEEE, pp. 1–5.
- 44 Gasarch, W. (2004) A survey on private information retrieval. *The Bulletin of the European Association for Theoretical Computer Science*, 82 (72–107), 1.
- 45 Knijnenburg, B.P. and Berkovsky, S. (2017) Privacy for recommender systems: Tutorial abstract, in *Proceedings of the 11th ACM Conference on Recommender Systems (RecSys '17)*, pp. 394–395.
- 46 Canny, J. (2002) Collaborative filtering with privacy, in *2002 Proceedings of the IEEE Symposium on Security and Privacy*, IEEE, pp. 45–57.
- 47 Canny, J. (2002) Collaborative filtering with privacy via factor analysis. *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '02)*, Tampere, Finland, pp. 238–245, ACM, New York.
- 48 Berkovsky, S., Kuflik, T., and Ricci, F. (2012) The impact of data obfuscation on the accuracy of collaborative filtering. *Expert Systems with Applications*, 39 (5), 5033–5042.
- 49 Parameswaran, R. and Blough, D.M. (2007) Privacy preserving collaborative filtering using data obfuscation, in *Proceedings of the IEEE International Conference on Granular Computing (GRC '07)*, pp. 380–380.
- 50 Weinsberg, U., Bhagat, S., Ioannidis, S., and Taft, N. (2012) Blurme: Inferring and obfuscating user gender based on ratings, in *Proceedings of the 6th ACM Conference on Recommender systems (RecSys '12)*, pp. 195–202.
- 51 Larson, M., Zito, A., Loni, B., and Cremonesi, P. (2017) Towards minimal necessary data: The case for analyzing training data requirements of recommender algorithms, in *ACM RecSys 2017 Workshop on Responsible Recommendation (FATRec '17)*, pp. 1–6.

Part III

Scalability in Multimedia Access

8

Data Storage and Management for Big Multimedia

Björn Þór Jónsson, Gylfi Þór Guðmundsson, Laurent Amsaleg and Philippe Bonnet

Recent years have witnessed an ever-increasing scale and complexity of multimedia collections. At the same time, a variety of big data management techniques and tools have been developed, primarily intended for business and logging data. An obvious question, which we attempt to answer in this chapter, is then: which of these big data storage and management techniques could apply to “big multimedia” applications? We first outline a generic architecture for large-scale multimedia systems, concluding that the main tasks of a big multimedia repository are storing, processing, and serving media. We then focus on each of these tasks in turn, and discuss the applicable (big data) techniques and systems for each group. We conclude with a list of potential research directions towards web-scale multimedia applications.

8.1 Introduction

Historically, storage and management of multimedia have not been major concerns in the multimedia research community. Collecting multimedia data in digitized form for research purposes was extremely hard work and the resulting media collections were tiny in comparison with today's collections. Early system prototypes were therefore only required to handle data that could be stored on a single server and configuring those servers was left in the hands of system administrators. Since then, however, three developments have triggered significant changes in the way media is generated and consumed. First, produced media (such as television shows, movies, and music) is increasingly distributed online in digital form (both legally and illegally). Second, and more importantly, the advent of camera-phones, and later smart-phones, was the prime driver in re-writing the rulebook for media generation. Third, the advent of Internet-based “social” services for storing and sharing this media has completely changed the way media is distributed. Today, almost everyone constantly carries a device that combines a (video) camera and a screen, and media can be instantly shared and consumed over the Internet.

8.1.1 Multimedia Applications and Scale

In recent years we have observed an ever-increasing scale of media collections (e.g., various social collections). Complexity has also been growing, as more information is associated with media items (e.g., online discussions) and automated media analysis tools become better and better (e.g., deep learning). We have now reached the point where very large collections are available for use and analysis. The recent YFCC100M collection¹ represents a massive jump in the size of experimental collections, with almost 100 million images and close to a million videos [1]. This collection is of a similar scale to, for example, the Europeana art collection² with 50+ million digitized artworks, and DeviantArt,³ which hosts a collection of a few hundred million born-digital artworks. The YFCC100M collection is nevertheless far from being a web-scale collection, as Facebook⁴ stores hundreds of billions of media items [2] and YouTube⁵ users upload an estimated 400 hours of video content every minute [3], to give examples. At this scale, data storage, management, and processing become quite difficult: copying collections between storage solutions may take days or weeks, and any processing typically requires distributed computing.

It is interesting to note that while multimedia applications have been studied extensively for decades, most large-scale industry applications involving media are conceptually relatively simple, although their implementation at scale usually requires complex engineering efforts. Very few of the sophisticated methods that the multimedia community has proposed, even those that have considered non-trivial collections, have seen acceptance in industry. Recently, deep learning has gained popularity, partly because the technology is quickly maturing and partly because it allows mapping the content of media to an information retrieval problem, for which scalable solutions have existed for quite some time. Even deep learning, however, is still far from robust (e.g., see [4]).

Many of today's large-scale media collections, however, may harbour information that could be used beneficially, in some cases for financial gain but more often for social wellness, if we can manage to extract the knowledge and insight that they encode [5]. For this purpose, the new field of multimedia analytics, which combines visual analytics with multimedia analysis, has been developing over the last half decade. The goal of multimedia analytics is to produce the processes, techniques, and tools to allow users to efficiently and effectively analyse multimedia collections in order to gain insight and knowledge. While early systems focused on user interaction models at small scale [5], researchers have recently started considering scalability in this area; most recently [6] proposed an interactive analytics process able to learn user preferences and score the 100M images of the YFCC100M collection in 1 second using a high-end workstation.

We believe that as the community progresses towards larger multimedia collections, we must move from multimedia retrieval towards multimedia analytics methodologies, as they involve using a number of techniques to support users in making decisions. From the point of view of system architecture, however, a retrieval system is a subset of an analytics system, and hence in this chapter we focus on the latter.

1 <http://multimediacommons.org/>.

2 <http://www.europeana.eu/portal/en>.

3 <http://www.deviantart.com/>.

4 <http://www.facebook.com/>.

5 <http://www.youtube.com/>.

8.1.2 Big Data Management

Over the last decade and a half—roughly the same time period that multimedia retrieval researchers have spent slowly discovering the need for scale—there has also been a proliferation of “big data” management techniques and tools, which have arisen mostly in the context of much simpler business and logging data. These big data tools can be largely grouped into three categories:

Distributed file systems: These file systems (e.g., GFS [7], HDFS [8]) provide an abstract layer in the computing cloud, which manages distribution and provides availability guarantees for the data collection.

Automatically distributed computing frameworks: These frameworks (most notably Hadoop⁶ and Spark⁷) are used to automate and distribute complex processing tasks, such as machine learning processes or high-dimensional indexing on top of a distributed file system.

NoSQL systems: These systems provide a non-relational data model, along with associated data querying and manipulation capabilities, and usually focus on exploiting distributed computing facilities [9].

Which of these big data storage and management techniques apply to today’s large-scale multimedia collections and applications? Answering this question is not a trivial task, as multimedia systems are complex systems, both in terms of the data stored and the applications served. We therefore start by outlining an prototypical system architecture for “big multimedia” analytics systems. In the subsequent discussion we focus on the requirements of the components of the architecture, as understanding requirements is the key to understanding which technologies can satisfy them.

8.1.3 System Architecture Outline

Figure 8.1 shows an outline of the overall architecture. This architecture is prototypical in the sense that it is not based on a particular system, but rather serves as an abstraction of such systems. As media items enter the system, via either users or administrators, they typically go through two types of processing. First, various metadata are extracted and stored, including operational metadata (e.g., file information, user information, and process information), low-level features (e.g., various high-dimensional descriptors), and high-level semantic information (e.g., output of concept classifiers). Second, various configurations of the media items may be produced for efficient presentation (e.g., thumbnails, video excerpts). The two can, of course, be combined; producing scale-invariant feature transform (SIFT) features [10] for images, for example, requires rescaling the images and the rescaled images may be stored for presentation at various scales.

In a typical multimedia application, most of the query processing and analysis concerns extracted metadata, while the user is mostly concerned with the actual media objects. Many (perhaps most) of today’s media applications are based on keywords associated with, or extracted from, the media items. However, other applications require low-level features (e.g., copy detection), while yet others may make extensive use of the

6 <http://hadoop.apache.org>.

7 <http://spark.apache.org>.

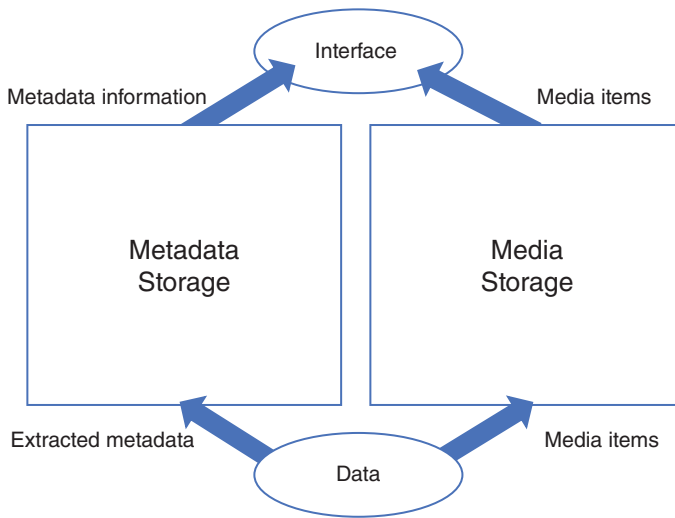


Figure 8.1 Simplified big multimedia system architecture. The metadata is extracted from media items, often using a significant processing effort, and stored in a separate module. While metadata is typically smaller than the media from which it is derived, most of the interaction with the user takes place in this module. The media items are stored in media storage, often in several versions with varying size and quality parameters. They are then served to users for presentation once interaction with metadata storage has determined which media items the user needs.

operational metadata (e.g., mining applications). We consider all the variants of metadata useful and significant to the discussion in this chapter.

Metadata storage is the more complex of the two components, as its interactions with users can involve arbitrarily complex operations, while the typical request to the media storage is for individual media items. (Note that a pure data mining application would only access the metadata storage component.) The next subsection therefore outlines the architecture for Metadata Storage in more detail.

8.1.4 Metadata Storage Architecture

Figure 8.2 shows an outline of the Metadata Storage architecture, which is discussed below. The Metadata Storage architecture is based on a recent development in big data systems, the Lambda Architecture by Marz and Warren [11]. As the Lambda Architecture runs contrary to the traditional database design methodology taught in most universities across the globe, we briefly explain the Lambda Architecture, before presenting the adapted Metadata Storage architecture.

8.1.4.1 Lambda Architecture

Traditional database design methodology focuses on maintaining a single current state of the system the database should model. As an example, if a customer moves, the current address is modified, thus deleting any record of the previous address. This has two major consequences.

First, maintaining this current state consistently involves complex transaction semantics and indexing machinery, resulting in heavy random write patterns to secondary storage. Expanding these transaction semantics to distributed settings for very large and high-throughput databases has proven a difficult task.

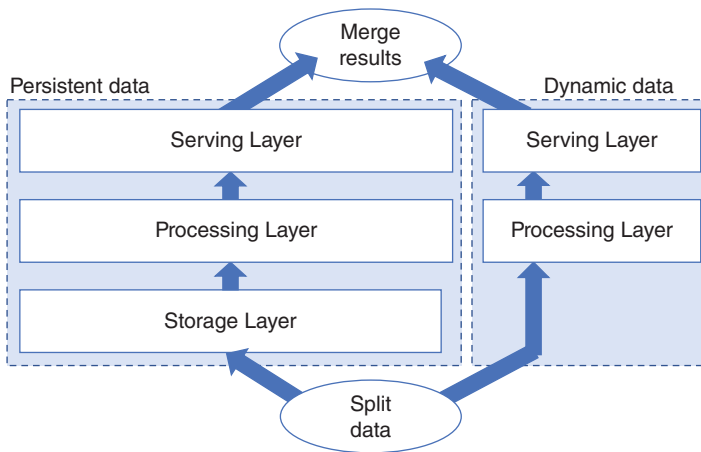


Figure 8.2 Overview of the Metadata Storage architecture (based on the Lambda Architecture [11]). Most of the data is served with the layers on the left side of the figure. The Storage Layer stores all the metadata in the collection. The Serving Layer consists of the (high-dimensional or inverted) indexes to facilitate efficient online query processing. The Processing Layer is responsible for keeping the data structures in the Serving Layer up to date as data is added to the Storage Layer. As the output of the Processing Layer on the left side is never quite up to date, the most recent data is served by the corresponding layers on the right side of the figure, which only serve a very small fraction of the data; once data is represented in the Serving Layer on the left side, it can be removed from the Serving Layer on the right side. In order to work with the two sides of the Metadata Storage architecture, the stream of incoming metadata items must split into two streams which are processed separately, while the results of querying the two sides must be merged before presentation to the user.

Second, in order to analyse historical data, the previous address is actually of importance as, say, purchase behavior may depend on the residence and hence change when the customer moves. In order to maintain historical data, companies have typically employed data warehouses, at significant cost, where historical data from multiple systems is stored and re-organized to be suitable for analysis and mining.

These two issues are addressed by the Lambda Architecture. Rather than asking a single central data server to keep a big data collection up to date at all times, thus creating a potential performance bottleneck, the Lambda Architecture is based on the three key observations and corresponding principles of Table 8.1. These principles, in turn, lead us to the Metadata Storage architecture of Figure 8.2, which is described next.⁸

8.1.4.2 Storage Layer

Turning back to Figure 8.2, the role of the Storage Layer is to store all the metadata in the multimedia collection, with a focus on managing the data volume. As Marz and Warren note, in a big multimedia repository data should not be deleted or updated, only appended,⁹ and the corresponding principle is that all data must be stored in a single data collection which can be maintained by appending time-stamped data. This

⁸ Two notes are in order. First, the terminology of the architecture presented here does not precisely match that of the Lambda Architecture, for easier exposition. Second, while the Lambda Architecture does have its critics and some readers may have objections to it, it is nevertheless a good vehicle for framing the discussion in this chapter.

⁹ Note that deletions and updates may occur, not as part of regular processing, but, for example, to correct the effects of software errors or remove old data deemed no longer relevant [11].

Table 8.1 Key observations and principles on which the Lambda Architecture is founded. We refer to Marz and Warren [11] for details.

Observation	Corresponding principle
Data is typically not deleted or updated, only appended	All data must be stored in a single collection which can be maintained by appending time-stamped data fields
Almost all the data in the collection is old and stale	For stale data, use batch processing to create data views that achieve interactive performance
Recently added data is a very small portion of the collection	For recent data, use a distributed data store with transactional properties to achieve interactive performance

allows avoiding random updates, and also allows a distributed file system to be used as the main data storage solution. Using the inherent redundancy of typical distributed file systems, system resilience is improved.

8.1.4.3 Processing Layer

The Storage Layer is generally unsuitable for interactive query processing since retrieving the data needed to find the most recent address of a single customer, to continue the same example, may require massive scans. The second observation of Table 8.1, however, is that since data is never deleted or updated, but only appended, almost all the data in the master collection is old and stale. For this data, it is possible to use batch processes to create views that are suitable for interactive performance. These views are produced by the Processing Layer of Figure 8.2 and stored in the Serving Layer. The emphasis here is on the throughput of batch processing; by using the scaling properties of automatically distributed computing frameworks, more resources can be applied to the batch process as the data grows.

8.1.4.4 Serving Layer

The Serving Layer implements the interaction with users and their interfaces. The focus here is exclusively on the low latency required for interactive response time, typically achieved through the extensive pre-computation of the Processing Layer. The Serving Layer hosts any high-dimensional indices, inverted indices, and other data structures needed for efficient query processing.

8.1.4.5 Dynamic Data

The third observation of Table 8.1 is that due to the time required for producing the batched views of the Serving Layer, the data that has been added since the latest batch in the Processing Layer started is not reflected in the Serving Layer and must be served separately for real-time processing. This issue is addressed with the layers on the right side of Figure 8.2. The views of the Serving Layer for dynamic data match the views on the left side and must be kept up to date in a transactional manner even in the face of high-frequency updates. Since these views only cover data from the start of the latest batch, however, they only represent a very small portion of the master collection, making it feasible to store them in a distributed data store with transactional properties. Once a new processing batch is completed, data older than the start of the batch can be deleted from the Serving Layer for dynamic data. The part of the collection stored on the Dynamic Data side thus remains small.

8.1.5 Summary and Chapter Outline

The preceding discussion shows that the tasks required to implement the prototypical multimedia system of Figure 8.1 can be roughly divided into three groups:

Storage: The repository must store the full spectrum of the multimedia data: raw data (images, video or audio, typically in several versions); meta-data (including discussion threads) associated with the multimedia items; derived data, such as low-level and semantic features extracted from the raw data; supplementary data structures, such as high-dimensional indices or inverted indices; and learned models used to extract features or semantics from new media files. The emphasis of storage is on managing data volume, as media items are often very large and must be stored efficiently and securely, and metadata may also lead to significant data volume (e.g., storing SIFT features for fine-grained video copy detection).

Processing: The repository must support analysis of the media files, e.g., feature extraction and deep learning, as well as the construction and maintenance of the associated data structures. The emphasis of processing is on scaling the throughput of the computations required to extract the metadata, in particular the low-level features and semantic metadata, and on building the necessary index structures. Frameworks to automate and distribute such complex processing tasks are particularly useful.

Delivery: Typically, the first step in serving media from a big multimedia application is to interact with the metadata storage to understand which items the user might be interested in. Subsequently, the media items of interest must be retrieved and served to the user. The repository must serve all these data files and data structures for a variety of workloads, ranging from the presentation of a single media item to complex queries covering multiple indices. The emphasis of delivery is on the low latency of the system in order to serve the user as fast as possible and maintain the interaction with the user.

In the remainder of this chapter we focus on these three groups of operations and discuss commonly used big data techniques and systems for each group. In section 8.2 we focus on media and metadata storage, in section 8.3 we discuss media and metadata processing, and in section 8.4 we discuss metadata and media delivery. We end the chapter with two case studies in section 8.5 before presenting our conclusions and some research directions in section 8.6.

8.2 Media Storage

In this section we describe storage issues in scalable cloud-based systems. We start by analysing the current status of the storage hierarchy before discussing distributed storage concepts.

8.2.1 Storage Hierarchy

The traditional storage hierarchy is a pyramid of layers representing memory and storage components—cache, RAM, secondary storage, and tertiary storage—attached to a host equipped with computing cores. Fundamentally, the storage hierarchy

differentiates between the following layers, where each layer is orders of magnitude faster, smaller, and more expensive than the next:

Main memory: Transient memory that is byte addressable, i.e., directly accessible from the processor (via load and store instructions). In practice, main memory is composed of several levels of caches embedded on the central processing unit (CPU) and RAM modules plugged on the system bus or, these days, even integrated with the CPU.

Secondary storage: Persistent storage that the CPU accesses through the I/O subsystem. Secondary storage devices expose a block device abstraction (i.e., read and write operations at the granularity of a fixed size block of bytes).

Tertiary storage: Persistent storage that is off-line and accessed on-demand, typically a tape system that is operated manually or via a robot.

The storage hierarchy has been relevant for building balanced systems where computing, memory, and storage performance are aligned. The goal has been to avoid diverging performance trends across layers, thus making it possible to introduce successive generations of hardware systems without changes to legacy software. However, emerging trends are shaking up the traditional storage hierarchy.

8.2.1.1 Secondary Storage

For the last 30 years, secondary storage has been exclusively based on hard disk drives (HDDs). This is no longer the case. In recent years, the growing performance gap between processors and HDDs has led to the introduction of solid-state drives (SSDs) as replacements for disks. SSDs are based on non-volatile memories such as NAND flash, 3DXpoint or ST-MRAM. They offer great performance at an ever-decreasing cost. Today, tens of NAND flash chips wired in parallel behind a safe cache deliver hundreds of thousands of accesses per second at a latency of tens of microseconds. Compared to modern HDDs, this is a massive improvement in terms of bandwidth and latency. Figure 8.3 shows an example of the performance differences between a high-end HDD and a high-end SSD [12].

Recent studies show that today it is cheaper to store a few terabytes on SSDs than on HDDs [13]. The main reason why SSDs exhibit a lower total cost of ownership is that data residing on an SSD can be accessed efficiently by multiple applications concurrently, unlike HDDs where it is recommended to duplicate data to isolate read access patterns from write access patterns and to promote sequential access patterns. In summary:

- While SSDs are smaller and more expensive to buy, they are faster and have a lower total cost of ownership than HDDs.
- For HDDs, sequential access patterns should be favoured as they are at least an order of magnitude faster than random accesses. For SSDs, random operations perform well.
- For HDDs, operation size is not important. For SSDs, small operations are faster.

8.2.1.2 The Five-Minute Rule

A key question in the context of storage hierarchy is where to place data. The closer to the CPU the faster the access, so why not always keep data as close as possible to the CPU? Obviously, a large-scale image collection cannot fit into a processor cache, but it might fit in RAM. Alternatively, a question is whether, given an image collection, it makes sense to dimension the system in such a way that all data fits in RAM.

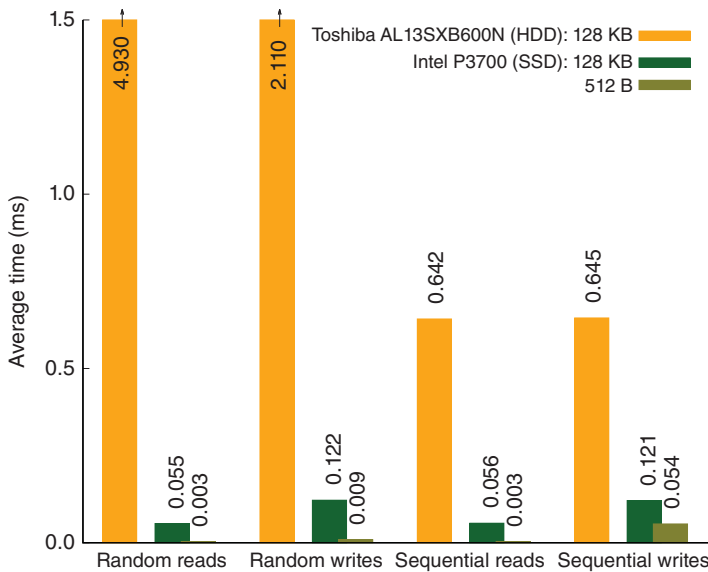


Figure 8.3 Performance comparison of a high-end HDD (600 GB, \$360) and a high-end SSD (400 GB, \$900). The y-axis (truncated) shows the time for each of the operations on the x-axis, in milliseconds. The numbers show that SSDs are faster than HDDs by a factor of 5x to 1000x, depending on operation and data quantity. Source: [12].

While it is only recently possible to consider systems equipped with terabytes of RAM, the question of whether data should be placed in RAM or on secondary storage has been studied for more than 30 years [14–16]. Over this time, the trade-off has remained much the same. If data is accessed often (i.e., the data is hot), then it makes sense to keep it in RAM, but if data is accessed infrequently (i.e., the data is cold) then it should be placed on secondary storage. For many years, the threshold between hot and cold data was 5 minutes: if data was accessed more than once every 5 minutes, it should be placed in RAM, otherwise the cost of holding the data in RAM was higher than the cost of holding the data on secondary storage and transferring it when needed. So, even if an image collection can fit in RAM, it is not economically optimal to always keep it in RAM if a subset of the data is accessed infrequently. A frequently used and randomly accessed index, on the other hand, should be kept in main memory. The exact threshold depends on the ratio between the cost of secondary storage and the cost of data transfer at each time.

8.2.1.3 Emerging Trends for Local Storage

As main-memory size increases, traffic to secondary storage has become dominated by writes. Data-intensive systems now rely on write-optimized data structures such as the LSM tree, that implement logarithmic methods for handing fine-grained updates to secondary storage [17]. Main memories are essentially composed of dynamic RAM modules (DRAM). New technologies improve on traditional DRAM: 3D stacking improves memory bandwidth (at higher production cost and reduced energy efficiency) and hybrid memories improve capacity (e.g., Diablo's Memory1, based on flash-backed

RAM, provides a 256GB DIMM module at a fraction of the storage price and of the energy consumption of DRAM).

Also, persistent memories are emerging [18]. Persistent memories are byte addressable non-volatile memories (NMV), directly accessible from the processor,¹⁰ just like RAM. Different classes of technologies are competing: resistive RAM, ST-MRAM, PCM, and 3DXpoint. Much remains unknown about the actual chips, but it is expected that PCM and resistive RAM will provide high capacity with write speed of hundreds of nanoseconds, while ST-MRAM will provide high endurance at write speed of tens of nanoseconds. Today, packaged ST-MRAM can be bought for \$1.5/MB, which is approximately the storage price of RAM in the late 1990s. However, without insider knowledge it is difficult to predict how NVMs will be priced when they hit the mass market.

The introduction of NVM is a significant disruption of the traditional storage hierarchy as persistence is no longer tied to storage devices. As a result, the modern storage hierarchy includes transient memory (RAM), persistent memory, and various forms of storage devices directly attached to a host or accessible via storage fabric (i.e., a switched or optical interconnect within a cluster rack).

Finally, open-channel SSDs [19] now make it possible to consider application-specific media management, where data placement and I/O scheduling are tailored to the needs of a particular application. Also, a new generation of programmable SSDs¹¹ makes it possible to revisit the concept of near-data processing [20], where application-specific code is executed on the storage device. Defining media management strategies that support efficient processing and serving of multimedia collections is an intriguing avenue for future work.

8.2.2 Distributed Storage

The quest for lower latency, high resource utilization, energy efficiency, and reduced cost is pushing cloud service providers to consider data centers composed of clusters of rack-scale computers. The storage hierarchy of data centers consists of (i) local resources, including main-memory and secondary storage as discussed above and (ii) remote resources, accessible within a rack, across racks within a cluster, or across clusters via network switches.¹² The storage hierarchy of the cloud is thus no longer a pyramid of layers, each orders of magnitude faster, smaller, and more expensive than the next. As Dean [21] described it, the performance and capacity profile is a “bumpy ride” when traversing a distributed storage hierarchy. Distributed resources can be accessed on many devices in parallel over a network bandwidth that is often greater than the capacity of local storage devices, effectively making the distributed option both faster and larger. However, the tradition is to use local resources first and leave the network free for incoming traffic. On public clouds, such traffic is often unpredictable and results in large latency or throughput variability when accessing distributed resources.

The distributed storage hierarchy is typically managed via a software-based file system abstraction layer, such as HDFS [22], which in turn uses, the local disks and

¹⁰ <http://pmem.io>.

¹¹ <https://github.com/DFC-OpenSource>.

¹² The term “fabric” is often used to denote such network interconnects within a data center.

file system of the machines in the cluster. In the remainder of this subsection we describe two fundamental concepts of distributed data systems before describing two distributed file systems in more detail, HDFS and Ceph.

8.2.2.1 Distributed Hash Tables

The fundamental abstraction underlying most distributed file and data management systems is that of distributed hashing. In a distributed hash table, the address of each file or file segment is used as input to a hash function, and each hash bucket is stored on its own server. Hash functions can be chosen in a variety of ways, but mechanisms are put in place to guide read and write requests to the correct server, regardless of where they originate. Often a particular name-node is used for this purpose, but variants exist where that is not necessary.

When redundancy is desired in the data storage, more than one such hash function can be applied to guide the redundant copies to different servers. A common practice is to store three copies of each file, two in relative proximity within the same data center and the third in a different (often distant) data center. This configuration facilitates scalability, fault tolerance, and autonomy.

8.2.2.2 The CAP Theorem and the PACELC Formulation

A key question in distributed environments is how to react when the network is (inevitably) partitioned due to failures, as clients may attempt to read from or write to any partition of the network. The well-known CAP theorem states that in such conditions it is impossible to achieve both consistency and availability of data [23]. Verifying the validity of the CAP theorem is a simple thought exercise: when a data item is updated in one network partition but requested in another, a system that desires to guarantee fully consistent data becomes unavailable, while a system that desires immediate resolution of queries will return data that is no longer up to date.

The CAP theorem only concerns failure scenarios, however, and therefore is not truly useful for the design of distributed database systems. This has led to the more comprehensive PACELC formulation (pronounced “pass-ellk”), explained as follows [24]: “if there is a partition (P), how does the system trade off availability and consistency (A and C); else (E), when the system is running normally in the absence of partitions, how does the system trade off latency (L) and consistency (C)?”

Depending on the application characteristics, systems may choose different strategies. For example, “social” systems typically aim for availability and low latency, at the potential cost of consistency, while systems intended for offline analytics can focus on consistency. Of course, the trade-off between availability/latency and consistency is not binary, as some systems implement so-called “tunable consistency”. In this case N copies of each data object are stored redundantly, each write operation is considered complete when W of those copies have been updated, and each read operation requires determining the most recent value from at least R of those copies. If $W + R \geq N$, the system can guarantee consistency (with some additional quorum protocols [24]), but if $W + R < N$ then inconsistent data may be returned. By varying these three parameters, the tradeoff between latency/availability and the likelihood of inconsistencies may be tuned.

8.2.2.3 The Hadoop Distributed File System

The Hadoop distributed file system (HDFS) is a distributed file system that is implemented as a software abstraction (service) on top of the resources of the underlying

machines. HDFS is the default file system for both Apache Hadoop and Apache Spark (see section 8.3) and the distribution of the HDFS blocks is an integral part of the distributed execution of both systems. HDFS is built for storing large (multi-GB) files and is optimized for a very specific access pattern, namely large bulk streaming reads and append only writes. In-place editing is simply forbidden and random read access is infeasibly slow.

Data stored in HDFS is organized into directories, files, and file blocks. A centralized name-node is responsible for keeping track of files, their metadata, and the location of their content (i.e., block locations), while the remaining machines are the data-nodes that host the HDFS file blocks on their local storage devices. When a file is written to HDFS, it is first registered with the name-node and then its data is partitioned into fixed-size blocks (typically 64MB or 128MB) that are distributed across the data-nodes. HDFS handles failures by replicating each block to multiple data-nodes (the default replication factor is 3).

Note that initially, when copying data into HDFS, the blocks may not be evenly distributed. This will be corrected over time, however, as HDFS periodically runs a balancing service to improve the distribution. The balancing bandwidth is typically kept low by default to avoid interference with cloud-based processing. Also, since writing to HDFS can be quite slow because of the replication factor, a common method to save time on the initial file copy is to write the file to HDFS with a lower replication factor and then change the replication setting. HDFS will then start replicating the missing blocks to other data-nodes until the desired replication factor is reached.

8.2.2.4 Ceph

Ceph is a highly scalable distributed storage solution based on the same underlying principles as HDFS. Ceph is built on top of a fully autonomous distributed object store called RADOS [25] and exposes access to this object store via three interfaces: (i) POSIX-compliant file system, (ii) block device (secondary storage API), and (iii) REST-ful object-store interface that has S3 and Swift compliant API. While the underlying concepts are the same as in other distributed file systems, these three different interfaces make Ceph a versatile system.

8.2.3 Discussion

It can be argued that the trends in local and distributed storage hierarchies are at odds. On the one hand, the advent of SSDs, which can efficiently serve small random operations, allows for much more fine-grained control of secondary storage, while on the other hand the complexity of cloud-based architectures demands large-grained control of the distributed data. As a result, it will be quite interesting to observe the overall development of distributed storage systems in the next (very few) years.

8.3 Processing Media

In the generic system architecture presented in the introduction, media processing occurs in three places.

Metadata extraction: Metadata is extracted from recently added media and inserted to the Storage Layer.

Processing Layer: The batched views (such as inverted and high-dimensional indices) of the Serving Layer, containing all the data from the collection, are updated using scalable batch processes.

Dynamic Data: Newly added metadata is integrated into the dynamic views maintained by the dynamic side of the system.

In this section we first consider metadata extraction, in particular deep learning. We then discuss the distributed computing frameworks that are most suitable for large-scale batch processing, namely Hadoop and Spark. Finally, we discuss stream processing, which is suitable for the maintenance of dynamic views.

8.3.1 Metadata Extraction

As discussed in the introduction, various metadata must be considered for multimedia applications, including operational metadata, low-level features, and high-level semantic information.

Today, using deep learning for extracting high-level semantic information is the most important metadata extraction technique. Deep learning has recently gained tremendous popularity and shown some excellent results for, amongst other applications, image content analysis and classification [26–36]. Much of the recent progress in deep learning is due to successful algorithmic use of massively parallel GPU units and distributed computing; algorithms for deep learning are covered in detail elsewhere in the book.

Caffe [37], TensorFlow [38], and MXNet [39] are commonly used deep learning systems. Distributed computing frameworks (discussed below) can be used in two ways for deep learning [40]: first, when tuning learning parameters, multiple learning sessions can be run in parallel and compared to find the optimal parameter values; second, cloud-based computing is very efficient for applying the learned models and classifiers to large-scale media collections. More recently, however, a library of deep learning pipelines has been added to Spark [41].

8.3.2 Batch Processing

Several researchers have investigated the use of automatically distributed computing frameworks (ADCFs), mainly Apache Hadoop¹³ and Apache Spark,¹⁴ to implement a variety of multimedia-related batch-processing tasks. As these ADCFs do not provide the response time required to implement interactive services for large multimedia collections, the focus of these works has been on background tasks, such as indexing and batched query processing. These tasks focus on the throughput of the processing pipeline, or how many items can be processed per second, rather than the response time.

Guðmundsson et al. [42] proposed common requirements that an ADCF should meet in order to form a good basis for implementing web-scale multimedia services:

¹³ <http://hadoop.apache.org/>.

¹⁴ <http://spark.apache.org/>.

- R1) Scalability:** Scale out with additional computing resources.
- R2) Computational flexibility:** Carefully balance system resources as needed.
- R3) Capacity:** Handle data that vastly exceeds main memory capacity.
- R4) Updates:** Gracefully update data structures for dynamic workloads.
- R5) Flexible pipelines:** Easily implement variations of background processes.
- R6) Simplicity:** Efficiently use implementer time through strength of abstraction.

In the context of the lambda architecture, requirement **R4** is the least important requirement, as the views of the Serving Layer are continually recomputed. Nevertheless, we consider this requirement for completeness.

In this subsection we focus on Hadoop and Spark as the main frameworks available for multimedia processing. Much of this discussion draws from [42] and [43]. The first example of implementing multimedia tasks on Hadoop is the work of Zhang et al. [44]. Since then multiple similar systems have been proposed, mostly working with relatively small collections, and some in the medical domain [45–49]. ImageTerrier [50] used the largest collection of these systems, indexing 10.9 million images using bag of words (BoW) features based on about 10 billion SIFT feature vectors. The largest image retrieval experiments on Hadoop indexed and searched around 100M images, or about 30 billion SIFT feature vectors, using a cluster of 100+ machines [43]. The largest image retrieval experiments on Spark indexed and searched the 100M images of the YFCC collection, using nearly 43 billion SIFT feature vectors on a cluster of 51 machines [42].

Hadoop and Spark have also found use in other domains related to multimedia. For example, [51] used Hadoop to implement various computer vision tasks, experimenting with the k -means algorithm clustering about 200GB of data. More recently, [52] proposed a library for Spark to improve the performance of image retrieval. While only k -means is described in detail, the library contains multiple algorithms for descriptor creation, image retrieval, and result processing. Their experiments focus on small collections of less than 500 million descriptors. The KeyStoneML project¹⁵ includes various machine learning algorithms implemented on top of Spark [53]; one is a pipeline for object recognition using Fisher Vectors [54] and SVM. In other recent projects, ADCFs were used for the training phase of deep learning processes, where massive collections feed the network to determine its parameters [55, 56].

8.3.2.1 Map-Reduce and Hadoop

The Map-Reduce framework was first applied to large-scale distributed computing by Google [57]. This framework exploits data independence through the *Map* and *Reduce* user-level functions. Input data is split into blocks stored on participating nodes using a distributed file system such as HDFS [22]. The most popular Map-Reduce framework, Hadoop, favours processing data locally, transparently handles scheduling of tasks, and deals with communications between nodes.

Moise et al. [43] showed that Hadoop fails at adequately solving the issues that multimedia systems raise. Hadoop forces systems to use only a single input source: the HDFS data blocks. Many multimedia services use two sources of data, however, such as retrieval systems which need a codebook at indexing time, in addition to the data that

¹⁵ <http://keystone-ml.org/>.

must be indexed. Hadoop does allow mappers to load distributed variables at launch time, so one such variable can store the codebook. But since mappers on the same physical node cannot share the main memory allocated for that variable, the codebook must be loaded again and again by each mapper, even when running on the same node. Hadoop thus only partially supports requirements **R1** through **R3**.

Hadoop fails with **R4** (updates) because it cannot easily add new feature vectors to the existing data structures. Old and new feature vectors must be merged outside Hadoop, so indexing is done from scratch. Finally, the inflexible two-step Map-Reduce architecture makes it extremely difficult to run iterative or recursive processes. Instead, Hadoop tasks must be embedded inside high-level wrapping code that repeatedly invokes Hadoop and stores the output of Hadoop on HDFS between iterations [58]. Having multiple data sources or complex operation pipelines also leads to complications and similarly poor performance. In short, Hadoop does not satisfy requirements **R5** and **R6**.

8.3.2.2 Spark

The notion of a resilient distributed dataset (RDD) [59, 60], a distributed data structure on disk or in memory, is central to Spark. An RDD could, for example, represent all the semantic features extracted from a media collection. Spark defines operators that transform and manipulate RDDs, and allows chaining operations in arbitrarily deep and complex pipelines. Spark uses a Master-Worker work flow where the main code base is executed on the master and the distributed executions operating on an RDD flow out to workers. Spark uses a lazy execution model where operations on RDDs are chained together until it becomes necessary to instantiate the data, which facilitates various optimizations.

Spark typically uses HDFS as its file system. Data in an RDD is thus typically partitioned and spread out over the computing cluster machines and Spark manipulates the data where it resides. Spark allows the programmer to choose to persist RDDs to various storage-levels; this fine-grained control allows, for example, a distinction between RAM, SSDs, and HDDs. Keeping RDDs in RAM preserves the performance of algorithms with iterative/recursive access patterns.

Guðmundsson et al. [42] described the engineering process for a prototypical (near) web-scale throughput-oriented multimedia service implemented on top of Spark. Their experiments indicate that the requirements of scalability, computational flexibility, and capacity, **R1** through **R3**, are all satisfied quite well by Spark. As an example, Spark was able to use main memory very effectively, meaning that the scalability of the retrieval process was only bound by the amount of RAM per machine and not by the amount of RAM per core, as was the case in Hadoop.

They also described how to implement a form of dynamic index management that went some way towards satisfying the update requirement **R4**, but report that better support is needed. Finally, they showed how Spark's flexibility and deep pipelines provide the tools necessary to implement a full-feature system seamlessly, and how some common post-processing steps, such as re-ranking, could be added with minimal overhead and code changes. None of this was considered remotely feasible with Hadoop [43]. The requirements for flexibility and simplicity of pipelines, **R5** and **R6**, are thus satisfied very well.

8.3.2.3 Comparison

We have discussed six requirements that automatically distributed computing frameworks should satisfy in order to effectively support throughput-oriented multimedia services: scalability, computational flexibility, capacity, updates, flexible pipelines, and simplicity. Results from the literature show that Hadoop partially satisfies the first three, but not the latter three, while Spark satisfies all six requirements, albeit only partially the fourth requirement for efficient updates. Given these results, as well as the rate of improvements between Spark versions, we conclude that Spark has very strong potential for implementing various families of large-scale throughput-oriented multimedia services.

8.3.3 Stream Processing

The Lambda Architecture proposes using stream-processing systems to build the views for the most recent, dynamic data [11]. Stream-processing systems first appeared in the database literature around the turn of the century [61]; early systems included Aurora [62] and STREAM [63]. In a streaming system, operators are applied to the a stream of incoming data, and possibly static relations as well, to process and filter the data stream for presentation or storage. In the case of big multimedia systems, they can be used to maintain the dynamic views.

Twitter was perhaps the first application of web-scale data streaming, originally running on their Storm system [64]. Spark also has a streaming framework, Spark Streaming, where the data stream is discretized and the Spark engine is used to process each fraction of the stream [65]. Their approach has some advantages when failure recovery is needed. More recently, Twitter developed the Heron system to replace Storm and subsequently placed it in the open source domain [66, 67].¹⁶

8.4 Multimedia Delivery

Once the multimedia has been processed and stored in all its forms (versions of raw data, metadata, and derived semantic features), the media must be served to users. A typical multimedia application first queries metadata index structures of the Serving Layers on both sides of the Metadata Storage architecture to determine which media items are of interest before retrieving and presenting the media items themselves. Given the different nature of metadata and media items, it is not surprising that the technologies required for supporting user interactions differ significantly. This chapter discusses these two issues.

As users are generally impatient, low latency of media delivery is of utmost importance. Latency comes primarily from two sources: storage and network access. In the first subsection we first discuss in-memory buffering, which is used in industry to minimize the impact of these two bottlenecks on the delivery of the media files. In the second subsection we discuss briefly the challenges of scaling metadata retrieval and examine the utility of NoSQL systems for that purpose.

¹⁶ <https://flink.apache.org/>. Finally, Apache Flink is a recent framework for automatically distributing streaming computations.

8.4.1 Distributed In-Memory Buffering

A distributed key-value store is a simple storage paradigm commonly used in web-based systems. Key-value stores are designed for caching web contents to avoid access to overloaded data servers; only if the object is not cached is the request sent to the server. The data model is simple: the “value” can essentially be any data object, which will be interpreted by the application itself, while the “key” is simply a unique identifier. In a distributed key-value store, some form of distributed hashing (see section 8.2.2) is then used to choose the server that will store the key-value pair. The idea behind distributed key-value stores is thus to pool the memory available on all the hardware into a common cache. This has two main advantages: first, underutilized memory on one machine can be used as a cache for others; and second, redundant caching of the same data by multiple hosts can be prevented, thus improving overall memory utilization.

Streaming video or audio (and related operations, such as fast-forwarding and pausing) is a special kind of service, especially when the content is generated from a live event. What makes streaming different from delivering smaller media items is that it requires a constant and even flow of data for a relatively long period of time. Streaming is thus vulnerable to disruptions in the connectivity between the client and the stream provider (server). Client-side buffering is commonly used when stream requirements (temporarily) exceed the network capacity, which can be very useful for enhancing the user experience and eliminating the negative effects of unreliable connections. Buffering, however, is not always easy. When streaming a live event, the only way to achieve a buffer is to delay the delivery of the event, but there is a limit to how much delay clients can tolerate. Buffering too far ahead can also be wasteful, especially when a client is “channel surfing”.

8.4.1.1 Memcached and Redis

Memcached¹⁷ and Redis¹⁸ are two high-performance distributed in-memory key-value stores that are both free and open source. To take advantage of either Memcached or Redis, software developers must alter their software such that the key-value store is probed first before the more costly action of contacting the server is taken.

Memcached is purely a main-memory solution, but persistence can be obtained using MemcacheDB¹⁹ or similar solutions. Although originally designed to cache results of SQL queries, Memcached has been applied to large binary files, such as images and videos, by both Facebook and Flickr.²⁰ Redis claims to be a more mature solution, e.g., supporting more structured data and having better security features, and has also been applied on several high-performance social sites.

8.4.1.2 Alluxio

As discussed, the typical key-value stores require software developers to modify code. The alternative is a solution that integrates seamlessly with the existing software.

¹⁷ <https://memcached.org/>.

¹⁸ <https://redis.io/>.

¹⁹ <http://memcachedb.org/>.

²⁰ <https://www.flickr.com/>.

Alluxio²¹ (formerly Tachyon) is a distributed in-memory file system that supports the HDFS API and is designed to fit between your favorite ADCF (Hadoop or Spark) and the underlying distributed file system [68]. The primary advantage of using Alluxio is that it allows in-memory transfer of data between application instances, even if those applications were implemented on different ADCFs. One could, for example, store the output of a Hadoop job to Alluxio's in-memory storage and then load that same data as the input for an application running on Spark. Alluxio will try to keep data cached in-memory, but it can also spill the data down to an underlying storage system if needed. This way Alluxio integrates seamlessly as a middle-layer between existing ADCFs and storage solutions with minimal effort from developers.

8.4.1.3 Content Distribution Networks

The systems described above provide solutions that minimize latency due to access to storage by caching content in (distributed) main memory. When clients are remote, however, potentially located all over the world, network latency becomes more prominent. In this case, many services use content distribution networks (CDNs). These are distributed systems designed specifically to minimize latency and maximize availability in delivering multimedia services, such as audio or video, regardless of the location of the users or the content. The primary value of CDN hosting is that services can focus on what they do best—improving the multimedia system—and let others solve the task of delivering that service to the end user. Otherwise, engineering the delivery of the service can easily become a larger challenge than implementing the service itself. CDN vendors include specialized companies, such as Akamai²² and Cloudflare,²³ and major cloud vendors, such as Amazon²⁴, Google,²⁵ and Microsoft,²⁶ which have also implemented their own CDN services to facilitate distributing data and workloads among their data centers.

8.4.2 Metadata Retrieval and NoSQL Systems

As mentioned in the introduction, multimedia retrieval researchers have slowly been discovering the need for scale in their metadata indexing work. We can broadly divide multimedia metadata into three categories: high-dimensional features, unstructured (textual) metadata and structured (operational) metadata. Due to the variety of metadata, its retrieval must be implemented using a variety of methods.

Keyword indexing is commonly implemented using inverted indices: web-search engines, such as Google²⁷ and Bing,²⁸ for example, employ highly scalable distributed inverted engines. A popular alternative for implementing inverted index retrieval is ElasticSearch,²⁹ a highly scalable open-source full-text search engine.

Work on indexing high-dimensional features is detailed elsewhere in this book, but we observe here that so far there are really no “big data” technologies that focus on

²¹ <https://www.alluxio.com/>.

²² <https://www.akamai.com/>.

²³ <https://www.cloudflare.com/>.

²⁴ <https://aws.amazon.com/>.

²⁵ <https://cloud.google.com/>.

²⁶ <https://azure.microsoft.com/>.

²⁷ <http://www.google.com>.

²⁸ <http://www.bing.com>.

²⁹ <https://www.elastic.co/>.

low-latency implementation of feature-based media services, and likewise we have seen no investigations into the suitability of some of the specialized NoSQL systems for implementing such services. This is clearly a direction for future work in our field, and to facilitate that work we now briefly discuss four different NoSQL system categories that should be considered for this purpose. Note that the systems mentioned in each category are only representative; a much more complete list can be found on Wikipedia.³⁰

8.4.2.1 Key-Value Stores

As described above, a distributed key-value store is a simple storage paradigm commonly used in web-based systems. The data model is simple: the “value” can essentially be any data object, which will be interpreted by the application itself, while the “key” is simply a unique identifier. While the discussion above focused on main-memory key-value stores, several variants focus on large-scale collections and offer a variety of latency/consistency tradeoffs in the spirit of the PACELC formulation of the CAP theorem of section 8.2.2. Some well-known systems include Dynamo [69] and Voldemort.³¹

8.4.2.2 Document Stores

Document stores are similar to key-value stores, except the “value” is a whole document, such as a JSON document containing metadata information. Examples include MongoDB [70] and CouchDB [71]. While the documents can be retrieved using the “key”, such systems typically also offer query capabilities based on the contents of the documents.

8.4.2.3 Wide Column Stores

A wide-column store is also a form of a key-value store, but in this case the “key” can consist of many columns, and the name and format of columns can vary among rows. Examples include Google’s BigTable [72] and Apache Cassandra.³²

8.4.2.4 Graph Stores

Some computations are well modelled using graph abstractions. Examples include social graphs and graphs of reciprocal nearest neighbours. A few systems have been proposed to address these problems, including Neo4j,³³ GraphLab,³⁴ and GraphX [73]. The last is an API on top of Spark for graphs and graph-parallel computations, utilizing the power of cloud computing to allow manipulation of graph data as both graphs and RDD collections.

8.4.3 Discussion

In the first part of this section we illustrated issues related to delivery of media items, while in the latter part we discussed different NoSQL systems that could be considered for implementing metadata retrieval. Studying the applicability of these systems to multimedia applications is an interesting direction for future work in our field.

³⁰ <https://en.wikipedia.org/wiki/NoSQL>.

³¹ <http://www.project-voldemort.com/>.

³² <http://cassandra.apache.org/>.

³³ <https://neo4j.com/>.

³⁴ <https://turi.com/>.

8.5 Case Studies: Facebook

Facebook is the world's largest online social network with nearly two billion active users each month; they also own other well-known online services such as Instagram and WhatsApp. The extent of their service and user base explains why they have by far the largest BLOB (image, video and other “binary large object”) collection in the world. Not only is there much data but it is growing rapidly. In 2013, they reported storing over 240 billion BLOBs [74] and by 2014 that number had grown to over 400 billion [2, 74]. We now discuss two cases of interest that describe some of the multimedia management issues faced by Facebook. The first case study focuses on media storage, while the second case study focuses on media delivery.

8.5.1 Data Popularity: Hot, Warm or Cold

A key attribute of this kind of data is that it is immutable. Users typically upload the data, but they cannot edit the data in place. To deal with the enormous data quantity, Facebook has started classifying their data depending on its popularity, i.e., how frequently it is requested. Initially there were only two categories, “hot” and “cold”, but as of 2014 the classification was extended to include “warm” as well.

Haystack [75] is the storage system for the hot data: the most recent and the most popular. Haystack is designed for high throughput, high availability, and low latency. This is achieved in part by using an in-memory index of the metadata, issuing only a single I/O instruction per BLOB request and by distributing the load by keeping multiple copies of the data. As with many other systems, they default to keeping three copies, and they make sure each copy is hosted in a different data center. Not only does this increase Haystack's fault tolerance but it also increases the likelihood of serving content from a data center close to the user. The storage devices on each Haystack node are arranged in a RAID 6 array, adding another level of performance and fault tolerance, so the final storage requirement is $3.6\times$ the actual data size.

While storage is fairly cheap, maintaining $3.6\times$ storage capacity for even moderately hot data is still costly and was the prime motivator for the development of an alternative. The solution was the definition of “warm” data that was stored in a new system called *f4* [2]. The main idea behind *f4* is that the replication factor can be reduced from $3.6\times$ to as low as $2.1\times$, while still retaining fairly good throughput and fault tolerance. Instead of replicating the data, *f4* uses erasure encoding [76] in combination with classical XOR coding to create a fault-tolerant storage solution that can even handle the loss of an entire data center. Considering that *f4* stores 65 petabytes of logical BLOB data, reducing the physical storage requirements by a factor of 1.5 is a substantial gain [2]. The downside of this trade-off is that there is added overhead in rebuilding and recovering warm data in case of failures and *f4* only handles the loss of a single data center while Haystack can cope with losing two.

The final resting place of the data, so to speak, is “cold storage”, where data is reliably stored but accessing it may take hours or even days. The typical data stored here consists of backups and datasets that are very unlikely to be needed but should still be kept. The primary focus of cold storage is energy consumption and how to detect device failures when the storage devices are powered on. Storage hardware at this level can range from traditional SSDs or HDDs to optical disks or tapes.

8.5.2 Mentions Live

This is a recent service from Facebook that allows public figures to stream live videos from their mobile devices [77]. This created a problem for their in-house CDN that they call the “thundering herd” problem [78]. The way Facebook’s CDN was set up for video streaming was that the public figure streams the video to a live stream processing server, which can be seen as the first layer in a three-level caching hierarchy. The next layer is an origin cache server, which sits between the stream processor and the second cache layer, which is called the edge cache. In this setup there are only a few processing servers and several origin servers, while the vast majority of machines are in the edge cache.

The edge cache machines handle all the user connections and actually serve the content to users. By default, only a single edge cache computer is responsible for each live stream, but a single edge cache computer could be serving multiple streams, possibly from different origins, depending on its load. All incoming requests from users thus first arrive at the edge cache. If the edge cache request is a miss, i.e., the data is not in the edge cache yet, the request is forwarded to the responsible origin server. There the request could also get a cache-miss that would result in the request being forwarded again, this time from the origin to the live stream processing server.

The “thundering herd” problem is that when big celebrities, with millions of followers, start a live stream, the demand for that stream instantly spikes because the millions of followers are notified and the stream is automatically added to their news feed. The result is a huge number of simultaneous requests to the edge server responsible for that stream, which in turn would flood the origin server by forwarding all those requests because they all resulted in cache-misses. The origin server would in turn flood the stream processors for much the same reasons.

The solution that Facebook engineers came up with [78, 79] was to coalesce incoming requests, both on the edge servers and the origin servers, such that as few requests as possible would get forwarded up-stream. After coalescing, the traffic between the edge cache and the origin servers was reduced by 98.2% in one instance (from 200,000 to 3600 forwarded requests) and between the origin and the live stream server to a single request. While this fixed the problem of up-stream overloading, edge cache computers were still overloaded with requests, as each edge server can safely handle 200,000 simultaneous users but exceeding that number can cause issues. Thus, they also introduced an automatic scale-out plan that makes more edge servers available for a single stream if the load per edge server exceeds 200,000 users.

8.6 Conclusions and Future Work

In this chapter we have attempted to answer the question of which of the recently proposed big data storage and management techniques apply to today’s large-scale multimedia collections. We started by presenting an architectural abstraction for such a system, showing that the requirements for such a system could be divided into three categories: media storage, media processing, and media delivery. We then considered each of these requirement categories in turn, and described the techniques and systems most relevant for each. Finally, we presented a couple of case studies from Facebook as a means of demonstrating some of the concepts. A key conclusion is that while media

storage and processing are well served by big data systems, this is not (yet) the case for scalable query processing over all media metadata.

As mentioned in the introduction, the new field of multimedia analytics, which combines visual analytics with multimedia analysis, has been proposed to extract the knowledge and insight that large-scale multimedia collections encode in collaboration between man and machine [5]. In 2016, Jónsson et al. proposed ten research questions for scalable multimedia analytics [80]. We believe that a successful implementation of a large-scale architecture, such as the one proposed in this chapter, would go a long way towards answering some of these questions, in particular:

RQ1: How can data management techniques facilitate multimedia analytics for increasingly large-scale collections of multimedia items and metadata?

RQ5: How can the multimedia analytics system facilitate up-to-date interactive analysis of collections that are dynamically and rapidly evolving?

RQ9: How can data management techniques be leveraged to improve the user's interactive experience?

Further work is, of course, needed to fully understand the weaknesses and strengths of the architecture and its components. We believe, however, that working at scale is imperative, as the time of analysing small media sets is simply behind us.

But what does it take to work at scale? In section 8.3 we reviewed the work of [42], who indexed 43 billion high-dimensional features with Spark. In addition to reporting on their implementation, they also reported on the engineering process of the project. A key observation was that working with collections at their scale (6 TB of storage) is a very time-consuming process. As one example, converting the feature collection from text format to binary format took weeks. Furthermore, during their work they hit a number of bugs in Spark, which slowed progress as detecting, isolating, and uncovering each bug typically took weeks. We conclude that researchers in the multimedia field must anticipate significant effort to manage large-scale collections and the corresponding research environment in order to solve grand issues in multimedia retrieval and analytics. In order to make progress towards solving web-scale problems, however, this effort is simply necessary!

8.6.1 Acknowledgments

We wish to thank Edward Y. Chang, Jens Dittrich, Rasmus Pagh, Irina Shklovski, and Ýmir Vigfússon, as well as the anonymous reviewers, for their constructive and detailed feedback, which greatly helped improve the presentation of the chapter.

References

- 1 Thomee, B., Shamma, D.A., Friedland, G., Elizalde, B., Ni, K., Poland, D., Borth, D., and Li, L.J. (2016) YFCC100M: The new data in multimedia research. *Communications of the ACM*, 59 (2), 64–73.
- 2 Muralidhar, S., Lloyd, W., Roy, S., Hill, C., Lin, E., Liu, W., Pan, S., Shankar, S., Sivakumar, V., Tang, L. et al. (2014) f4: Facebook's warm blob storage system, in *Proceedings of the Symposium on Operating Systems Design and Implementation (OSDI)*, Broomfield, CO, USA, pp. 383–398.

- 3 Smith, C. (2017), 160 amazing YouTube statistics (July 2017), Online report, <http://expandedramblings.com/index.php/youtube-statistics/>. (accessed 19 August 2017).
- 4 Moosavi-Dezfooli, S., Fawzi, A., and Frossard, P. (2016) DeepFool: A simple and accurate method to fool deep neural networks, in *Proceedings of the IEEE International Conference on Computer Vision & Pattern Recognition*, Las Vegas, NV, USA, pp. 2574–2582.
- 5 Zahálka, J. and Worring, M. (2014) Towards interactive, intelligent, and integrated multimedia analytics, in *Proceedings of the IEEE Conference on Visual Analytics Science and Technology (VAST)*, Paris, France, pp. 3–12.
- 6 Zahálka, J., Rudinac, S., Jónsson, B.P., Koelma, D.C., and Worring, M. (2016) Interactive multimodal learning on 100 million images, in *Proceedings of the ACM International Conference on Multimedia Retrieval (ICMR)*, New York, NY, USA, pp. 333–337.
- 7 Ghemawat, S., Gobioff, H., and Leung, S.T. (2003) The Google file system, in *Proceedings of the ACM Symposium on Operating Systems Principles (SOSP)*, pp. 29–43.
- 8 Shvachko, K., Kuang, H., Radia, S., and Chansler, R. (2010) The Hadoop distributed file system, in *Proceedings of the 2010 IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST '10)*, IEEE Computer Society, Washington, DC, pp. 1–10, doi: 10.1109/MSST.2010.5496972. <http://dx.doi.org/10.1109/MSST.2010.5496972>.
- 9 Mohan, C. (2013) History repeats itself: Sensible and nonsensql aspects of the nosql hoopla, in *Proceedings of the 16th International Conference on Extending Database Technology (EDBT '13)*, ACM, New York, pp. 11–16, doi: 10.1145/2452376.2452378. <http://doi.acm.org/10.1145/2452376.2452378>.
- 10 Lowe, D.G. (2004) Distinctive image features from scale-invariant keypoints. *International Journal on Computer Vision*, 60 (2), doi: 10.1023/B:VISI.0000029664.99615.94.
- 11 Marz, N. and Warren, J. (2015) *Big Data: Principles and best practices of scalable real-time data systems*, Manning Publication Co.
- 12 Jónsson, B.P., Amsaleg, L., and Lejsek, H. (2016) SSD technology enables dynamic maintenance of persistent high-dimensional indexes, in *Proceedings of the ACM International Conference on Multimedia Retrieval (ICMR)*, New York, NY, USA, pp. 347–350.
- 13 Floyer, D. (2015), Enterprise flash vs. HDD projections 2012-2026, Wikibon, August 11, 2015. <https://wikibon.com/enterprise-flash-vs-hdd-forecasts-2012-2026/>.
- 14 Gray, J. and Putzolu, G.R. (1987) The 5 minute rule for trading memory for disk accesses and the 10 byte rule for trading memory for CPU time, in *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD)*, San Francisco, CA, USA, pp. 395–398.
- 15 Gray, J. and Graefe, G. (1997) The five-minute rule ten years later, and other computer storage rules of thumb. *SIGMOD Record*, 26 (4), 63–68.
- 16 Graefe, G. (2009) The five-minute rule 20 years later (and how flash memory changes the rules). *Communications of the ACM*, 52 (7), 48–59.
- 17 O'Neil, P., Cheng, E., Gawlick, D., and O'Neil, E. (1996) The log-structured merge-tree (LSM-tree). *Acta Informatica*, 33 (4), 351–385.
- 18 Roberts, D., Chang, J., Ranganathan, P., and Mudge, T.N. (2010) Is storage hierarchy dead? Co-located compute-storage NVRAM-based architectures for data-centric workloads. *Technical Report HPL-2010-119*.

- 19 Björling, M., González, J., and Bonnet, P. (2017) Lightnvm: The linux open-channel ssd subsystem, in *Proceedings of the 15th Usenix Conference on File and Storage Technologies (FAST'17)*, pp. 359–374.
- 20 Gray, J. (1998), Put everything in the disk controller, http://jimgray.azurewebsites.net/talks/Gray_NASD_Talk.ppt.
- 21 Dean, J. (2009) Designs, lessons and advice from building large distributed systems, Keynote at LADIS 2009. <https://www.cs.cornell.edu/projects/ladis2009/talks/dean-keynote-ladis2009.pdf>.
- 22 Shvachko, K., Kuang, H., Radia, S., and Chansler, R. (2010) The Hadoop distributed file system, in *Proceedings of the IEEE Symposium on Mass Storage Systems and Technologies (MSST)*, Lake Tahoe, NV, USA, pp. 1–10.
- 23 Brewer, E. (2012) CAP twelve years later: How the “rules” have changed. *IEEE Computer*, 45 (2), 23–29.
- 24 Abadi, D.J. (2012) Consistency tradeoffs in modern distributed database system design. *IEEE Computer*, 45 (2), 37–42.
- 25 Weil, S.A., Leung, A.W., Brandt, S.A., and Maltzahn, C. (2007) RADOS: a scalable, reliable storage service for petabyte-scale storage clusters, in *Proceedings of the International Petascale Data Storage Workshop (PDSW)*, Reno, NV, USA, pp. 35–44.
- 26 Hinton, G., Deng, L., Yu, D., Dahl, G.E., Mohamed, A., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T.N., and Kingsbury, B. (2012) Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29 (6), 82–97.
- 27 Kavukcuoglu, K., Sermanet, P., Boureau, Y., Gregor, K., Mathieu, M., and LeCun, Y. (2010) Learning convolutional feature hierarchies for visual recognition, in *Proceedings of the Neural Information Processing Systems Conference (NIPS)*, Vancouver, BC, Canada, pp. 1090–1098.
- 28 Lee, H., Grosse, R.B., Ranganath, R., and Ng, A.Y. (2009) Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations, in *Proceedings of the International Conference on Machine Learning (ICML)*, Montreal, Quebec, Canada, pp. 609–616.
- 29 Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., and Li, F. (2014) Large-scale video classification with convolutional neural networks, in *Proceedings of the IEEE International Conference on Computer Vision & Pattern Recognition*, Columbus, OH, USA, pp. 1725–1732.
- 30 Lee, C., Xie, S., Gallagher, P.W., Zhang, Z., and Tu, Z. (2015) Deeply-supervised nets, in *Proceedings of the International Conference on Artificial Intelligence and Statistics*, San Diego, CA, USA, pp. 562–570.
- 31 Krizhevsky, A., Sutskever, I., and Hinton, G.E. (2017) Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60 (6), 84–90.
- 32 Krizhevsky, A., Sutskever, I., and Hinton, G.E. (2012) Imagenet classification with deep convolutional neural networks, in *Proceedings of the Neural Information Processing Systems Conference (NIPS)*, Lake Tahoe, NV, USA, pp. 1106–1114.
- 33 Song, H.O., Lee, Y.J., Jegelka, S., and Darrell, T. (2014) Weakly-supervised discovery of visual pattern configurations, in *Proceedings of the Neural Information Processing Systems Conference (NIPS)*, Montreal, Quebec, Canada, pp. 1637–1645.
- 34 Xu, Y., Mo, T., Feng, Q., Zhong, P., Lai, M., and Chang, E.I. (2014) Deep learning of feature representation with multiple instance learning for medical image analysis,

- in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Florence, Italy, pp. 1626–1630.
- 35 Babenko, A. and Lempitsky, V.S. (2016) Efficient indexing of billion-scale datasets of deep descriptors, in *Proceedings of the IEEE International Conference on Computer Vision & Pattern Recognition*, Las Vegas, NV, USA, pp. 2055–2063.
 - 36 Awad, G., Kraaij, W., Over, P., and Satoh, S. (2017) Instance search retrospective with focus on TRECVID. *International Journal of Multimedia Information Retrieval*, 6 (1), 1–29.
 - 37 Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., and Darrell, T. (2014) Caffe: Convolutional architecture for fast feature embedding, in *Proceedings of the ACM International Conference on Multimedia (MM)*, Orlando, FL, USA, pp. 675–678.
 - 38 Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., Kudlur, M., Levenberg, J., Monga, R., Moore, S., Murray, D.G., Steiner, B., Tucker, P.A., Vasudevan, V., Warden, P., Wicke, M., Yu, Y., and Zheng, X. (2016) TensorFlow: A system for large-scale machine learning, in *Proceedings of the Symposium on Operating Systems Design and Implementation (OSDI)*, Savannah, GA, USA, pp. 265–283.
 - 39 Chen, T., Li, M., Li, Y., Lin, M., Wang, N., Wang, M., Xiao, T., Xu, B., Zhang, C., and Zhang, Z. (2015) MXNet: A flexible and efficient machine learning library for heterogeneous distributed systems. *Computing Research Repository*, abs/1512.01274.
 - 40 Hunter, T. (2016), Deep learning with Apache Spark and TensorFlow, Databricks Engineering Blog, January 25, 2016. <https://databricks.com/blog/2016/01/25/deep-learning-with-apache-spark-and-tensorflow.html>.
 - 41 Hong, S.A., Hunter, T., and Xin, R. (2017), A vision for making deep learning simple, Databricks Engineering Blog, June 6, 2017. <https://databricks.com/blog/2017/06/06/databricks-vision-simplify-large-scale-deep-learning.html>.
 - 42 Guðmundsson, G.P., Amsaleg, L., Jónsson, B.P., and Franklin, M.J. (2017) Towards engineering a web-scale multimedia service: A case study using spark, in *Proceedings of the ACM Multimedia Systems Conference (MMSys)*, ACM, Taipei, Taiwan.
 - 43 Moise, D., Shestakov, D., Guðmundsson, G.P., and Amsaleg, L. (2013) Indexing and searching 100M images with Map-Reduce, in *Proceedings of the ACM International Conference on Multimedia Retrieval (ICMR)*, Dallas, TX, USA, pp. 17–24.
 - 44 Zhang, J., Liu, X., Luo, J., and Lang, B. (2010) DIRS: Distributed image retrieval system based on MapReduce, in *Proceedings of the International Conference on Pervasive Computing and Applications (PCA)*, Maribor, Slovenia, pp. 93–98.
 - 45 Gu, C. and Gao, Y. (2012) A content-based image retrieval system based on Hadoop and Lucene, in *Proceedings of the International Conference on Cloud and Green Computing (CGC)*, Xiangtan, China, pp. 684–687.
 - 46 Premchaiswadi, W., Tungkatsathan, A., Intarasema, S., and Premchaiswadi, N. (2013) Improving performance of content-based image retrieval schemes using Hadoop MapReduce, in *Proceedings of the International Conference on High Performance Computing and Simulation (HPCS)*, Helsinki, Finland, pp. 615–620.
 - 47 Grace, R.K., Manimegalai, R., and Kumar, S.S. (2014) Medical image retrieval system in grid using Hadoop framework, in *Proceedings of the International Conference on Computer Science and Computational Intelligence (ICCSCI)*, Las Vegas, NV, USA, pp. 144–148.

- 48 Yao, Q.A., Zheng, H., Xu, Z.Y., Wu, Q., Li, Z.W., and Yun, L. (2014) Massive medical images retrieval system based on Hadoop. *Journal of Multimedia*, 9 (2).
- 49 Jai-Andalousi, S., Elabdouli, A., Chaffai, A., Madrane, N., and Sekkaki, A. (2013) Medical content based image retrieval by using the Hadoop framework, in *International Conference on Telecommunications (ICT)*, Casablanca, Morocco, pp. 1–5.
- 50 Hare, J.S., Samangoeei, S., Dupplaw, D.P., and Lewis, P.H. (2012) ImageTerrier: An extensible platform for scalable high-performance image retrieval, in *Proceedings of the ACM International Conference on Multimedia Retrieval (ICMR)*, Hong Kong, China, p. 40.
- 51 White, B., Yeh, T., Lin, J., and Davis, L.S. (2010) Web-scale computer vision using MapReduce for multimedia data mining, in *Proceedings of the International Workshop on Multimedia Data Mining (MDMKDD)*, Washington, DC, USA, pp. 9:1–9:10.
- 52 Wang, H., Xiao, B., Wang, L., and Wu, J. (2015) Accelerating large-scale image retrieval on heterogeneous architectures with Spark, in *Proceedings of the ACM International Conference on Multimedia (MM)*, Brisbane, Australia, pp. 1023–1026.
- 53 Sparks, E.R., Venkataraman, S., Kaftan, T., Franklin, M.J., and Recht, B. (2017) KeystoneML: Optimizing pipelines for large-scale advanced analytics, in *Proceedings of the IEEE International Conference on Data Engineering (ICDE)*, San Diego, CA, USA, pp. 535–546.
- 54 Perronnin, F., Sánchez, J., and Mensink, T. (2010) Improving the Fisher kernel for large-scale image classification, in *Proceedings of the European Conference on Computer Vision (ECCV)*, *Lecture Notes in Computer Science*, vol. 6314, pp. 143–156.
- 55 Ooi, B.C., Tan, K.L., Wang, S., Wang, W., Cai, Q., Chen, G., Gao, J., Luo, Z., Tung, A.K., Wang, Y., Xie, Z., Zhang, M., and Zheng, K. (2015) SINGA: A distributed deep learning platform, in *Proceedings of the ACM International Conference on Multimedia (MM)*, Brisbane, Australia, pp. 685–688.
- 56 Moritz, P., Nishihara, R., Stoica, I., and Jordan, M.I. (2015) SparkNet: Training deep networks in Spark, *arXiv:1511.06051*.
- 57 Dean, J. and Ghemawat, S. (2008) Mapreduce: simplified data processing on large clusters. *Communications of the ACM*, 51 (1), 72–77.
- 58 Owen, S., Anil, R., Dunning, T., and Friedman, E. (2011) *Mahout in Action*, Manning Publications Co.
- 59 Zaharia, M., Chowdhury, M., Franklin, M.J., Shenker, S., and Stoica, I. (2010) Spark: Cluster computing with working sets, in *Proceedings of the USENIX Workshop on Hot Topics in Cloud Computing (HotCloud)*, Boston, MA, USA.
- 60 Zaharia, M., Chowdhury, M., Das, T., Dave, A., Ma, J., McCauley, M., Franklin, M.J., Shenker, S., and Stoica, I. (2012) Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing, in *Proceedings of the Symposium on Networked Systems Design and Implementation (NSDI)*, San Jose, CA, USA, pp. 15–28.
- 61 Carney, D., Çetintemel, U., Cherniack, M., Convey, C., Lee, S., Seidman, G., Stonebraker, M., Tatbul, N., and Zdonik, S.B. (2002) Monitoring streams – A new class of data management applications, in *Proceedings of the International Conference on Very Large Data Bases (VLDB)*, Hong Kong, China, pp. 215–226.
- 62 Abadi, D.J., Carney, D., Çetintemel, U., Cherniack, M., Convey, C., Lee, S., Stonebraker, M., Tatbul, N., and Zdonik, S.B. (2003) Aurora: a new model and architecture for data stream management. *VLDB Journal*, 12 (2), 120–139.

- 63 Arasu, A., Babcock, B., Babu, S., Datar, M., Ito, K., Motwani, R., Nishizawa, I., Srivastava, U., Thomas, D., Varma, R., and Widom, J. (2003) STREAM: the Stanford Stream Data Manager. *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, 26 (1), 19–26.
- 64 Toshniwal, A., Taneja, S., Shukla, A., Ramasamy, K., Patel, J.M., Kulkarni, S., Jackson, J., Gade, K., Fu, M., Donham, J., Bhagat, N., Mittal, S., and Ryaboy, D.V. (2014) Storm@twitter, in *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD)*, Snowbird, UT, USA, pp. 147–156.
- 65 Zaharia, M., Das, T., Li, H., Hunter, T., Shenker, S., and Stoica, I. (2013) Discretized streams: fault-tolerant streaming computation at scale, in *Proceedings of the ACM Symposium on Operating Systems Principles (SOSP)*, Farmington, PA, USA, pp. 423–438.
- 66 Fu, M., Agrawal, A., Floratou, A., Graham, B., Jorgensen, A., Li, M., Lu, N., Ramasamy, K., Rao, S., and Wang, C. (2017) Twitter heron: Towards extensible streaming engines, in *Proceedings of the IEEE International Conference on Data Engineering (ICDE)*, San Diego, CA, USA, pp. 1165–1172.
- 67 Kulkarni, S., Bhagat, N., Fu, M., Kedigehalli, V., Kellogg, C., Mittal, S., Patel, J.M., Ramasamy, K., and Taneja, S. (2015) Twitter heron: Stream processing at scale, in *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD)*, Melbourne, Australia, pp. 239–250.
- 68 Li, H., Ghodsi, A., Zaharia, M., Shenker, S., and Stoica, I. (2014) Tachyon: Reliable, memory speed storage for cluster computing frameworks, in *Proceedings of the ACM Symposium on Cloud Computing (SoCC)*, Santa Clara, CA, USA, pp. 1–15.
- 69 DeCandia, G., Hastorun, D., Jampani, M., Kakulapati, G., Lakshman, A., Pilchin, A., Sivasubramanian, S., Voshall, P., and Vogels, W. (2007) Dynamo: Amazon's highly available key-value store, in *Proceedings of the ACM Symposium on Operating Systems Principles (SOSP)*, Stevenson, WA, USA, pp. 205–220.
- 70 Chodorow, K. and Dirolf, M. (2010) *MongoDB – The Definitive Guide: Powerful and Scalable Data Storage*, O'Reilly.
- 71 Brown, M.C. (2012) *Getting Started with CouchDB – Extreme Scalability at Your Fingertips*, O'Reilly.
- 72 Chang, F., Dean, J., Ghemawat, S., Hsieh, W.C., Wallach, D.A., Burrows, M., Chandra, T., Fikes, A., and Gruber, R. (2006) Bigtable: A distributed storage system for structured data, in *Proceedings of the Symposium on Operating Systems Design and Implementation (OSDI)*, Seattle, WA, USA, pp. 205–218.
- 73 Gonzalez, J.E., Xin, R.S., Dave, A., Crankshaw, D., Franklin, M.J., and Stoica, I. (2014) Graphx: Graph processing in a distributed dataflow framework, in *USENIX Symposium on Operating Systems Design and Implementation*, Broomfield, CO, USA, pp. 599–613.
- 74 Gasperetti, J. and Pan, S. (2014), F4 – Photo Storage at Facebook, Presentation at Data @ Scale. https://www.youtube.com/watch?v=34e_g-Ji_30.
- 75 Beaver, D., Kumar, S., Li, H.C., Sobel, J., and Vajgel, P. (2010) Finding a needle in Haystack: Facebook's photo storage, in *Proceedings of the Symposium on Operating Systems Design and Implementation (OSDI)*, Vancouver, BC, Canada, pp. 47–60.
- 76 Reed, I.S. and Solomon, G. (1960) Polynomial codes over certain finite fields. *Journal of the Society for Industrial and Applied Mathematics*, 8 (2), 300–304.

- 77 Lavrusik, V. (2015), Introducing Live for Facebook Mentions, *Facebook Blog*, August 5, 2015. <https://media.fb.com/2015/08/05/introducing-live-for-facebook-mentions/>.
- 78 Larumbe, F. and Mathur, A. (2015), Under the hood: Broadcasting live video to millions, Facebook blog. <https://code.facebook.com/posts/1653074404941839/under-the-hood-broadcasting-live-video-to-millions/>.
- 79 Larumbe, F. (2016), Scaling Facebook Live, Presentation at Networking @ Scale. <https://atscaleconference.com/videos/scaling-facebook-live/>.
- 80 Jónsson, B.P., Worring, M., Zahálka, J., Rudinac, S., and Amsaleg, L. (2016) Ten research questions for scalable multimedia analytics, in *Proceedings of the International Conference on MultiMedia Modeling (MMM)*, Miami, FL, USA, pp. 290–302.

9

Perceptual Hashing for Large-Scale Multimedia Search

Li Weng, I-Hong Jhuo and Wen-Huang Cheng

Among various challenges to multimedia search, accuracy and scalability are of critical importance. Conventional techniques typically focus on the former. Recently, more and more attention has been paid to the latter because the amount of multimedia data is rapidly increasing like never before. Due to the curse of dimensionality, conventional feature representations are not in favor of fast search. The big data era needs new solutions for multimedia indexing and retrieval. In this chapter, a technique called perceptual hashing is presented together with a particular category of algorithms called perceptual hash algorithms. These algorithms are used for generating hash values from multimedia objects, such as images, audio, and video. A hash value is a compact string of, e.g., hundreds of bits. The basic property of perceptual hashing is robustness, i.e., similar content should result in similar hash values. Another important property is discrimination, i.e., different content should result in different hash values. Because of these properties, hash values can be considered as fingerprints of the corresponding content. Thus one possibility to achieve content-based multimedia search is to compare hash values. There are several advantages of using hash values instead of other content representations. Typically, since hash values are compact, they facilitate fast in-memory indexing and search. This is particularly interesting for large-scale media repositories. In this chapter, the basics of perceptual hashing are introduced, with a focus on learning-based (data-oriented) algorithms. The text is organized into five parts. The first part introduces definitions and properties of perceptual hashing, as well as applications, performance evaluation, etc. The second part is about unsupervised perceptual hash algorithms. The third part is about supervised perceptual hash algorithms. A few representative algorithms are described in detail for both categories. Most of them have approximately linear structures. They help to understand the mechanism of perceptual hashing. In addition, some kernel and neural network based algorithms are also presented. They are more flexible because nonlinear transformations are used. However, the implementation and the training costs are higher. In the fourth part, two versatile frameworks for constructing hash algorithms are described. The first approach generates optimal hash values from training data, then trains general classifiers to realize the hash algorithm. The second approach is about hash bit selection and combination. Since there are many existing

hash algorithms, it can be more efficient and effective to combine existing algorithms instead of designing new ones from scratch. A method for this purpose is presented. The last part concludes the chapter with a discussion on the pros and cons of perceptual hashing and future research topics.

9.1 Introduction

Efficient multimedia search requires compact and effective representation of data. A promising way to achieve this goal is perceptual hashing. The terminology “hashing” generally means to compress data with a hash algorithm. A hash algorithm $H(\cdot)$ converts an input string \mathbf{x} into an output string \mathbf{y} , also known as a *hash value*:

$$\mathbf{y} = H(\mathbf{x}), \quad (9.1)$$

where $\mathbf{x} = [x_1, \dots, x_m]^T \in \mathbb{X}^m$, $\mathbf{y} = [y_1, \dots, y_n]^T \in \mathbb{Y}^n$. Mathematically, hashing is a mapping from one space \mathbb{X}^m to another \mathbb{Y}^n , where \mathbb{X} is the input alphabet and \mathbb{Y} is the output alphabet. In the multimedia domain, typically \mathbf{x} is a real (feature) vector and \mathbf{y} is a binary vector, i.e., $\mathbb{X} = \mathbb{R}$, $\mathbb{Y} = \{0, 1\}$. Given two hash values \mathbf{h}_1 and \mathbf{h}_2 , their difference is typically calculated using the Hamming distance

$$d_H(\mathbf{h}_1, \mathbf{h}_2) = \sum_{i=1}^n |h_{1,i} - h_{2,i}|, \quad (9.2)$$

which can be efficiently computed by the exclusive or operation. In literature, it is more convenient to represent a “0” bit as “−1” for algorithm presentation. In the rest of the chapter, we use $\{-1, 1\}$ representation by default.

A hash value can be thought of as a label of a multimedia object, such as an image, video, or audio. Perceptual hashing is essentially about assigning hash values to certain multimedia objects. Ideally, n -bit hash values can represent 2^n objects. Thus a small n is sufficient for many applications. Compared with conventional approaches, perceptual hashing has two major advantages:

- comparison of hash values can be extremely fast
- hash values can be massively stored in memory.

They enable large-scale applications of multimedia search, including images [1–3], audio [4, 5], and video [6, 7].

9.1.1 Related work

Perceptual hashing is one of the approaches that seek compact representations of multimedia data. It is typically applied to extracted features to achieve further compression. Alternatively, similar effects can be obtained by using compact feature descriptors. In the image domain, there are already many efforts along this line to derive efficient descriptors. They typically produce global descriptors by summarizing local features such as scale-invariant feature transform (SIFT) (see chapter 2, ref. [4]). A classic representation is bag-of-features [8], which may incur vocabularies of hundreds to millions of dimensions. A more compact representation is the Fisher vector [9],

which models visual words as Gaussian mixtures. Another popular global descriptor is GIST [10], which is typically 960-dimensional. A recent approach is vector of locally aggregated descriptors (VLAD) (see chapter 2, ref. [8]), which is a simplified version of the Fisher vector, and can be as compact as 128 bits. It is extended to NetVLAD [11] in a deep learning context. Besides feature representations, dimension reduction and quantization methods, such as principal component analysis (PCA) [12] and k -means clustering [13], can also help with compression. A recent popular technique is product quantization [14], which is particularly useful for compressing high-dimensional feature descriptors.

The aforementioned techniques can either be used alone or as building blocks of more sophisticated schemes, including perceptual hash algorithms.

9.1.2 Definitions and Properties of Perceptual Hashing

Perceptual hashing mainly consists of two parts: (i) hash generation and (ii) hash verification. These are illustrated in Figure 9.1. Hash generation is the focus of hash algorithm design. There are a few essential components: feature extraction, feature transformation, dimension reduction, quantization, and randomization. While all components are useful, learning-based hash algorithms (including those presented in this chapter) typically focus on improving feature transformation and dimension reduction. They assume general features of any vector form, ranging from image pixels to descriptors such as SIFT (see chapter 2, ref. [4]) and GIST [10].

Hash verification is typically made simple in order to be fast. Most algorithms use the Hamming distance, which can be as fast as one billion comparisons per second [15]. The distance is compared with a threshold. If the distance is not larger than the threshold,

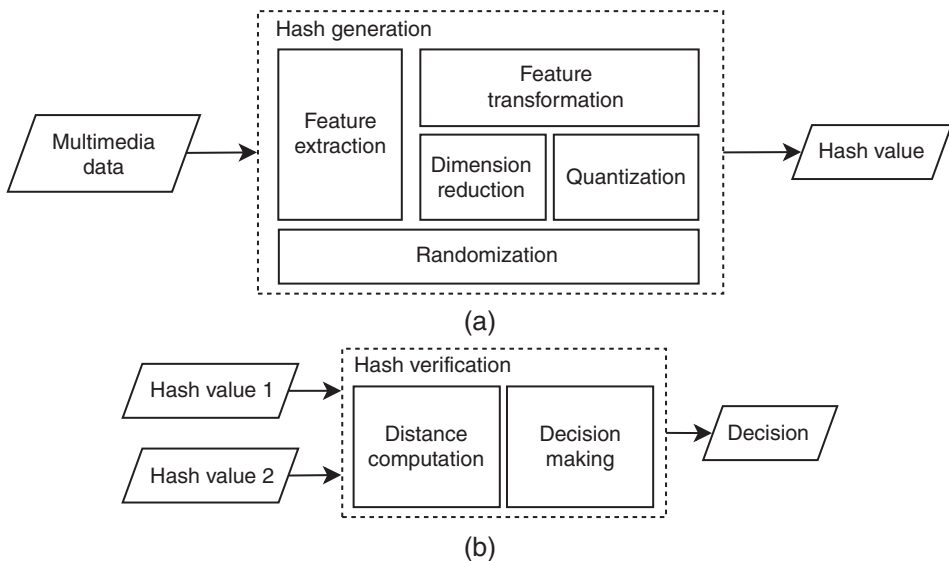


Figure 9.1 Schematic diagrams of perceptual hashing: (a) hash generation and (b) hash verification

the corresponding multimedia objects are judged as similar (or relevant); otherwise they are judged as dissimilar (or irrelevant).

Perceptual hashing is different from conventional hashing techniques, mainly because the hash value depends on multimedia content rather than file representation. The basic properties are robustness and discrimination:

Robustness – relevant multimedia objects have similar hash values

Discrimination – irrelevant multimedia objects have dissimilar hash values.

In other words, the goal of perceptual hashing is to approximate the perceptual distance between two multimedia objects with the distance between their hash values. The perceptual distance, or relevance, can be defined in many ways. Typically, it can be the Euclidean distance or the cosine similarity between feature vectors, or it can be a semantic distance defined by label consistency. In order to proactively achieve the targeted approximation, many learning-based approaches have emerged. In general, they require a training dataset $\{\mathbf{D}_i\}_{i=1}^M$ with M samples. According to the properties of training data, they can be divided into two categories. In unsupervised learning, each sample is just a data point (feature vector), i.e., $\mathbf{D}_i = \mathbf{x}_i$. In supervised learning, each sample additionally contains some information associated with the corresponding feature vector, i.e., $\mathbf{D}_i = \mathbf{x}_i | \mathbf{t}_i$. Typically, \mathbf{t}_i represents a class label (vector). A simple and commonly used representation of “knowledge” in training data is a matrix form

$$\mathbf{D}^p = [D_{ij}^p] = [d_p(\mathbf{x}_i, \mathbf{x}_j)], \quad (9.3)$$

which is called the affinity or similarity matrix, where D_{ij}^p is the *perceptual* distance between \mathbf{x}_i and \mathbf{x}_j . Correspondingly, a hash affinity matrix can be built in a similar way

$$\mathbf{D}^h = [D_{ij}^h] = [d_h(\mathbf{x}_i, \mathbf{x}_j)], \quad (9.4)$$

where D_{ij}^h is an affinity measure between $H(\mathbf{x}_i)$ and $H(\mathbf{x}_j)$. Note that the hash affinity measure can be different from the Hamming distance. Both \mathbf{D}^p and \mathbf{D}^h represent pairwise relationships, in feature space and hash space respectively. Typically they are symmetric. In general, perceptual hash algorithms enforce \mathbf{D}^h to approximate \mathbf{D}^p in one way or another.

Given a list of hash values $\{\mathbf{y}_i\}_{i=1}^M$ generated from a training dataset, there are some basic expectations or constraints:

- **Constraint 0:** Binary hash values

$$\mathbf{y}_i \in \{-1, 1\}^n. \quad (9.5)$$

- **Constraint 1:** Equal probable hash bits

$$\lim_{M \rightarrow \infty} \sum_{i=1}^M \mathbf{y}_i = \mathbf{0}, \quad (9.6)$$

where $\mathbf{0}$ is an $n \times 1$ zero vector.

- **Constraint 2:** Uncorrelated hash bits

$$\lim_{M \rightarrow \infty} \frac{1}{M} \sum_{i=1}^M \mathbf{y}_i \cdot \mathbf{y}_i^T = \mathbf{I}, \quad (9.7)$$

where \mathbf{I} is an $n \times n$ identity matrix.

In practice, the constraints are used with $M \ll \infty$. Very often they need to be relaxed in order to efficiently find a solution.

Typically, an n -bit hash algorithm consists of n hash functions $h_i(\cdot)$, $i = 1, \dots, n$, each producing one bit. For an input \mathbf{x} , the output hash value $\mathbf{y} = [y_1, \dots, y_n]^T$ is computed with all hash functions:

$$y_i = h_i(\mathbf{x}) = \text{sgn}(f_i(\mathbf{x})) = \begin{cases} 1, & \text{if } f_i(\mathbf{x}) \geq 0 \\ -1, & \text{otherwise} \end{cases}, \quad (9.8)$$

where the sign function $\text{sgn}(\cdot)$ is commonly used for binarization. When hash bits are represented by $\{-1, 1\}$, the Hamming distance can also be computed in terms of the inner product [16]

$$d_H(\mathbf{y}_i, \mathbf{y}_j) = (n - \mathbf{y}_i^T \cdot \mathbf{y}_j)/2. \quad (9.9)$$

Designing hash algorithms is about finding suitable forms of $h_i(\cdot)$ or $f_i(\cdot)$.

9.1.3 Multimedia Search using Perceptual Hashing

Using hash values, there are two basic ways to perform multimedia search:

- linear scan
- hash table look-up.

The former achieves the best accuracy, while the latter improves efficiency. They both start with computing a hash value from a query input.

Linear scan means to compare the query hash value with all hash values in the database. All database items are ranked according to the hash distance. Afterwards, the top k items are retrieved. It is simple and effective when the size of the database N is small. The search complexity is $O(N \log N)$.

Hash table look-up means to create an inverted index structure for all database items. Basically, all items that result in the same hash value are put into the same “hash bucket”. Given a query hash value, relevant items can be found by directly accessing the corresponding bucket in $O(1)$ complexity. Typically, each query probes only one hash bucket. However, it is possible to check multiple buckets with a multi-probing strategy if more items are needed [17]. A common multi-probing strategy is to consider all buckets within a Hamming ball of radius r , i.e., $d_H(\mathbf{h}_q, \mathbf{h}_x) \leq r$.

With n -bit hash values, a hash table would have 2^n buckets. The indexing structure might not fit into memory when n is large. In that case, a “product hashing” approach can be used [17]. Each hash value is split into k parts with lengths n_1, \dots, n_k , which are called sub-hash values. A hash table can be built for each sub-hash value. Typically, $n_1 = \dots = n_k = n/k$. During retrieval, each table is probed to get a list of candidates. These lists are merged and re-ranked using the full hash values.

Sometimes, ranking by Hamming distance does not have sufficient granularity due to a small hash length. In that case, the original feature vectors can be used to further refine the ranking.

9.1.4 Applications of Perceptual Hashing

Roughly speaking, perceptual hashing is used for efficient content search. More specifically, most applications can be divided into three categories: (i) content identification,

(ii) content classification, and (iii) content authentication. The essential difference among the three is the level of robustness and discrimination. For content identification (a.k.a. near-duplicate detection), the goal is to find similar content in the presence of incidental distortion, such as noise addition and geometric distortion. In this case, similarity between multimedia objects is typically defined on a global level. For content classification, the goal is similar, but the similarity criterion is more broadly defined, sometimes based on semantics. In fact, content identification can be considered as a special case of classification with infinite classes.

Content authentication is relatively less studied. It is interesting in security-related applications, where the goal is to detect malicious modification in the presence of incidental or legitimate distortion. This typically requires similarity to be defined on a local level, resulting in strong discrimination.

In addition to conventional scenarios, perceptual hashing is also used in novel applications. Since hash computation is “one-way”, hash-based approaches are suitable for privacy protection in multimedia search [17, 18]. The compactness of hash values also facilitates signal quality estimation with reduced reference [19, 20]. Although sending hash values takes little effort, it is possible to further reduce data transmission by combining perceptual hashing with digital watermarking [21].

9.1.5 Evaluating Perceptual Hash Algorithms

Perceptual hash algorithms are typically evaluated by two metrics: (i) precision-recall (PR) and (ii) receiver operating characteristic (ROC) [22]. The former calculates the precision and the recall; the latter calculates the true positive rate (which is the same as recall) and the false positive rate. While both metrics are useful for algorithm comparison, their usage depends on the application. In retrieval scenarios where positive cases are the main concern, PR is typically used. In decision-making scenarios where negative cases also matter, ROC is more appropriate. In general, ROC might be used for performance evaluation because it is unbiased irrespective of applications.

In addition to performance, the selection of perceptual hash algorithms also depends on several other factors: training complexity, running complexity, generality, etc. From learning’s point of view, performance might be improved by increasing model complexity. For example, nonlinear representations might replace linear ones, multiple optimization goals might replace single ones, etc. The conventional trade-offs between model complexity and generality also apply, and so does Occam’s razor principle [23]. Normally the simplest method with satisfactory performance should be chosen. If performance is unknown, then another principle might be considered as a reference: a good algorithm should take into account both robustness and discrimination, and such a balance should be observed in the design.

Besides performance and complexity, flexibility is also an important factor. An algorithm should provide ways for performance trade-offs. A basic leverage is the hash length. In order to support different scenarios, a hash algorithm should be able to adjust the hash length. When comparing two hash algorithms, the conclusion might vary for different hash lengths and datasets, thus a thorough comparison typically

involves several hash lengths and datasets. When comparing a hash approach with a non-hash approach, it is important to have approximately equal entropy in the feature representations for a fair comparison.

9.2 Unsupervised Perceptual Hash Algorithms

Many perceptual hash algorithms are approximately linear. They have a simple structure like the following:

$$\mathbf{y} = \text{sgn}(\mathbf{W}^T \mathbf{x} + \mathbf{b}), \quad (9.10)$$

where $\text{sgn}(\cdot)$ is the element-wise sign function, $\mathbf{W} = [\mathbf{w}_1, \dots, \mathbf{w}_n]$ is an $m \times n$ matrix, and \mathbf{b} is an $n \times 1$ bias vector. In this form, each hash function in Eqn. (9.8) is a hyperplane in \mathbb{X}^m , represented by a column of \mathbf{W} , i.e., $\mathbf{w}_i^T \mathbf{x} + b_i = 0$. Existing approaches typically differ in the derivation of \mathbf{W} . Without learning, \mathbf{W} can be generated randomly, but the performance is limited. In this chapter we focus on computing \mathbf{W} from training data.

In the following, we first consider perceptual hash algorithms with unsupervised learning. In particular, spectral hashing [24], iterative quantization [25], k -means hashing [15], and kernelized locality sensitive hashing [26] are presented.

9.2.1 Spectral Hashing

Spectral hashing (SH) [24] is a hash algorithm based on spectral clustering. It is also related to the graph Laplacian and PCA [12]. This algorithm focuses on the robustness requirement, and enforces relevant multimedia objects to result in similar hash values by minimizing the correlation between the affinity matrix \mathbf{D}^p and the hash affinity matrix \mathbf{D}^h :

$$\min : \sum_{i,j} D_{ij}^p \cdot D_{ij}^h, \quad (9.11)$$

where $D_{ij}^p = \exp(\|\mathbf{x}_i - \mathbf{x}_j\|^2 / \epsilon^2)$ and $D_{ij}^h = \|\mathbf{y}_i - \mathbf{y}_j\|^2$. The parameter ϵ is a constant which defines similarity in the input space. This problem is converted to the following equivalent representation:

$$\min : \text{trace}(\mathbf{Y}^T (\mathbf{D} - \mathbf{D}^p) \mathbf{Y}), \quad (9.12)$$

where \mathbf{Y} is an $M \times n$ matrix whose i th row is \mathbf{y}_i^T and \mathbf{D} is a diagonal matrix with $D_{ii} = \sum_j D_{ij}^p$. This is an NP-hard problem [24]. By relaxing Constraint 0, the problem becomes easier, and the solutions are the n eigenvectors of $\mathbf{D} - \mathbf{D}^p$ with the smallest eigenvalues (excluding 0). Afterwards, the eigenvectors can be quantized by Eqn. (9.8) to obtain hash values.

The above method works for existing (training) data, but cannot compute hash values for new data. For unknown data, the Nystrom method [27] can be used for interpolation. However, that is inefficient when the training dataset is large. A more efficient way is to approximate the data with eigenfunctions. The data points \mathbf{x} are assumed to be samples

of a probability distribution $p(\mathbf{x})$. Then problem (9.11) becomes an expectation form:

$$\min : \int \|y(\mathbf{x}_1) - y(\mathbf{x}_2)\|^2 D^p(\mathbf{x}_1, \mathbf{x}_2) p(\mathbf{x}_1) p(\mathbf{x}_2) d\mathbf{x}_1 d\mathbf{x}_2, \quad (9.13)$$

where $D^p(\mathbf{x}_1, \mathbf{x}_2) = \exp(\|\mathbf{x}_1 - \mathbf{x}_2\|^2 / \epsilon^2)$. The corresponding Constraint 1 and Constraint 2 take the following forms:

$$\int y(\mathbf{x}) p(\mathbf{x}) d\mathbf{x} = \mathbf{0}, \quad (9.14)$$

$$\int y(\mathbf{x}) y(\mathbf{x})^T p(\mathbf{x}) d\mathbf{x} = \mathbf{I}. \quad (9.15)$$

Again by relaxing Constraint 0, the solutions to the new spectral problem are eigenfunctions of the weighted Laplace–Beltrami operators L_p defined on manifolds [28–32] with minimal eigenvalues. With proper normalization, the eigenvectors of the discrete graph Laplacian defined by sample points from $p(\mathbf{x})$ converge to eigenfunctions of L_p .

It is a special case when \mathbf{x} follows a separable distribution $\Pr(\mathbf{x}) = \prod_{i=1}^m u_i(x_i)$. If $u_i(x_i)$ is a uniform distribution over $[a, b]$, the eigenfunctions $\Psi_k(\mathbf{x})$ and eigenvalues λ_k take the following product forms:

$$\Psi_k(\mathbf{x}) = \prod_{i=1}^m \sin\left(\frac{\pi}{2} + \frac{k\pi}{b-a} x_i\right), \quad (9.16)$$

$$\lambda_k = \left[1 - \exp\left(-\frac{\epsilon^2}{2} \left|\frac{k\pi}{b-a}\right|^2\right)\right]^m. \quad (9.17)$$

Once the eigenfunctions are known, new data can be evaluated to generate hash values. To summarize, spectral hashing takes the following steps:

- Find the principal components of data by PCA.
- Approximate each principal direction with a uniform distribution and compute the eigenvalues using Eqn. (9.17).
- Select the eigenfunctions corresponding to the smallest n eigenvalues.
- A new data point is evaluated by the selected n eigenfunctions and the results are thresholded to derive a hash value.

Although data in practice typically do not follow a uniform distribution, experiments in [24] show acceptable results. Nevertheless, the optimization goal of SH is biased towards robustness, while discrimination is not considered, which limits the actual performance. Spectral hashing has been extended to multidimensional spectral hashing for improved data approximation. Details can be found in [33].

9.2.2 Iterative Quantization

Iterative quantization (ITQ) [25] is a hash algorithm closed related to PCA. This algorithm requires data to be zero-centered, i.e., $\sum_{i=1}^M \mathbf{x}_i = \mathbf{0}$, and uses a simple form of Eqn. (9.10):

$$\mathbf{Y} = \text{sgn}(\mathbf{W}^T \mathbf{X}), \quad (9.18)$$

where \mathbf{Y} is a matrix whose columns are hash values. In addition, the algorithm adopts a relaxed version of Constraint 2 by requiring the hyperplanes to be orthogonal, i.e., $\mathbf{W}^T \mathbf{W} = \mathbf{I}$.

The initial goal of ITQ is to maximize the variance of each bit. The corresponding objective function is:

$$\max : \sum_i \text{var}(h_i(\mathbf{x})) = \sum_i \text{var}(\text{sgn}(\mathbf{w}_i^T \mathbf{x})). \quad (9.19)$$

By relaxing Constraint 0, the above objective function can be converted to

$$\begin{aligned} \max : \sum_i E\{(\mathbf{w}_i^T \mathbf{x})^2\} &= \frac{1}{M} \sum_i \mathbf{w}_i^T \mathbf{X} \mathbf{X}^T \mathbf{w}_i \\ &= \frac{1}{M} \text{trace}(\mathbf{W}^T \mathbf{X} \mathbf{X}^T \mathbf{W}). \end{aligned} \quad (9.20)$$

The solution to the relaxed problem is the top n eigenvectors of the data covariance matrix $\mathbf{X} \mathbf{X}^T$.

After an initial \mathbf{W} is obtained, the next step is to minimize the quantization loss $\|\mathbf{W}^T \mathbf{X} - \text{sgn}(\mathbf{W}^T \mathbf{X})\|_F^2$, where F denotes the Frobenius norm. This is achieved by multiplying \mathbf{W} with an $n \times n$ orthogonal matrix \mathbf{R} . Geometrically, this means the projected data is rotated before quantization. According to Eqn. (9.20), if \mathbf{W} is an optimal solution, then so is $\mathbf{W} \mathbf{R}$. Therefore, the rotation does not influence the maximal variance criterion. The new objective function is:

$$\begin{aligned} \min : Q(\mathbf{Y}, \mathbf{R}) &= \|\mathbf{Y} - \mathbf{R}^T \mathbf{W}^T \mathbf{X}\|_F^2 \\ &= \|\mathbf{Y}\|_F^2 + \|\mathbf{R}^T \mathbf{W}^T \mathbf{X}\|_F^2 - 2 \cdot \text{trace}(\mathbf{Y} \mathbf{X}^T \mathbf{W} \mathbf{R}). \end{aligned} \quad (9.21)$$

Since $\|\mathbf{Y}\|_F^2$ and $\|\mathbf{R}^T \mathbf{W}^T \mathbf{X}\|_F^2 = \|\mathbf{W}^T \mathbf{X}\|_F^2$ are constant, the above optimization is equivalent to:

$$\max : \text{trace}(\mathbf{Y} \mathbf{X}^T \mathbf{W} \mathbf{R}). \quad (9.22)$$

The optimal \mathbf{R} can be obtained by a two-step alternating procedure:

- 1) Fix \mathbf{R} and update \mathbf{Y} .
- 2) Fix \mathbf{Y} and update \mathbf{R} .

The algorithm is initialized with a random orthogonal \mathbf{R} . The first step obtains $\mathbf{Y} = \text{sgn}(\mathbf{X}^T \mathbf{W} \mathbf{R})$. In the second step, a new optimal \mathbf{R} is found by singular value decomposition of the $m \times m$ matrix $\mathbf{S} \mathbf{S}^T$, where $\mathbf{S} = \mathbf{Y} \mathbf{X}^T \mathbf{W}$. If $\mathbf{S} \mathbf{S}^T = \mathbf{U} \mathbf{D} \mathbf{U}^T$, then $\mathbf{R} = \mathbf{U} \mathbf{U}^T$. This is a generalized orthogonal Procrustes problem [34]. The alternating procedure is repeated until \mathbf{R} converges.

ITQ does not make any assumption on data distribution, thus it is versatile and serves as a proper baseline. ITQ not only works with PCA, but also supports other kinds of analysis. An extension is based on canonical correlation analysis (CCA), which is a supervised version. Details can be found in [35].

9.2.3 K-Means Hashing

K -means hashing (KMH) [15] is a hash algorithm based on the k -means clustering [13]. The basic idea is to quantize feature vectors in a k -means manner and to approximate

the feature distance by the Hamming distance between cell indices. The hash algorithm $H(\cdot)$ is a quantization procedure:

$$\mathbf{y} = H(\mathbf{x}) = \{i | q(\mathbf{x}) = \mathbf{c}_i\}, \quad (9.23)$$

where $q(\cdot)$ is a vector quantization function [36] and \mathbf{c}_i is the center of the i th quantization cell. The affinity matrix \mathbf{D}^p can be represented as

$$D_{ij}^p = d(q(\mathbf{x}_i), q(\mathbf{x}_j)), \quad (9.24)$$

where $d(\cdot)$ means the Euclidean distance. The hash affinity matrix \mathbf{D}^h is defined as

$$D_{ij}^h = s \cdot d_H(\mathbf{y}_i, \mathbf{y}_j)^{1/2} = s \cdot d_H(i, j)^{1/2}, \quad (9.25)$$

where s is a constant scale factor and $d_H(\cdot)$ means the Hamming distance. The goal of KMH is to assign hash values to the cluster centers so that the affinity error is minimized:

$$\min : E_{\text{aff}} = \frac{1}{M^2} \|\mathbf{D}^p - \mathbf{D}^h\|_F^2. \quad (9.26)$$

When $k = 2^n < M$, it is more practical to compute the following alternative form

$$E_{\text{aff}} = \sum_{i=1}^k \sum_{j=1}^k w_{ij} (d(\mathbf{c}_i, \mathbf{c}_j) - d_H(i, j))^2, \quad (9.27)$$

where $w_{ij} = M_i M_j / M^2$, and M_i and M_j are the numbers of samples having indices i and j respectively. In addition, the algorithm also aims to minimize the quantization error:

$$E_{\text{quan}} = \frac{1}{M} \sum_{i=1}^M (\mathbf{x} - q(\mathbf{x}))^2. \quad (9.28)$$

These two terms are taken into account together by the following objective function

$$E = E_{\text{quan}} + \lambda E_{\text{aff}}, \quad (9.29)$$

where λ is a fixed weight factor. The above objective function is minimized in a two-step alternating procedure:

- 1) Fix $\{\mathbf{c}_i\}$ and optimize $H(\mathbf{x})$.
- 2) Fix $H(\mathbf{x})$ and optimize $\{\mathbf{c}_i\}$.

The first step is like conventional vector quantization. In the second step, each code-word \mathbf{c}_j is sequentially updated (with others fixed) according to the following objective function:

$$\begin{aligned} \mathbf{c}_j = \arg \min_{\mathbf{c}_j} & \left(\frac{1}{M} \sum_{\mathbf{x}: H(\mathbf{x})=j} (\mathbf{x} - \mathbf{c}_j)^2 \right. \\ & \left. + \lambda \sum_{i: i \neq j} w_{ij} (d(\mathbf{c}_i, \mathbf{c}_j) - d_H(i, j))^2 \right). \end{aligned} \quad (9.30)$$

This can be solved by the quasi-Newton method [37].

In practice, when the number of feature dimensions m is too large, it is not feasible to directly perform vector quantization. Product quantization [14] can be used instead. The idea is to divide feature dimensions into small groups so that vector quantization is feasible for each group. Then KMH can be applied to each group.

KMH is similar to fast product quantization, but is even faster during retrieval. Compared with hyperplane-based algorithms, KMH can be considered as a “centroid”-based algorithm. Since the number of centroids is larger than the number of hyperplanes for the same space partitioning, KMH might be slower than other algorithms during hash generation.

9.2.4 Kernelized Locality Sensitive Hashing

Kernelized locality sensitive hashing (KLSH) [26, 38] is an extension of locality sensitive hashing [39]. Conventional hash algorithms look for hyperplanes to separate multimedia objects in a feature space. Nevertheless, sometimes it is better to represent feature data in a higher dimensional space, where originally inseparable data points might become separable. This involves a kernel representation [12]. In KLSH, the perceptual affinity is measured in a kernel-induced feature space:

$$\mathbf{D}^p = [K(\mathbf{x}_i, \mathbf{x}_j)] = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j). \quad (9.31)$$

The basic form of a hash function $h_i(\cdot)$ for KLSH is

$$y_i = h_i(\mathbf{x}) = \text{sgn}(\mathbf{r}^T \phi(\mathbf{x})), \quad (9.32)$$

where \mathbf{r} is a p -dimensional hyperplane in the kernel space. The goal of KLSH is to construct \mathbf{r} in terms of kernel space instances. The construction takes the following form:

$$\mathbf{z}_t = \frac{1}{t} \sum_{i \in S} \phi(\mathbf{x}_i), \quad (9.33)$$

where S is a set of t randomly chosen data point indices. The desired \mathbf{r} can then be obtained by normalizing \mathbf{z}_t :

$$\mathbf{r} = \Delta^{-1/2} \tilde{\mathbf{z}}_t, \quad (9.34)$$

where $\tilde{\mathbf{z}}_t = \sqrt{t}(\mathbf{z}_t - \boldsymbol{\mu})$. According to the central limit theorem, the random vector $\tilde{\mathbf{z}}_t$ follows a multi-variate Gaussian distribution $N(\mathbf{0}, \Delta)$. By decomposing Δ into $\Delta = \mathbf{V}\Lambda\mathbf{V}^T$, the hyperplane \mathbf{r} can be represented as

$$\begin{aligned} \mathbf{r} &= \mathbf{V}\Lambda^{-1/2}\mathbf{V}^T\tilde{\mathbf{z}}_t \\ &= \sum_{k=1}^p \frac{1}{\sqrt{\theta_k}} \mathbf{v}_k \mathbf{v}_k^T \tilde{\mathbf{z}}_t, \end{aligned} \quad (9.35)$$

where θ_k, \mathbf{v}_k are the k th eigenvalue and eigenvector of the covariance matrix Δ . When data is zero-centered, \mathbf{v}_k can be represented as a weighted sum of p “anchor points” $\{\phi(\mathbf{x}_i^A)\}_{i=1}^p$ in the kernel space:

$$\mathbf{v}_k = \sum_{i=1}^p \frac{1}{\sqrt{\theta_k}} \mathbf{u}_k(i) \phi(\mathbf{x}_i^A), \quad (9.36)$$

where \mathbf{u}_k is the k th eigenvector of the kernel affinity matrix \mathbf{D}^p and $\{\mathbf{x}_i^A\}$ are the corresponding anchors in the feature space. Putting everything together,

$$\mathbf{r} = \sum_{i=1}^p w(i) \phi(\mathbf{x}_i^A), \quad (9.37)$$

where $w(i)$ represents a weight factor

$$w(i) = \sum_{j=1}^p \sum_{k=1}^p \frac{1}{\theta_k^{3/2}} \mathbf{u}_k(i) \mathbf{u}_k(j) \phi(\mathbf{x}_j^A)^T \tilde{\mathbf{z}}_t. \quad (9.38)$$

More details can be found in [38].

Compared with other learning-based hash algorithms, KLSH has a smaller degree of data orientation, as only the anchor points are data-dependent. Although kernel representation potentially improves data separability, the hyperplanes are random. This is a major performance limit.

9.3 Supervised Perceptual Hash Algorithms

Unsupervised perceptual hash algorithms capture structure information in the feature data representation. They are versatile, but typically do not involve any “semantic” of data. In order to take semantics into account, supervised algorithms are needed. In this section, some supervised perceptual hash algorithms are presented, including semi-supervised hashing [40], kernel-based supervised hashing [16], restricted Boltzmann machine based hashing [1], and supervised semantic-preserving deep hashing [41].

9.3.1 Semi-Supervised Hashing

Semi-supervised hashing (SSH) [40] is a hash algorithm that takes both semantic relevance and “maximal bit variance” into account. In particular, there are two training datasets, corresponding to the two criteria. The elements of the affinity matrix \mathbf{D}^p are defined as

$$D_{ij}^p = \begin{cases} 1, & \text{if } (\mathbf{x}_i, \mathbf{x}_j) \in \mathbb{M} \\ -1, & \text{if } (\mathbf{x}_i, \mathbf{x}_j) \in \mathbb{C} \\ 0, & \text{otherwise} \end{cases}, \quad (9.39)$$

where \mathbb{M} denotes a set of relevant object pairs and \mathbb{C} denotes a set of irrelevant object pairs. Recall that relevant objects are expected to have similar hash values and irrelevant objects are expected to have dissimilar hash values. SSH tries to maximize the empirical measure of accuracy:

$$\max : J(H) = \sum_{(\mathbf{x}_i, \mathbf{x}_j) \in \mathbb{M}} \mathbf{y}_i^T \mathbf{y}_j - \sum_{(\mathbf{x}_i, \mathbf{x}_j) \in \mathbb{C}} \mathbf{y}_i^T \mathbf{y}_j. \quad (9.40)$$

Assuming that the data is zero-centered and the hash algorithm takes an approximately linear structure $\mathbf{Y} = H(\mathbf{X}) = \text{sgn}(\mathbf{W}^T \mathbf{X})$, the objective function can be represented as

$$\begin{aligned} J(H) &= \frac{1}{2} \text{tr}(\mathbf{Y}_{t1} \mathbf{D}^p \mathbf{Y}_{t1}^T) \\ &= \frac{1}{2} \text{tr}(\text{sgn}(\mathbf{W}^T \mathbf{X}_{t1}) \mathbf{D}^p \text{sgn}(\mathbf{W}^T \mathbf{X}_{t1})^T), \end{aligned} \quad (9.41)$$

where t_1 denotes the first training set. By relaxing Constraint 0, a new objective function is obtained in terms of \mathbf{W} :

$$J(\mathbf{W}) = \frac{1}{2} \text{tr}(\mathbf{W}^T \mathbf{X}_{t_1} \mathbf{D}^p \mathbf{X}_{t_1}^T \mathbf{W}). \quad (9.42)$$

In addition, a regularization term is introduced to maximize the variance of each hash bit:

$$\begin{aligned} R(\mathbf{W}) &= \frac{1}{\beta} \sum_i E[(\mathbf{w}_i^T \mathbf{x})^2] = \frac{1}{\beta M} \sum_i \mathbf{w}_i^T \mathbf{X}_{t_2} \mathbf{X}_{t_2}^T \mathbf{w}_i \\ &= \frac{1}{\beta M} \text{tr}(\mathbf{W}^T \mathbf{X}_{t_2} \mathbf{X}_{t_2}^T \mathbf{W}), \end{aligned} \quad (9.43)$$

where t_2 denotes the second training set and β is an upper bound of the norm of training data. Combining Eqns. (9.42) and (9.43), the final objective function is the sum of the two:

$$\begin{aligned} J'(\mathbf{W}) &= J(\mathbf{W}) + R(\mathbf{W}) \\ &= \frac{1}{2} \text{tr}(\mathbf{W}^T \mathbf{X}_{t_1} \mathbf{D}^p \mathbf{X}_{t_1}^T \mathbf{W}) + \frac{\eta}{2} \text{tr}(\mathbf{W}^T \mathbf{X}_{t_2} \mathbf{X}_{t_2}^T \mathbf{W}) \\ &= \frac{1}{2} \text{tr}(\mathbf{W}^T \mathbf{M} \mathbf{W}), \end{aligned} \quad (9.44)$$

where $\mathbf{M} = \mathbf{X}_{t_1} \mathbf{D}^p \mathbf{X}_{t_1}^T + \eta \mathbf{X}_{t_2} \mathbf{X}_{t_2}^T$ is called the adjusted covariance matrix, and parameters M and β are absorbed into η . In order to make each hash bit independent from others, one can require \mathbf{W} to be orthogonal, i.e., $\mathbf{W}^T \mathbf{W} = \mathbf{I}$. In this case, maximizing Eqn. (9.44) becomes an eigenproblem. The solution \mathbf{W} consists of the top n eigenvectors with the largest eigenvalues.

The orthogonality constraint can be problematic in practice because orthogonal directions with low variance lead to hash bits of low entropy. Sometimes it is better to choose directions with high variance and low empirical error. A further improved objective function takes into account a penalty term for non-orthogonality

$$\begin{aligned} J''(\mathbf{W}) &= J'(\mathbf{W}) - \frac{\rho}{2} \|\mathbf{W}^T \mathbf{W} - \mathbf{I}\|_F^2 \\ &= \frac{1}{2} \text{tr}(\mathbf{W}^T \mathbf{M} \mathbf{W}) - \frac{\rho}{2} \text{tr}[(\mathbf{W}^T \mathbf{W} - \mathbf{I})^T (\mathbf{W}^T \mathbf{W} - \mathbf{I})]. \end{aligned} \quad (9.45)$$

By taking the derivative, one gets

$$\frac{\partial J''(\mathbf{W})}{\partial \mathbf{W}} = \mathbf{0} \Rightarrow \left(\mathbf{W} \mathbf{W}^T - \mathbf{I} - \frac{1}{\rho} \mathbf{M} \right) \mathbf{W} = \mathbf{0}. \quad (9.46)$$

A solution must fulfil the equation

$$\mathbf{W} \mathbf{W}^T \mathbf{W} = \mathbf{Q} \mathbf{W}, \quad (9.47)$$

where $\mathbf{Q} = \mathbf{I} + \frac{1}{\rho} \mathbf{M}$. It is shown that \mathbf{Q} is positive-definite when $\rho > -\lambda_{\min}$, where λ_{\min} denotes the smallest eigenvalue of \mathbf{M} . In that case, \mathbf{W} can be found from the singular value decomposition of \mathbf{Q} :

$$\begin{aligned} \mathbf{Q} &= \mathbf{I} + \mathbf{U} \text{diag} \left(\frac{\lambda_1}{\rho}, \dots, \frac{\lambda_m}{\rho} \right) \mathbf{U}^T \\ &= \mathbf{U} \text{diag} \left(\frac{\lambda_1}{\rho} + 1, \dots, \frac{\lambda_m}{\rho} + 1 \right) \mathbf{U}^T, \end{aligned} \quad (9.48)$$

where \mathbf{U} and $\{\lambda_i\}$ are the eigenvectors and eigenvalues of \mathbf{M} . In practice, \mathbf{W} is truncated:

$$\mathbf{W} = \mathbf{U}_k \mathbf{\Lambda}_k^{1/2}, \quad (9.49)$$

where \mathbf{U}_k denotes the top k eigenvectors of \mathbf{M} and $\mathbf{\Lambda}_k^{1/2}$ is the corresponding diagonal matrix with $\{\sqrt{\frac{\lambda_i}{\rho} + 1}\}_{i=1}^k$.

SSH is an example of hash generation based on multiple constraints, so the performance depends on the properness of the constraints. In addition, it is not straight-forward to decide the corresponding weights when there are multiple objective functions. More extensions have been made to compute \mathbf{W} in a sequential way. The details can be found in [42].

9.3.2 Kernel-Based Supervised Hashing

Kernel-based supervised hashing (KSH) [16] is a supervised hash algorithm based on a kernel representation. The hash function is defined as

$$h(\mathbf{x}) = \text{sgn}(f(\mathbf{x}) - b), \quad (9.50)$$

where $f(\mathbf{x}) = \mathbf{w}^T \mathbf{k}(\mathbf{x}) = \sum_{i=1}^p k(\mathbf{x}_i^A, \mathbf{x}) w_i$. Note that the data \mathbf{x} is transformed to the kernel space representation $\mathbf{k}(\mathbf{x})$, using a kernel function $k(\cdot)$ and p anchor points $\{\mathbf{x}_i^A\}_{i=1}^p$. Ideally, b should be the median of training data in the kernel space. For simplicity, mean thresholding is used by defining

$$b = \frac{1}{M} \sum_{j=1}^M f(\mathbf{x}_j) = \frac{1}{M} \sum_{j=1}^M \sum_{i=1}^p k(\mathbf{x}_i^A, \mathbf{x}_j) w_i. \quad (9.51)$$

Thus the hash function has the form:

$$h(\mathbf{x}) = \text{sgn}(\mathbf{w}^T \bar{\mathbf{k}}(\mathbf{x})), \quad (9.52)$$

where $\bar{\mathbf{k}}(\mathbf{x}) = \mathbf{k}(\mathbf{x}) - \frac{1}{M} \sum_{j=1}^M \mathbf{k}(\mathbf{x}_j)$.

The affinity matrix \mathbf{D}^p is defined the same way as in Eqn. (9.39). The goal of KSH is to approximate \mathbf{D}^p with the hash affinity matrix \mathbf{D}^h based on the training data. In particular, the Hamming distance is used for \mathbf{D}^h and expressed in terms of the inner product

$$\min : \|\mathbf{D}^h - \mathbf{D}^p\|_F^2 = \left\| \frac{1}{n} \mathbf{Y}^T \mathbf{Y} - \mathbf{D}^p \right\|_F^2, \quad (9.53)$$

where \mathbf{Y} is the hash value matrix and the second step is because of Eqn. (9.9). Since $\mathbf{Y} = \text{sgn}(\mathbf{W}^T \bar{\mathbf{K}})$ (where $\bar{\mathbf{K}}$ is the matrix form of $\bar{\mathbf{k}}(\mathbf{x})$), one can get the following objective function:

$$\min : \left\| \frac{1}{n} (\text{sgn}(\mathbf{W}^T \bar{\mathbf{K}}))^T \text{sgn}(\mathbf{W}^T \bar{\mathbf{K}}) - \mathbf{D}^p \right\|_F^2. \quad (9.54)$$

By considering each hash function $h_i(\mathbf{x}) = \text{sgn}(\mathbf{w}_i^T \bar{\mathbf{K}})$ separately, the above objective is equivalent to

$$\min : \left\| \sum_{i=1}^n r_i - n \mathbf{D}^p \right\|_F^2, \quad (9.55)$$

where $r_i = (\text{sgn}(\mathbf{w}_i^T \bar{\mathbf{K}}))^T \text{sgn}(\mathbf{w}_i^T \bar{\mathbf{K}})$. By defining a residue matrix

$$\mathbf{R}_k = n\mathbf{D}^p - \sum_{i=1}^k (\text{sgn}(\mathbf{w}_i^T \bar{\mathbf{K}}))^T \text{sgn}(\mathbf{w}_i^T \bar{\mathbf{K}}), \quad (9.56)$$

\mathbf{w}_k can be pursued by minimizing the following cost

$$\begin{aligned} & \|\mathbf{r}_k - \mathbf{R}_{k-1}\|_F^2 \\ &= \text{tr}[(\mathbf{r}_k - \mathbf{R}_{k-1})(\mathbf{r}_k - \mathbf{R}_{k-1})^T] \\ &= \text{tr}[\mathbf{r}_k \mathbf{r}_k^T + \mathbf{R}_{k-1} \mathbf{R}_{k-1}^T - 2\mathbf{r}_k \mathbf{R}_{k-1}^T] \\ &= \text{tr}[-2\mathbf{r}_k \mathbf{R}_{k-1}^T] + \text{const} \\ &= -2\text{tr}[(\text{sgn}(\mathbf{w}_k^T \bar{\mathbf{K}}))^T \text{sgn}(\mathbf{w}_k^T \bar{\mathbf{K}}) \mathbf{R}_{k-1}^T] + \text{const} \\ &= -2\text{sgn}(\mathbf{w}_k^T \bar{\mathbf{K}}) \mathbf{R}_{k-1}^T (\text{sgn}(\mathbf{w}_k^T \bar{\mathbf{K}}))^T + \text{const}. \end{aligned} \quad (9.57)$$

By relaxing Constraint 0, the minimization problem becomes the following maximization problem:

$$\max : g(\mathbf{w}_k) = \mathbf{w}_k^T \bar{\mathbf{K}} \mathbf{R}_{k-1}^T (\mathbf{w}_k^T \bar{\mathbf{K}})^T = \mathbf{w}_k^T \bar{\mathbf{K}} \mathbf{R}_{k-1}^T \bar{\mathbf{K}}^T \mathbf{w}_k. \quad (9.58)$$

By enforcing $\mathbf{w}_k^T \bar{\mathbf{K}} \bar{\mathbf{K}}^T \mathbf{w}_k = p$, \mathbf{w}_k can be solved as the generalized eigenvector corresponding to the largest generalized eigenvalue for the problem $\mathbf{A}\mathbf{w} = \lambda\mathbf{B}\mathbf{w}$, where $\mathbf{A} = \bar{\mathbf{K}} \mathbf{R}_{k-1}^T \bar{\mathbf{K}}^T$, $\mathbf{B} = \bar{\mathbf{K}} \bar{\mathbf{K}}^T$, and \mathbf{w} and λ represent the generalized eigenvectors and eigenvalues. The solved eigenvector is scaled to have a proper norm and saved as \mathbf{w}_k^0 for further optimization.

A smooth surrogate of $g(\mathbf{w}_k)$ is defined as

$$\tilde{g}(\mathbf{w}_k) = \varphi(\mathbf{w}_k^T \bar{\mathbf{K}}) \mathbf{R}_{k-1}^T (\varphi(\mathbf{w}_k^T \bar{\mathbf{K}}))^T, \quad (9.59)$$

where $\varphi(x) = 2/(1 + e^{-x}) - 1$ is an element-wise sigmoid-shaped function. The derivative of $\tilde{g}(\mathbf{w}_k)$ can be computed as

$$\frac{\partial \tilde{g}}{\partial \mathbf{w}_k} = \bar{\mathbf{K}}((\mathbf{R}_{k-1} \mathbf{b}) \odot (\mathbf{1} - \mathbf{b} \odot \mathbf{b})), \quad (9.60)$$

where \odot denotes the element-wise (Hadamard) product, $\mathbf{b} = \varphi(\mathbf{w}_k^T \bar{\mathbf{K}})$, and $\mathbf{1}$ is a constant vector of p “1”s. Regular gradient descent techniques such as Nesterov’s method [43] can be used to find a local optimum.

Compared with KLSH, KSH not only takes advantage of the kernel representation, but also actively seeks the best hyperplanes. Although there is only a single objective function, a balance is implicitly achieved between robustness and discrimination by enforcing the hash affinity matrix to approximate the affinity matrix. Using greedy sequential optimization and smooth surrogates is also useful practice. Thus KSH generally outperforms other algorithms of the same complexity.

9.3.3 Restricted Boltzmann Machine-Based Hashing

Previously presented hash algorithms are approximately linear. Linear algorithms are efficient and easy to implement. On the other hand, they are limited by the simple structure. As computing power grows, nonlinear approaches are receiving more and

more attention. A representative nonlinear approach is the neural network [44]. A neural network is an ensemble of nodes organized as layers. Each node functions like a neuron. Compared with linear approaches, neural networks are better at modelling complex semantics. In the following, two hash algorithms based on neural networks are presented.

The restricted Boltzmann machine (RBM) is a typical neural network structure [45]. It can capture higher-order correlations in input data. Multiple layers of RBMs can be used for dimensionality reduction [46]. An RBM is a network of random binary units arranged in two layers, one visible and one hidden. Units \mathbf{v} in the visible layer are connected via a set of symmetric weights \mathbf{W} to units \mathbf{h} in the hidden layer. The “energy” of the link between \mathbf{v} and \mathbf{h} is defined as

$$E(\mathbf{v}, \mathbf{h}) = - \sum_{i \in \text{visible}} b_i v_i - \sum_{j \in \text{hidden}} b_j h_j - \sum_{i,j} v_i h_j w_{ij}, \quad (9.61)$$

where v_i and h_j denote the binary states of i th visible and j th hidden units; w_{ij} , b_i/b_j are weight and bias parameters, respectively. The probability of a particular input vector \mathbf{v} is defined as

$$p(\mathbf{v}) = \sum_{\mathbf{h}} \frac{\exp(-E(\mathbf{v}, \mathbf{h}))}{\sum_{\mathbf{u}, \mathbf{g}} \exp(-E(\mathbf{u}, \mathbf{g}))}. \quad (9.62)$$

Since units in the same layer are assumed to be independent, given the state of one layer the conditional probability of the other layer can be derived as a product of Bernoulli distributions:

$$p(h_j = 1 | \mathbf{v}) = \sigma \left(b_j + \sum_i w_{ij} v_i \right), \quad (9.63)$$

$$p(v_i = 1 | \mathbf{h}) = \sigma \left(b_i + \sum_j w_{ij} h_j \right), \quad (9.64)$$

where $\sigma(x) = 1/(1 + e^{-x})$ is a logistic function. The goal of using RBMs is to map relevant objects to similar binary outputs (hash values). The parameters w_{ij} , b_i , and b_j are learned in two stages:

- 1) unsupervised pre-training
- 2) supervised fine-tuning.

Both steps update the parameters w_{ij} , b_i , and b_j via gradient descent, but with different objectives. In the first step, the network tries to learn the data distribution (without label information) by minimizing the contrastive divergence [47]. An approximation of the gradient of the objective function for w_{ij} is defined as

$$\Delta w_{ij} = \epsilon \frac{\partial \log p(\mathbf{v})}{\partial w_{ij}} = \epsilon (E[v_i h_j]_{\text{data}} - E[v_i h_j]_{\text{recon}}), \quad (9.65)$$

where ϵ is the learning rate, and $E[\cdot]_{\text{data}}$ and $E[\cdot]_{\text{recon}}$ represent the expectation over training and reconstructed data, respectively. The objective functions for b_i and b_j are defined in a similar way. The pre-training is sequentially carried out for each RBM in

the stack, with the output of one RBM becoming the input of the next. The number of output units is gradually decreased to achieve dimension reduction. In the second step, the RBM stack is symmetrically “unfolded” to become a multi-layer autoencoder, which is fine-tuned by back-propagation [44] with label information. The objective function depends on the application. In [1], the objective is defined according to neighbourhood component analysis [48, 49]. Given the training data $\{\mathbf{x}_i, t_i\}_{i=1}^M$, the goal is to preserve the semantic neighborhood structure by maximizing the expected number of correctly classified neighbours around each query that have the same class labels:

$$\max : O_{\text{NCA}} = \sum_{i=1}^M \sum_{j: t_i=t_j} = \frac{\exp(-\|f(\mathbf{x}_i|\mathbf{W}) - f(\mathbf{x}_j|\mathbf{W})\|^2)}{\sum_{k \neq i} \exp(-\|f(\mathbf{x}_i|\mathbf{W}) - f(\mathbf{x}_k|\mathbf{W})\|^2)}, \quad (9.66)$$

where $f(\mathbf{x}|\mathbf{W})$ is the projection of data point \mathbf{x} via the network with parameters \mathbf{W} , i.e., the hash value.

Originally, RBMs only supported binary inputs and outputs, which meant multimedia data had to be converted to binary first. In order to support real-valued inputs, the first (input) layer of an RBM stack can be modified to have Gaussian distributed units, with the mean determined by hidden units:

$$p(v_i = x|\mathbf{h}) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{x - b_i - \sigma_i \sum_j w_{ij} h_j}{2\sigma_i^2}\right), \quad (9.67)$$

$$p(h_j = 1|\mathbf{v}) = \sigma\left(b_j + \sum_i w_{ij} \frac{v_i}{\sigma_i}\right). \quad (9.68)$$

More details can be found in [49].

RBM and other neural network based hash algorithms typically require more training data than conventional hash algorithms because there are more parameters for tuning. Sometimes the training cost can be prohibitive. Since RBMs still use hand-crafted features, the latter might become a performance bottleneck. Nevertheless, RBMs serve as a good reference for studying more recent algorithms.

9.3.4 Supervised Semantic-Preserving Deep Hashing

The previously presented RBM is an early version of neural networks, which are mainly considered as classification techniques. Typically, the network input is a feature vector [1]. More recent neural networks no longer rely on existing features because they combine feature extraction and classification into a unified framework. They differ from conventional networks mainly in two aspects:

- they are deep, i.e., have many layers
- new components, such as convolutional layers and rectified linear units, are introduced.

Typically, the output units are with the softmax functions and the network is trained to maximize the multinomial logistic regression objective function for multi-class classification. Representative examples include AlexNet (see chapter 2, ref. [10]), VGG [50], and ResNet [51].

Supervised semantic-preserving deep hashing (SSDH) [41] is a hash algorithm based on latest deep learning [44] architectures. In order to generate n -bit hash values, a latent layer of n units is added before the output layer. The latent layer is fully connected to the neighbouring layers. Sigmoid activation is used. A hash function is defined as

$$h(\mathbf{x}) = \text{sgn}(\sigma(\mathbf{w}^T \mathbf{x} + b) - 0.5), \quad (9.69)$$

where $\sigma(z) = 1/(1 + \exp(-z))$, $z \in \mathbb{R}$ is the sigmoid logistic function. SSDH tries to ensure that semantically similar images are mapped to similar hash values. The objective function is:

$$\min_{\mathbf{W}} : \{E_1(\mathbf{W}) = \sum_{i=1}^M L(\mathbf{t}_i, \hat{\mathbf{t}}_i) + \lambda \|\mathbf{W}\|^2\}, \quad (9.70)$$

where $L(\cdot)$ represents the loss for the true label vector \mathbf{t}_i and the predicted label vector $\hat{\mathbf{t}}_i$. In order to support different types of label information, the loss function takes the general form:

$$L(\mathbf{t}_i, \hat{\mathbf{t}}_i) = - \sum_{j=1}^K l(t_{ij}, \hat{t}_{ij}), \quad (9.71)$$

where $l(\cdot)$ depends on the application. For single-label classification,

$$l(t_{ij}, \hat{t}_{ij}) = t_{ij} \ln(\hat{t}_{ij}). \quad (9.72)$$

For multi-label classification,

$$l(t_{ij}, \hat{t}_{ij}) = \begin{cases} 0, & \text{if } t_{ij} = 1 \text{ and } \hat{t}_{ij} \geq 1 \\ 0, & \text{if } t_{ij} = 0 \text{ and } \hat{t}_{ij} \leq 0 \\ \frac{1}{2} |t_{ij} - \hat{t}_{ij}|^p, & \text{otherwise} \end{cases}, \quad (9.73)$$

where $p \in \{1, 2\}$. Such a loss function actually implements a linear L1-norm (or L2-norm) support vector machine (SVM) thresholded at 0.5.

In addition, two more constraints are enforced:

$$\max : \left\{ E_2(\mathbf{W}) = \frac{1}{n} \sum_{i=1}^M \left\| \mathbf{a} - \frac{1}{2} \mathbf{1} \right\|^p \right\}, \quad (9.74)$$

$$\min : \left\{ E_3(\mathbf{W}) = \sum_{i=1}^M \left| \frac{1}{n} \mathbf{a}^T \mathbf{1} - 0.5 \right|^p \right\}, \quad (9.75)$$

where $\mathbf{a} = [a_1, \dots, a_n]^T$, $a_i = \sigma(\mathbf{w}_i^T \mathbf{x} + b)$ is the activation of i th latent unit, and $\mathbf{1}$ is a vector of "1"s. The first constraint makes the activation output close to 0 or 1. The second constraint aims for equal probable bits. Combining all three constraints, the overall objective function is

$$\min_{\mathbf{W}} : \alpha E_1(\mathbf{W}) - \beta E_2(\mathbf{W}) + \gamma E_3(\mathbf{W}), \quad (9.76)$$

where α, β , and γ are weight factors. Optimal parameters \mathbf{W} of the latent layer can be found by back-propagation and stochastic gradient descent [44].

SSDH is a flexible framework because it can be implemented by inserting a latent layer to almost any existing neural network. Thus the performance depends on the underlying

network. With the advancement of deep learning, SSDH can be a cost-effective approach to derive a well-performing perceptual hash algorithm.

9.4 Constructing Perceptual Hash Algorithms

Ever since perceptual hashing raised interest in large-scale multimedia retrieval, many algorithms have been proposed. Those presented in the previous sections are just representative examples of a big selection. Nevertheless, novel algorithms are still being proposed. According to the classical structure, new algorithms could differ in any of the algorithm components, i.e., feature extraction, feature transformation, feature quantization, etc. A natural question here is whether there exists a single best hash algorithm. The answer is “no”, because there is no best feature. Hash comparison is essentially a classification procedure: two multimedia objects are classified either as relevant or as irrelevant. According to the no-free-lunch theorem [52, 53], there is no best classifier. Each realization achieves a trade-off between robustness and discrimination.

Given the existence of many hash algorithms, an interesting intuition is that perhaps there is no need to design new algorithms from scratch – one can simply take components from different algorithms and combine them into a better algorithm. This is the topic of this section. We focus on two approaches:

- two-step hashing [54]
- hash bit selection [55, 56].

What they have in common is that they both consider hash bits as binary classifiers. An n -bit hash value is considered as n binary classifiers.

9.4.1 Two-Step Hashing

Two-step hashing (TSH) [54] is a flexible framework that decomposes hash algorithm design into two steps: a hash value inference step and a hash function learning step. Given the training data, a general objective function is defined as follows:

$$\min_{h(\cdot)} : \sum_{i,j} \delta_{ij} L(h(\mathbf{x}_i), h(\mathbf{x}_j); D_{ij}^p), \quad (9.77)$$

where $L(\cdot)$ is a loss function and $\delta_{ij} \in \{0, 1\}$ is an indicator that indicates if the relationship between objects i and j is defined (“1”) or not (“0”). This formulation summarizes many conventional approaches. They first define a particular form of the hash function as well as the loss function, then use optimization to find the function parameters. This is typically difficult because the two issues are coupled.

Given the loss function, instead of finding hash functions directly, one can first find the optimal hash values \mathbf{Y} by minimization:

$$\min_{\mathbf{Y}} : \sum_{i,j} \delta_{ij} L(\mathbf{y}_i, \mathbf{y}_j; D_{ij}^p). \quad (9.78)$$

After \mathbf{Y} is known, the next step is to find the suitable hash algorithm $H(\cdot)$:

$$\min_{H(\cdot)} : \sum_i F(\mathbf{y}_i; H(\mathbf{x}_i)), \quad (9.79)$$

where $F(\cdot)$ is another loss function. By considering each bit individually, the above objective function becomes

$$\min_{h_k(\cdot)} : \sum_i F'(y_{ik}; h_k(\mathbf{x}_i)), \quad (9.80)$$

where $F'(\cdot)$ is a bit level loss function. Overall, TSH consists of the following two steps:

- 1) Solve the optimization problem in Eqn. (9.78) to obtain hash values for training data.
- 2) Solve the binary classification problem in Eqn. (9.80) for each bit based on the hash values obtained in Step 1.

Step 1 can be solved using a block coordinate descent algorithm that iteratively optimizes a subset of variables at a time. For each iteration, one bit is picked for optimization in a cyclic fashion. The optimization for the k th bit can be written as

$$\min_{\mathbf{y}_{(k)}} : \sum_{i,j} \delta_{ij} l_k(y_{ik}, y_{jk}; D_{ij}^p). \quad (9.81)$$

It is shown in [54] that any Hamming distance-based loss function has the following representation:

$$\begin{aligned} l(b_1, b_2) &= \frac{1}{2} \{b_1 b_2 [l(1, 1) - l(-1, 1)] + l(1, 1) + l(-1, 1)\} \\ &= \frac{1}{2} b_1 b_2 [l(1, 1) - l(-1, 1)] + \text{const.} \end{aligned} \quad (9.82)$$

Note that $l(1, 1) = l(-1, -1)$ and $l(-1, 1) = l(1, -1)$. Therefore, the above objective function becomes

$$\min_{\mathbf{y}_{(k)}} : \sum_{i,j} \delta_{ij} [l(1, 1; D_{ij}^p) - l(-1, 1; D_{ij}^p)] y_{ik} y_{jk}. \quad (9.83)$$

By defining matrix \mathbf{A} as

$$A_{ij} = \delta_{ij} [l(1, 1; D_{ij}^p) - l(-1, 1; D_{ij}^p)], \quad (9.84)$$

the objective function takes a quadratic form:

$$\min : \mathbf{y}_{(k)}^T \mathbf{A} \mathbf{y}_{(k)}. \quad (9.85)$$

By relaxing Constraint 0 and enforcing $\|\mathbf{y}_{(k)}\|_2^2 = M$, the solution to the above optimization problem is the eigenvector of \mathbf{A} with the smallest eigenvalue. This solution is used as the initialization of an L-BFGS-B solver [57] for further refinement.

Once the first step is done, the second step is straightforward: train n binary classifiers using the training data and the optimal hash values from Step 1. The most interesting part of this framework is that any classifier can be used in the second step, making this approach versatile.

9.4.2 Hash Bit Selection

Given the loss function, the previous method constructs classifiers to generate hash bits. Efforts are needed for training the classifiers. In this section, we consider one step further. Given two hash algorithms, one can actually combine them into another algorithm. Without loss of generality, the problem is defined as follows.

Problem: Given two n -bit hash algorithms, design another n -bit hash algorithm with better performance.

Weng et al. [55] propose solving the hash construction problem by the following bit selection procedure:

- 1) Define a quality measure Q .
- 2) Compute Q for each hash bit.
- 3) Rank the $2n$ bits according to Q .
- 4) Select the best n bits.

If each hash bit is considered as a binary classifier, the above problem is equivalent to selecting the best n classifiers out of $2n$. Thus the key is to find a suitable quality metric. In [55], a quality measure called “bit reliability” is defined in terms of classification performance. Specifically, it is a weighted average of the false positive rate p_{fp} and the false negative rate p_{fn} for each hash bit. When two hash values are compared, a decision is made from two hypotheses:

- \mathbb{H}_0 – the corresponding multimedia objects are irrelevant.
- \mathbb{H}_1 – the corresponding multimedia objects are relevant.

Denote the difference between two hash values at position i as $e_i \in \{0, 1\}$. When each bit is individually used as a classifier, \mathbb{H}_1 is chosen if $e_i = 0$; otherwise \mathbb{H}_0 is chosen. The bit reliability is characterized by p_{fp} and p_{fn} of a hash bit:

- p_{fp} = Probability $\{e_i = 0 | \mathbb{H}_0\}$
- p_{fn} = Probability $\{e_i = 1 | \mathbb{H}_1\}$,

which measure discrimination and robustness, respectively. Overall, the bit reliability is defined as

$$Q = w_{fp}p_{fp} + w_{fn}p_{fn}, \quad (9.86)$$

where w_{fp} and w_{fn} are weight factors. A *smaller* Q corresponds to better reliability. After the reliability is computed, bit selection becomes a sorting procedure.

The basic bit selection framework assumes hash bits to be independent. This is a reasonable assumption when feature vectors have been decorrelated by methods such as PCA before hash bit generation. In addition to the simple selection procedure, another advantage of this assumption is that the bit reliability can be computed in parallel. However, when raw features are used, this assumption is typically not true. Therefore, sometimes the dependence among hash bits needs to be taken into account. For this reason, the framework is further extended in [56]. An undirected graph model is used to characterize the relationship between hash bits. The model has the following types of elements:

- vertex – representing a hash bit
- edge – representing the independence between two hash bits.

Both vertices and edges are given some weights. The basic idea for bit selection is to extract a dense sub-graph (maximal clique). There might be different ways of weight allocation. In [56], the vertex weight is defined in terms of the bit reliability $(1-Q)$; the edge weight is defined in terms of the mutual information between a pair of bits. Given

the probability $p(b_i)$ for the i th bit and the joint probability $p(b_i, b_j)$ for i th and j th bits, the edge weight measuring the independence between the i th and j th bits is empirically computed as:

$$I_{ij} = \exp \left\{ -\alpha \sum_{b_i, b_j} p(b_i, b_j) \log \frac{p(b_i, b_j)}{p(b_i)p(b_j)} \right\}, \quad (9.87)$$

where $\alpha > 0$ is a scaling factor. The larger value of I_{ij} , the more independence between bits b_i and b_j . Once we obtain the weighted graph, different optimization approaches can be applied to find a dense sub-graph, such as [58, 59]. In [56], the normalized dominant set approach [60] is adopted for sub-graph selection.

Another commonly used definition of the vertex weight is the embedding loss (similarity preservation) [60]. Nevertheless, the bit reliability turns out to work better, as shown in [56].

9.5 Conclusion and Discussion

In this chapter, a class of techniques called perceptual hashing is presented. The central idea is to represent multimedia objects by content-based hash values, which are typically compact binary strings. Hash-based indexing and retrieval approaches are generally faster than others because Hamming distance computation and in-memory storage can be used. A few representative perceptual hash algorithms are described. They are divided into unsupervised and supervised categories. The former focuses on data distribution; the latter also takes into account semantic relevance. In addition, two frameworks for hash algorithm construction are presented.

Perceptual hashing is essentially about shattering a high dimensional feature space. It is closely related to vector quantization and dimension reduction. On the one hand, data points are represented by their corresponding quantization cells; on the other hand, less significant cells are abandoned or merged. Therefore, perceptual hashing inherits both the advantages and the disadvantages of the aforementioned techniques. Fast speed and robustness (against incidental distortion or across semantic gaps) are obtained as the consequence of strong information loss, but discrimination is lost due to information processing. Although perceptual hashing reduces data precision, it is generally helpful to multimedia search because high-dimensional features are typically redundant and sparse. The lesson learnt from the emergence of perceptual hash algorithms is that while multimedia resources get richer and richer, proper representations should be constantly sought to facilitate efficient search and retrieval, especially for large-scale applications.

The algorithms presented in this chapter have some typical pros and cons, which are summarized in Table 9.1. Although it is not straightforward to select the best algorithm, the good characteristics of existing algorithms can help with new designs. For example, eigenproblem formulation [24, 40], kernel representation [16, 38], and two-step alternative optimization [15, 35] might be useful practice. A comprehensive survey of learning-based hash algorithms can be found in [61].

Perceptual hashing is still evolving. Some topics might be interesting for future research. First, since deep learning is gaining popularity, there is also a trend to develop

Table 9.1 Typical pros and cons of the presented perceptual hash algorithms.

Algorithm	Pros	Cons
SH [24]	Eigenproblem formulation	Assume uniform data distribution, need out-of-sample extension
ITQ [25]	No assumption on data, minimize quantization loss	Need extension to support semantics
KMH [15]	Compatible to k -means and product quantization	Potentially slow hash generation
KLSH [38]	Kernel representation	Random hyperplanes
SSH [40]	Maximize bit variance, multiple constraint optimization	Weight allocation for multiple constraints
KSH [16]	Balanced optimization, smooth surrogate, sequential training	Limited non-linearity
RBM [45]	Neural network based dimension reduction	Two-stage training, hand-crafted features
SSDH [41]	Deep/transfer learning, end-to-end training	High cost for training and running

hash algorithms based on deep neural networks. Second, there is a need for *online* hash algorithms [62, 63]. Current algorithms are typically offline – they are trained once and put into use. In reality, new data is consecutively generated, so it is necessary to support training with streaming data. Third, it would be interesting to test hashing in domains where conventional feature descriptors are still widely used, such as vision-based localization and biometric recognition. These are just a few examples.

References

- 1 Torralba, A., Fergus, R., and Weiss, Y. (2008) Small codes and large image databases for recognition, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–8.
- 2 Sun, X., Wang, C., Xu, C., and Zhang, L. (2013) Indexing billions of images for sketch-based retrieval, in *Proceedings of the ACM International Conference on Multimedia*, pp. 233–242.
- 3 Gong, Y., Pawlowski, M., Yang, F., Brandy, L., Boundev, L., and Fergus, R. (2015) Web scale photo hash clustering on a single machine, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 19–27.
- 4 Cano, P., Batle, E., Kalker, T., and Haitsma, J. (2002) A review of algorithms for audio fingerprinting, in *Proceedings of the IEEE Workshop on Multimedia Signal Processing*, pp. 169–173.
- 5 Chandrasekhar, V., Sharifi, M., and Ross, D. (2011) Survey and evaluation of audio fingerprinting schemes for mobile query-by-example applications, in *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pp. 801–806.

- 6 Cao, L., Li, Z., Mu, Y., and Chang, S.F. (2012) Submodular video hashing: A unified framework towards video pooling and indexing, in *Proceedings of the ACM International Conference on Multimedia*, pp. 299–308.
- 7 Ye, G., Liu, D., Wang, J., and Chang, S.F. (2013) Large-scale video hashing via structure learning, in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 2272–2279.
- 8 Sivic, J. and Zisserman, A. (2003) Video Google: a text retrieval approach to object matching in videos, in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 1470–1477.
- 9 Sánchez, J., Perronnin, F., Mensink, T., and Verbeek, J. (2013) Image classification with the fisher vector: Theory and practice. *International Journal of Computer Vision*, 105 (3), 222–245.
- 10 Oliva, A. and Torralba, A. (2001) Modeling the shape of the scene: A holistic representation of the spatial envelope. *International Journal of Computer Vision*, 42 (3), 145–175.
- 11 Arandjelović, R., Gronat, P., Torii, A., Pajdla, T., and Sivic, J. (2016) NetVLAD: CNN architecture for weakly supervised place recognition, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5297–5307.
- 12 Bishop, C.M. (2006) *Pattern Recognition and Machine Learning*, Springer.
- 13 Lloyd, S. (1982) Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28 (2), 129–137.
- 14 Jegou, H., Douze, M., and Schmid, C. (2011) Product quantization for nearest neighbor search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33 (1), 117–128.
- 15 He, K., Wen, F., and Sun, J. (2013) K-means hashing: An affinity-preserving quantization method for learning binary compact codes, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2938–2945.
- 16 Liu, W., Wang, J., Ji, R., Jiang, Y.G., and Chang, S.F. (2012) Supervised hashing with kernels, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2074–2081.
- 17 Weng, L., Amsaleg, L., Morton, A., and Marchand-Maillet, S. (2015) A privacy-preserving framework for large-scale content-based information retrieval. *IEEE Transactions on Information Forensics and Security*, 10 (1), 152–167.
- 18 Weng, L., Amsaleg, L., and Furon, T. (2016) Privacy-preserving outsourced media search. *IEEE Transactions on Knowledge and Data Engineering*, 28 (10), 2738–2751.
- 19 Doets, P.J.O. and Lagendijk, R.L. (2008) Distortion estimation in compressed music using only audio fingerprints. *IEEE Transactions on Audio, Speech, and Language Processing*, 16 (2), 302–317.
- 20 Weng, L. and Preneel, B. (2011) Image distortion estimation by hash comparison, in *Proceedings of the International Multimedia Modeling Conference (MMM)*, pp. 62–72.
- 21 Weng, L., Braeckman, G., Dooms, A., Preneel, B., and Schelkens, P. (2012) Robust image content authentication with tamper location, in *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME)*, pp. 380–385.
- 22 Powers, D.M.W. (2011) Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation. *International Journal of Machine Learning Technology*, 2 (1), 37–63.

- 23 Alpaydin, E. (2014) *Introduction to Machine Learning*, 3rd edn, MIT Press.
- 24 Weiss, Y., Torralba, A., and Fergus, R. (2009) Spectral hashing, in *Advances in Neural Information Processing Systems in Proceedings of the 21st International Conference on Neural Information Processing Systems (NIPS'08)*, Vancouver, British Columbia, Canada, pp. 1753–1760, Curran Associates Inc.
- 25 Gong, Y. and Lazebnik, S. (2011) Iterative quantization: A procrustean approach to learning binary codes, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 817–824.
- 26 Kulis, B. and Grauman, K. (2009) Kernelized locality-sensitive hashing for scalable image search, in *Proceedings of the IEEE International Conference on Computer Vision (CVPR)*, pp. 2130–2137.
- 27 Fowlkes, C., Belongie, S., Chung, F., and Malik, J. (2004) Spectral grouping using the nystrom method. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26 (2), 214–225.
- 28 Bengio, Y., Paiement, J., Vincent, P., Delalleau, O., Roux, N.L., and Ouimet, M. (2003) Out-of-sample extensions for LLE, isomap, MDS, eigenmaps, and spectral clustering, in *Advances in Neural Information Processing Systems*, pp. 177–184, MIT Press.
- 29 Bengio, Y., Delalleau, O., Le Roux, N., Paiement, J.F., Vincent, P., and Ouimet, M. (2004) Learning eigenfunctions links spectral embedding and kernel PCA. *Neural Computation*, 16 (10), 2197–2219.
- 30 Coifman, R.R., Lafon, S., Lee, A.B., Maggioni, M., Nadler, B., Warner, F., and Zucker, S.W. (2005) Geometric diffusions as a tool for harmonic analysis and structure definition of data: Diffusion maps. *Proceedings of the National Academy of Sciences of the United States of America*, 102 (21), 7426–7431.
- 31 Nadler, B., Lafon, S., Coifman, R.R., and Kevrekidis, I.G. (2006) Diffusion maps, spectral clustering and reaction coordinates of dynamical systems. *Applied and Computational Harmonic Analysis*, 21 (1), 113–127.
- 32 Belkin, M. and Niyogi, P. (2008) Towards a theoretical foundation for Laplacian-base manifold methods. *Journal of Computer and System Sciences*, 74 (8), 1289–1308.
- 33 Weiss, Y., Fergus, R., and Torralba, A. (2012) Multidimensional spectral hashing, in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 340–353.
- 34 Schönemann, P.H. (1966) A generalized solution of the orthogonal procrustes problem. *Psychometrika*, 31 (1), 1–10.
- 35 Gong, Y., Lazebnik, S., Gordo, A., and Perronnin, F. (2013) Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35 (12), 2916–2929.
- 36 Gray, R. (1984) Vector quantization. *IEEE ASSP Magazine*, 1 (2), 4–29.
- 37 Shanno, D.F. (1970) Conditioning of quasi-Newton methods for function minimization. *Mathematics of Computation*, 24 (111), 647–656.
- 38 Kulis, B. and Grauman, K. (2012) Kernelized locality-sensitive hashing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34 (6), 1092–1104.
- 39 Slaney, M. and Casey, M. (2008) Locality-sensitive hashing for finding nearest neighbors [lecture notes]. *IEEE Signal Processing Magazine*, 25 (2), 128–131.
- 40 Wang, J., Kumar, S., and Chang, S.F. (2010) Semi-supervised hashing for scalable image retrieval, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3424–3431.

- 41 Yang, H.F., Lin, K., and Chen, C.S. (2017) Supervised learning of semantics-preserving hash via deep convolutional neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40 (2), 437–451.
- 42 Wang, J., Kumar, S., and Chang, S.F. (2012) Semi-supervised hashing for large-scale search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34 (12), 2393–2406.
- 43 Nesterov, Y. (2013) *Introductory lectures on convex optimization: A basic course*, vol. 87, Springer Science & Business Media.
- 44 LeCun, Y., Bengio, Y., and Hinton, G. (2015) Deep learning. *Nature*, 521 (7553), 436–444.
- 45 Salakhutdinov, R. and Hinton, G. (2009) Semantic hashing. *International Journal of Approximate Reasoning*, 50 (7), 969–978.
- 46 Hinton, G.E. and Salakhutdinov, R.R. (2006) Reducing the dimensionality of data with neural networks. *Science*, 313 (5786), 504–507.
- 47 Hinton, G.E. (2002) Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14 (8), 1771–1800.
- 48 Goldberger, J., Roweis, S., Hinton, G., and Salakhutdinov, R. (2004) Neighbourhood components analysis, in *Advances in Neural Information Processing Systems*, pp. 513–520, MIT Press.
- 49 Salakhutdinov, R. and Hinton, G.E. (2007) Learning a nonlinear embedding by preserving class neighbourhood structure, in *Proceedings of the International Conference on Artificial Intelligence and Statistics*, pp. 412–419.
- 50 Simonyan, K. and Zisserman, A. (2015) Very deep convolutional networks for large-scale image recognition, in *Proceedings of the International Conference on Learning Representations (ICLR)*.
- 51 He, K., Zhang, X., Ren, S., and Sun, J. (2016) Deep residual learning for image recognition, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778.
- 52 Wolpert, D.H. (1996) The lack of a priori distinctions between learning algorithms. *Neural Computation*, 8 (7), 1341–1390.
- 53 Wolpert, D.H. and Macready, W.G. (1997) No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1 (1), 67–82.
- 54 Lin, G., Shen, C., Suter, D., and Van Den Hengel, A. (2013) A general two-step approach to learning-based hashing, in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 2552–2559.
- 55 Weng, L., Jhuo, I.H., Shi, M., Sun, M., Cheng, W.H., and Amsaleg, L. (2015) Supervised multi-scale locality sensitive hashing, in *Proceedings of the International Conference on Multimedia Retrieval (ICMR)*, pp. 259–266.
- 56 Jhuo, I.H., Weng, L., Cheng, W.H., and Lee, D.T. (2016) A feature fusion framework for hashing, in *Proceedings of the International Conference on Pattern Recognition (ICPR)*, 2288–2293.
- 57 Zhu, C., Byrd, R.H., Lu, P., and Nocedal, J. (1997) Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization. *ACM Transactions on Mathematical Software*, 23 (4), 550–560.
- 58 Pavan, M. and Pelillo, M. (2007) Dominant sets and pairwise clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29 (1), 167–172.

- 59 Liu, S., Liu, H., Latecki, L.J., Yan, S., Xu, C., and Lu, H. (2011) Size adaptive selection of most informative features, in *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 392–297.
- 60 Liu, X., He, J., Lang, B., and Chang, S.F. (2013) Hash bit selection: a unified solution for selection problems in hashing, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1570–1577.
- 61 Wang, J., Liu, W., Kumar, S., and Chang, S.F. (2016) Learning to hash for indexing big data – A survey. *Proceedings of the IEEE*, 104 (1), 34–57.
- 62 Leng, C., Wu, J., Cheng, J., Bai, X., and Lu, H. (2015) Online sketching hashing, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2503–2511.
- 63 Huang, L.K., Yang, Q., and Zheng, W.S. (2017) Online hashing. *IEEE Transactions on Neural Networks and Learning Systems*, 29 (6), 2309–2322.

Part IV

Applications of Large-Scale Multimedia Search

10

Image Tagging with Deep Learning: Fine-Grained Visual Analysis

Jianlong Fu and Tao Mei

10.1 Introduction

Recent years have witnessed the ubiquity of mobile devices (e.g., smart phones, digital cameras, tablets, etc.) and media cloud services. This has led to an unprecedented growth in the number of personal photos. People are taking photos using their smart devices every day and everywhere. It is reported that Flickr¹ has 1.7 million photos uploaded every day and Instagram² claimed 40 million photos per day in 2015. Such a great number of images demands effective image accessing techniques. As evidenced by the success of commercial search engines (e.g., Google³ and Bing⁴), one of the most effective ways for common users to access diverse image content is through text. Therefore, image tagging⁵ has become an active research topic in the past decade that aims to annotate an image with human-friendly semantic tags.

Most previous image tagging approaches depended on hand-crafted feature-based image representations, e.g., scale-invariant feature transform (SIFT) [1], histogram of oriented gradients (HOG) [2], GIST [3], and so on. Once these low-level feature descriptors are extracted, visual representation algorithms, e.g., bag-of-word features [4] or spatial pyramid features [5] have been proposed to describe image content and associate the content with natural language based keywords. However, hand-crafted feature descriptors are designed to describe low-level visual patterns by pre-defined feature types such as color, shape, and texture. Although promising results have been achieved by combining multi-type feature representations [6–9], these features are still inadequate to detect and describe high-level semantic tags.

With the recent success in many research areas, deep learning techniques have attracted great attention [10]. For example, convolutional neural networks (CNNs) have achieved a winning top-5 test error rate of 15.3%, compared to the 26.2% achieved by the second-best approach which combines scores from many classifiers trained by a

1 <https://www.flickr.com>.

2 <http://instagram.com>.

3 <http://www.google.com>.

4 <http://www.bing.com>.

5 “Tagging” and “recognition” are considered as interchangeable terms, and we do not differentiate them in this chapter.

set of hand-crafted features [11] in 2012. Recently, the top-5 image classification error has been further reduced to 3.57% on the ImageNet⁶ test set by using the promising deep residual nets [12], which has been achieved superior performance to humans. Promising results have been reported for image tagging by leveraging off-the-shelf deep learning techniques [13, 14].

However, existing image tagging approaches by deep learning are hardly able to recognize fine-grained categories with thousands of sub-categories, for instance bird species [15, 16], flower types [17, 18], car models [19, 20], or even the subtle human sentiment conveyed from images, which are usually more useful than general image recognition (e.g., of bird, flower, car, face, person, etc.) for real users. For example, a recognizer for different flower species can benefit children, hikers, plant enthusiasts or even plant experts because some fine-grained flower categories are difficult to recognize, even for domain experts. The challenge is mainly derived from the subtle visual differences that exist to in local regions from similar fine-grained categories, which are usually difficult to learn by general deep learning techniques. Besides, understanding the emotion and sentiment (e.g., “positive,” “negative”) from visual content by fine-grained recognition techniques has attracted great attention, since the sentiment conveyed from visual content can explain or even strengthen the sentiment conveyed from text. The capability of automatic visual sentiment analysis will promote visual understanding and benefit a broad range of applications, such as affective computing [21–23], opinion mining [24–26], and so on.

In this chapter we will introduce the techniques and applications of deep learning frameworks on fine-grained image tagging, i.e., recognizing image sub-categories or visual sentiment. Compared with previous publications [16] and [26], we provide more comparison between different deep learning models and deeper analysis with visualization results over widely used fine-grained image tagging datasets. The rest of this chapter is organized as follows. Section 10.2 provides a brief introduction to deep learning techniques for image tagging. Sections 10.3 and 10.4 introduce the deep learning architecture for fine-grained image recognition and visual sentiment analysis, respectively. Finally, we conclude this chapter in section 10.5.

10.2 Basic Deep Learning Models

As CNNs have shown superior performance on the learning of discriminative image feature representation, CNN and its variances have been widely used in image recognition and tagging tasks. A typical convolutional neural network often consists of several convolutional and fully connected layers. The exact number of layers generally depends on the requirement of network capacity and memory cost for a specific classification task. For example, we use eight layers for AlexNet [11] and 19 layers for VGG-19 net [27].

⁶ ImageNet is an image database organized according to the noun hierarchy from WordNet, in which each node of the hierarchy is depicted by hundreds or even thousands of images. Details can be found in <http://www.image-net.org>.

In particular, let $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_N]$ be the matrix of image training data, where \mathbf{x}_i is the feature vector of the i th image. N is the total number of images. Denote $\mathbf{Y} = [\mathbf{y}_1, \dots, \mathbf{y}_N]^T \in \{0, 1\}^{N \times K}$, where $\mathbf{y}_i \in \{0, 1\}^{K \times 1}$ is the category indicator vector for \mathbf{x}_i . K is the number of categories. Suppose there are M layers in total and $\mathbf{W} = \{\mathbf{W}^{(1)}, \dots, \mathbf{W}^{(M)}\}$ are the model parameters. In each layer, we absorb the bias term into the weights and denote them as a whole. $\mathbf{W}^{(m)} = [\mathbf{w}_1^{(m)}, \dots, \mathbf{w}_{d_m}^{(m)}]^T \in R^{d_m \times d_{m-1}}$, where $\mathbf{w}_i^{(m)} \in R^{d_{m-1}}$, d_{m-1} is the dimension of the $(m-1)$ th feature map. $\mathbf{Z}^{(m)}(\mathbf{X}) = [\mathbf{z}^{(m)}(\mathbf{x}_1), \dots, \mathbf{z}^{(m)}(\mathbf{x}_N)]^T \in R^{N \times d_m}$ denotes the feature map produced by the m th layer.

Given an image \mathbf{x}_i , convolutional layers first task the image as input. The extracted deep representations are denoted as $\mathbf{W} * \mathbf{x}_i$, where $*$ denotes a set of operations of convolution, pooling, and activation, and \mathbf{W} denotes the overall parameters for convolutional and pooling operations. The pooling layer in a CNN is designed to summarize the outputs of neighboring groups of neurons in the same kernel map, which can be conducted by mean-pooling or max-pooling. Some earlier works adopt the neighborhood summarization strategy by adjacently pooling without overlaps. To make it clearer, a pooling layer can be considered as consisting of a grid of pooling units spaced s pixels apart. Each pooling unit summarizes a neighborhood of size $r \times r$ centered at the location in a CNN. If $s = r$, we can obtain the non-overlapping pooling. If $s < r$, we can obtain overlapping pooling. Extensive studies have shown that the overlapping pooling is difficult to overfit [11].

For the activation function, the standard way to activate a neuron's output is through the tan or sigmoid functions, which are considered as saturating nonlinear functions. However, for the training optimization with gradient descent, these saturating nonlinear functions show much slower convergence than the non-saturating nonlinear function, such as $\max(0, x)$. The nonlinear function is referred to as rectified linear units (ReLUs). It has been demonstrated that deep CNNs with ReLUs can be trained several times faster than their equivalents with tan units, which is crucial for the training of large models on large datasets.

Given the output from convolutional layers, we further feed it into a series of fully connected layers. The output of the last fully connected layer is considered as an input to a softmax classifier which can generate a distribution over the final category labels, given by:

$$\mathbf{p}(\mathbf{x}_i) = f(\mathbf{W} * \mathbf{x}_i), \quad (10.1)$$

where $f(\cdot)$ represents fully connected layers to map convolutional features to a feature vector that could be matched with the categories entries, as well as including a softmax layer to further transform the feature vector to probabilities. The goal is to minimize the following objective function in the form of a softmax regression with weight decay, which is given by:

$$\mathcal{L}(\mathbf{W}) = -\frac{1}{N} \left[\sum_{i=1}^N \sum_{j=1}^K \mathbf{1}_{Y_{ij}}(j) \log p(Y_{ij} = 1 | \mathbf{x}_i; \mathbf{W}) \right] + \frac{\beta}{2} \|\mathbf{W}\|_F^2, \quad (10.2)$$

where Y_{ij} is the (i, j) th entry of \mathbf{Y} . $\mathbf{1}_{Y_{ij}}(j)$ is the indicator function such that $\mathbf{1}_{Y_{ij}}(j) = 1$ if $Y_{ij} = 1$, otherwise zero. β is the weight decay coefficient, which is designed to reduce

Table 10.1 The comparison of different CNN architectures on model size, error rate, and model depth.

Model	Size (M)	Top-1/5 Err (%)	Depth	Description
AlexNet [11]	238	41.00 / 18.00	8	5 Conv, 3 fc
VGG-16 [27]	540	28.07 / 9.33	16	13 Conv, 3 fc
VGG-19 [27]	560	27.30 / 9.00	19	16 Conv, 3 fc
GoogLeNet [28]	40	29.81 / 10.04	22	21 Conv, 1 fc
ResNet-50 [12]	100	22.85 / 6.71	50	49 Conv, 1 fc
ResNet-152 [12]	235	21.43 / 3.57	152	151 Conv, 1 fc

model complexity and thus to prohibit overfitting. The probability p can be generated by a softmax function, which is given by:

$$p(Y_{ij} = 1 | \mathbf{x}_i; \mathbf{W}) = \frac{\exp(\mathbf{z}_j^{(M-1)})}{\sum_{k=1}^K \exp(\mathbf{z}_k^{(M-1)})}. \quad (10.3)$$

Parameters across different layers can be optimized by back-propagation (BP) [29]. Different CNN architectures have different numbers of convolutional layers and fully connected layers. Detailed performance comparisons for different CNN model architectures on ImageNet challenges are shown in Table 10.1, where “Conv” and “FC” indicate convolutional layers and fully connected layers, respectively. Although promising results have been achieved, further improvement on fine-grained and visual sentiment categories are still difficult, since these CNN architectures are not designed for capturing localized feature representations.

10.3 Deep Image Tagging for Fine-Grained Image Recognition

Different from general image tagging, fine-grained image tagging should be capable of localizing and representing the very marginal visual differences within subordinate categories. In this section, we will introduce the recurrent attention convolutional neural network (RA-CNN) for fine-grained image tagging [16]. The proposed RA-CNN is a stacked network which takes the input from full images to fine-grained local regions at multiple scales.

We consider the network with three scales as an example in Figure 10.1, and more finer scales can be stacked in a similar way. The inputs are recurrent from full-size images in a_1 to fine-grained discriminative regions in a_2 and a_3 , where a_2 and a_3 take the input as the attended regions from a_1 and a_2 , respectively. First, images at different scales are fed into convolutional layers (b_1 to b_3) to extract region-based feature representation. Second, networks proceed to predict both a probability score by fully connected and softmax layers (c_1 to c_3) and a region attention by an attention proposal network (d_1, d_2). The proposed RA-CNN is optimized to convergence by alternatively learning a softmax classification loss at each scale and a pairwise ranking loss across neighboring scales.

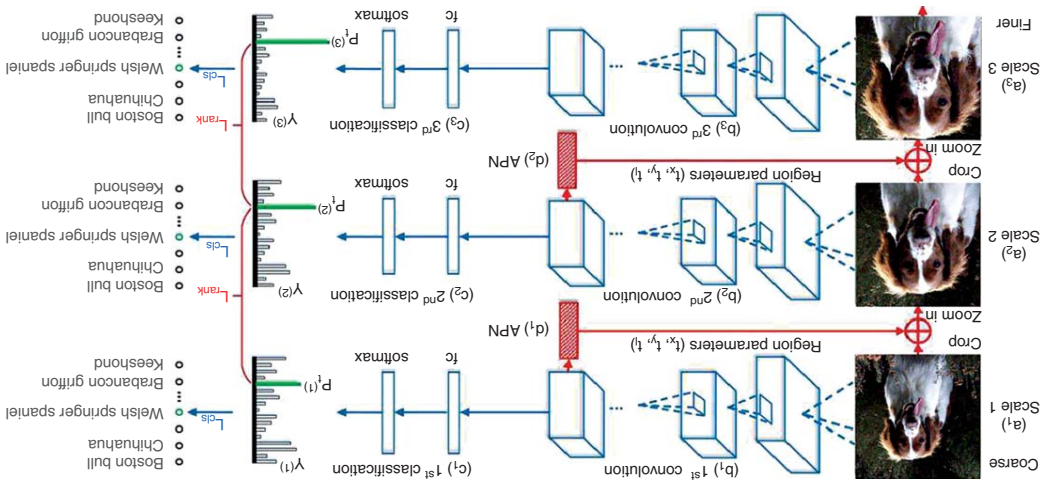


Figure 10.1 The framework of an RA-CNN. The inputs are from coarse full-size images to finer region attention (from top to bottom). Different network modules for classification (marked in blue) and attention proposal (marked in red) are alternatively optimized by classification losses L_{cls} between label prediction $Y^{(s)}$ and ground truth Y^* at each scale, and pairwise ranking losses L_{rank} between $p^{(s)}$ and $p^{(s+1)}$ from neighboring scales, where $p^{(s)}$ and $p^{(s+1)}$ denote the probabilities on the correct category, and s denotes the scale. APN is the attention proposal network, fc represents a fully connected layer, the softmax layer matches to category entries by an fc layer, followed by a softmax operation. \oplus represents a "crop" and "zoom in" operation.

10.3.1 Attention Proposal Network

Multi-task formulation. Inspired by the recent success of the region proposal network (RPN) [30], we propose an attention proposal network (APN) where the computation of region attention is nearly cost-free, and the APN can be trained end-to-end.

Given an input image \mathbf{X} , we first extract region-based deep features by feeding the images into pre-trained convolutional layers. The extracted deep representations are denoted as $\mathbf{W}_c * \mathbf{X}$, where $*$ denotes a set of operations of convolution, pooling, and activation, and \mathbf{W}_c denotes the overall parameters. We further model the network at each scale as a multi-task formulation with two outputs. The first task is designed to generate a probability distribution \mathbf{p} over fine-grained categories. The specific form of \mathbf{p} is given by:

$$\mathbf{p}(\mathbf{X}) = f(\mathbf{W}_c * \mathbf{X}), \quad (10.4)$$

where $f(\cdot)$ represents fully connected layers to map convolutional features to a feature vector that could be matched with the category entries, as well as including a softmax layer to further transform the feature vector to probabilities. The second task is proposed to predict a set of box coordinates of an attended region for the next finer scale. By approximating the attended region as a square with three parameters, the representation is given by:

$$[t_x, t_y, t_l] = g(\mathbf{W}_c * \mathbf{X}), \quad (10.5)$$

where t_x and t_y denote the square's center coordinates in terms of x - and y -axis, respectively, and t_l denotes the half of the square's side length. The specific form of $g(\cdot)$ can be represented by two-stacked fully connected layers with three outputs which are the parameters of the attended regions. Note that compared with the RPN in object detection, which uses strong supervision of ground truth boxes, the learning of the proposed APN is trained in a weakly supervised fashion, since the part-level annotation is often hard to obtain. The specific learning process and loss functions will be introduced in section 10.3.2.

Attention localization and amplification. Once the location of an attended region is hypothesized, we crop and zoom in the attended region to a finer scale with higher resolution to extract more fine-grained features. To ensure the APN can be optimized in training, we approximate the cropping operation by proposing a variant of a two-dimensional boxcar function as an attention mask. The mask can select the most significant regions in forward-propagation, and is readily to be optimized in backward-propagation due to the properties of continuous functions.

Assume the top-left corner in original images as the origin of a pixel coordinate system, whose x -axis and y -axis are defined from left to right and top to bottom, respectively. We can adopt the parameterizations of the top left (denoted as "tl") and bottom right (denoted as "br") points from the attended region as following:

$$\begin{aligned} t_{x(tl)} &= t_x - t_l, & t_{y(tl)} &= t_y - t_l, \\ t_{x(br)} &= t_x + t_l, & t_{y(br)} &= t_y + t_l. \end{aligned} \quad (10.6)$$

Based on the above representations, the cropping operation can be implemented by an element-wise multiplication between the original image at coarser scales and an attention mask, which can be computed as:

$$\mathbf{X}^{att} = \mathbf{X} \odot \mathbf{M}(t_x, t_y, t_l), \quad (10.7)$$

where \odot represents element-wise multiplication, \mathbf{X}^{att} denotes the cropped attended region, and $\mathbf{M}(\cdot)$ acts as an attention mask, with the specific form:

$$\begin{aligned} \mathbf{M}(\cdot) = & [h(x - t_{x(tl)}) - h(x - t_{x(br)})] \\ & \cdot [h(y - t_{y(tl)}) - h(y - t_{y(br)})], \end{aligned} \quad (10.8)$$

and $h(\cdot)$ is a logistic function with index k :

$$h(x) = 1 / \{1 + \exp^{-kx}\}. \quad (10.9)$$

Theoretically, when k is large enough, the logistic function can be considered as a step function and the two-dimensional boxcar function (i.e., $\mathbf{M}(\cdot)$) is zero over the entire real line along the x and y dimensions, except for a single area (i.e., x ranges from $t_{x(tl)}$ to $t_{x(br)}$, and y ranges from $t_{y(tl)}$ to $t_{y(br)}$) where it is equal to one. The advantages for using the boxcar function are two fold. First, the boxcar function can well approximate the cropping operation to select the most significant regions predicted from coarser-scale networks. Second, the boxcar function builds analytical representations between the attended region and box coordinates $\{t_x, t_y, t_l\}$, which is necessary when optimizing box parameters in backward-propagation.

Although attended regions have been localized, effective feature representation are sometimes still difficult to be extracted from the highly localized regions. Therefore, we further amplify the region to a larger size by adaptively zooming. Specifically, we use a bilinear interpolation to compute the amplified output \mathbf{X}^{amp} from the nearest four inputs in \mathbf{X}^{att} by a linear map, which is given by:

$$\mathbf{X}_{(i,j)}^{amp} = \sum_{\alpha, \beta=0}^1 |1 - \alpha - \{i/\lambda\}| |1 - \beta - \{j/\lambda\}| \mathbf{X}_{(m,n)}^{att}, \quad (10.10)$$

where $m = [i/\lambda] + \alpha$, $n = [j/\lambda] + \beta$, and λ is upsampling factor, which equals the value of enlarged size divided by t_l . $[\cdot]$ and $\{\cdot\}$ are the integral and fractional parts, respectively.

10.3.2 Classification and Ranking

Note that compared with the existing attention mechanism used in image captioning [31] and visual question answering (VQA) [32], [33], we do not have explicit supervision in our scenario (e.g., semantic concepts in captioning and questions in VQA) for attention learning. To solve this problem, the proposed recurrent attention CNN is optimized by two types of supervision, i.e., intra-scale classification loss and inter-scale pairwise ranking loss, for alternatively generating accurate region attention and learning more fine-grained features. Specifically, we minimize an objective function following a

multi-task loss. The loss function for an image sample is defined as:

$$L(\mathbf{X}) = \sum_{s=1}^3 \{L_{cls}(\mathbf{Y}^{(s)}, \mathbf{Y}^*)\} + \sum_{s=1}^2 \{L_{rank}(p_t^{(s)}, p_t^{(s+1)})\}, \quad (10.11)$$

where s denotes each scale, and $\mathbf{Y}^{(s)}$ and \mathbf{Y}^* denote the predicted label vectors from a specific scale and the ground truth label vector, respectively. L_{cls} represents classification loss, which predominantly optimizes the parameters of convolution and classification layers in Figure 10.1 (b_1 to b_3 and c_1 to c_3) for ensuring adequate discrimination ability at each scale. The training is implemented by fitting category labels on overall training samples via a softmax function. In addition, $p_t^{(s)}$ from pairwise ranking loss L_{rank} denotes the prediction probability on the correct category labels t . Specifically, the ranking loss is given by:

$$L_{rank}(p_t^{(s)}, p_t^{(s+1)}) = \max\{0, p_t^{(s)} - p_t^{(s+1)} + \text{margin}\}, \quad (10.12)$$

which enforces $p_t^{(s+1)} > p_t^{(s)} + \text{margin}$ in training. Such a design can enable networks to take the prediction from coarse scales as references, and gradually approach the most discriminative region by enforcing the finer-scale network to generate more confident predictions. Note that L_{cls} and L_{rank} take effect alternatively for different optimization purposes.

10.3.3 Multi-Scale Joint Representation

Once the proposed RA-CNN has been trained at each scale, we can obtain multi-scale representations from full-size images to multiple coarse-to-fine region attention. In particular, the image \mathbf{X} can be represented by a set of multiple-scale descriptors:

$$\{F_1, F_2, \dots, F_N\}, \quad (10.13)$$

where F_i denotes the feature descriptor at a specific scale generated from the fully connected layers in the classification net (c_1 to c_3 in Figure 10.1) and N is the total number of scales. To leverage the benefit of feature ensemble, we first normalize each descriptor independently and concatenate them together into a fully connected fusion layer with softmax function for the final classification. The application of softmax function instead of support vector machine (SVM) [34] is mainly for the technical consistency for feature extraction and classification, so that we can integrate the multi-scale descriptors and classification end-to-end in testing. In addition, we have verified that softmax and linear SVM can produce comparable results for classification.

10.3.4 Implementation Details

To better optimize attention localization and fine-grained classification in a mutually reinforced way, we take the following alternative training strategy.

- Step 1:* Initialize the convolutional/classification layers in Figure 10.1 (b_1 to b_3 and c_1 to c_3) by the same pretrained VGG network [27] from ImageNet.
- Step 2:* Consider a square (represented by t_x, t_y, t_l) with the half length of the side of the original image. The square is selected by searching regions in the original image, with the highest response value in the last convolutional layer (i.e., conv5_4 in

VGG-19). We can further obtain a smaller square by analyzing convolutional responses at the second scale in a similar way. These selected squares are used to pretrain APN to obtain the parameters (d_1), and (d_2) in Figure 10.1 by learning the transformation from convolutional feature maps to $\{t_x, t_y, t_l\}$.

Step 3: we Optimize the parameters in the above two steps in an alternative way. Specifically, we keep the APN parameters unchanged, and optimize the softmax losses at three scales to converge. Then we fix the parameters in convolutional/classification layers, and switch to ranking loss to optimize the two APNs. The learning process for the two parts is iterative until the two types of losses no longer change. In addition, t_l at each scale is constrained to be no less than one-third of the previous t_l at coarse scale, to avoid the incompleteness of object structures when t_l is too small. The model has been made publicly available at <https://github.com/Jianlong-Fu/Recurrent-Attention-CNN>.

10.3.5 Experiments on CUB-200-2011

The CUB-200-2011 dataset is widely used for fine-grained image recognition. It contains 200 bird species, with training/testing split of 5994 and 5794. We conducted an experiment on this dataset, and show the attended regions from multiple scales by the proposed PN for qualitative analysis. In Figure 10.2, we can observe that these localized regions at the second and third scales are discriminative to corresponding categories, and are easier to be classified than the first scale. The results are consistent with human perception that it would be helpful to look closer to see better on fine-grained categories.

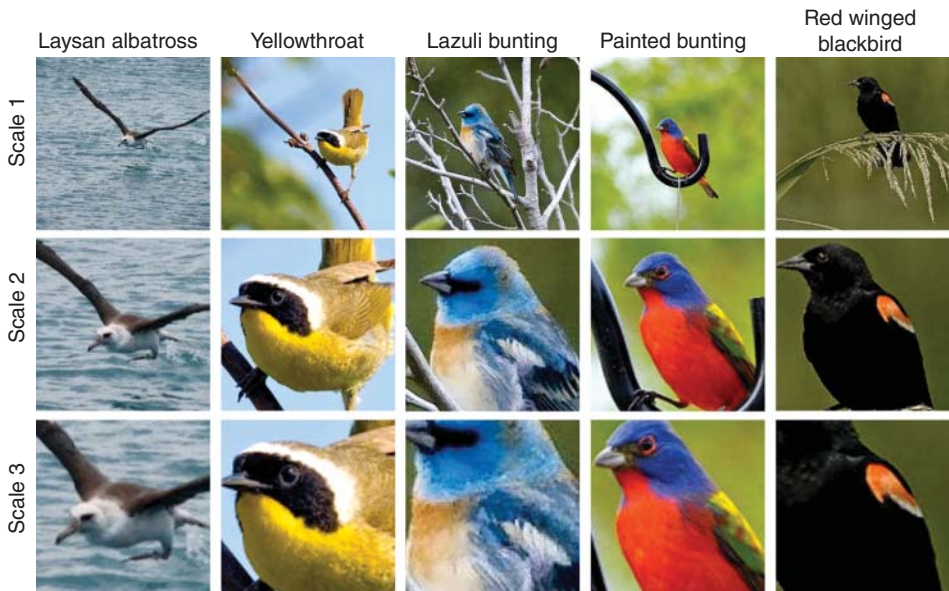


Figure 10.2 Five bird examples of the learned region attention at different scales. We can observe clear and significant visual cues for classification after gradually zooming into the attended regions.

Since the proposed APN is automatically learned by discovering the most discriminative regions or classification, instead of regressing a human-defined bounding box, we conducted a quantitative comparison on attention localization in terms of classification accuracy. For fair comparison, all compared methods use the VGG-19 model, but with different attention localization algorithms. We take the second-scale network to produce our results (denoted as RA-CNN (scale 2)), as attended regions at this scale can preserve both global bird structure and local visual cues, as shown in Figure 10.2. First, we can observe comparable results with the methods using the human-defined bounding box in Table 10.3. PA-CNN [19] and MG-CNN (with annotation) [36] achieve 82.8% and 83.0% accuracy, respectively. RA-CNN (scale 2) achieves 82.4% accuracy. Second, we can achieve significantly better results compared with existing unsupervised part learning-based methods. FCAN [35] and MG-CNN [36] are two relevant works to ours that also use a feature combination scheme from multiple scales/granularities. To make a fair comparison, we select single-attention and single-granularity based performance from [35] and [36], and show the results in Table 10.2. We can obtain 8.3% and 3.6% relative improvement compared with FCAN (single-attention) [35] and MG-CNN (single-granularity) [36], which shows the superior attention learning ability of the proposed approach. In addition, the result of RA-CNN with an initialized attended region and without ranking loss optimization is listed in the third row. From this result, we can know the key role of ranking loss for optimizing region attention.

Figure 10.3 shows the learning process for region attention, which is obtained by cropping the attended regions using the optimized t_x, t_y, t_l in each iteration. The initial regions (in iteration 1) are obtained by the highest response areas in convolutional layers from the previous coarse scale. We can observe that initial regions fail to capture the important head areas for the two birds because it is difficult for the coarse-scale network to find those discriminative regions with small size and low resolution. After 200 iterations, the proposed RA-CNN can gradually pinpoint more discriminative regions for the two birds by the proposed ranking losses.

We compare two types of baselines based on whether or not they use human-defined bounding box (bbox)/part annotations. PN-CNN [15] uses strong supervision of both a human-defined bounding box and ground truth parts. B-CNN [20] uses a bounding box with very high-dimensional feature representation (250k dimensions). As shown in Table 10.3, the proposed RA-CNN (scale 1+2+3) can achieve comparable results with PN-CNN [15] and B-CNN [20] even without bbox and part annotation, which demonstrates its effectiveness. Compared with unsupervised methods picking deep filter response (PDFR) [37] without additional Fisher vector learning, we can obtain a relative accuracy gain of 3.3% by our full model RA-CNN (scale 1+2+3). We even

Table 10.2 Comparison of attention localization in terms of classification accuracy on the CUB-200-2011 dataset.

Approach	Accuracy
FCAN (single-attention) [35]	76.1
MG-CNN (single-granularity) [36]	79.5
RA-CNN (scale 2) w/initial $\{t_x, t_y, t_l\}$ [16]	79.0
RA-CNN (scale 2) [16]	82.4

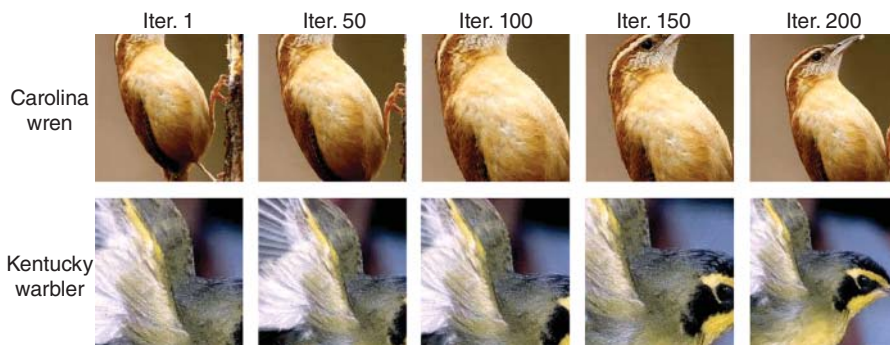


Figure 10.3 The learning process in each iteration of region attention at the second scale on the CUB-200-2011 bird dataset. We can observe that initial regions (in iteration 1) fail to capture the important head areas for the two birds. The proposed RA-CNN can gradually pinpoint the most discriminative regions for the two birds by the proposed ranking losses after 200 iterations.

Table 10.3 Comparison results on the CUB-200-2011 dataset.

Approach	Train anno.	Accuracy
DeepLAC [37]	✓	80.3
Part-RCNN [40]	✓	81.6
PA-CNN [19]	✓	82.8
MG-CNN [36]	✓	83.0
FCAN [35]	✓	84.3
B-CNN (250k-dims) [20]	✓	85.1
SPDA-CNN [41]	✓	85.1
PN-CNN [15]	✓	85.4
VGG-19 [27]		77.8
TLAN [42]		77.9
DVAN [39]		79.0
NAC [43]		81.0
MG-CNN [36]		81.7
FCAN [35]		82.0
PDFR [37]		82.6
B-CNN (250k-dims) [20]		84.1
ST-CNN (Inception) [38]		84.1
RA-CNN (scale 2) [16]		82.4
RA-CNN (scale 3) [16]		81.2
RA-CNN (scale 1+2) [16]		84.7
RA-CNN (scale 1+2+3) [16]		85.3

Train annotation is the bounding box/part annotation.

surpass B-CNN (w/o anno.) [20] and ST-CNN [38], which use either high-dimensional features or a stronger inception network as a baseline model with nearly 1.5% relative accuracy gains. Although FCAN (w/o anno.) [35] and DVAN [39] propose similar ideas to zoom into attended regions for classification, we can achieve better accuracy with 4.1% and 8.0% relative improvement because of the mutual reinforcement framework for attention localization and region-based feature learning. Note that RA-CNN (scale 2) outperforms VGG-19 results at scale 1 with clear margins (5.9% relative gains), which shows the necessity for “looking closer” on fine-grained categories. RA-CNN (scale 3) slightly drops compared to RA-CNN (scale 2), because of the missing of structural information in global bird images. By combining features at three scales via a fully connected layer, we achieve the best 85.3% accuracy. Note that the superior result benefits from the complementary advantages from multiple scales. The combination of triple single-scale networks with different initial parameters only achieves 78.0%, 83.5%, and 82.0% for the first, second, and third scales, respectively. We can extend RA-CNN to more scales, but the performance saturates as discriminative information is encoded into the previous scales.

10.3.6 Experiments on Stanford Dogs

Stanford Dogs is another widely used dataset for fine-grained image recognition. It contains 120 dog species, with the training/testing split of 12,000 and 8580. The classification accuracy on the Stanford Dogs dataset is summarized in Table 10.4. The VGG-16 at the first scale takes the original images as input and achieves 76.7% recognition accuracy. Relying on accurate attention localization, RA-CNN (scale 2) achieves a significant improvement to recognition accuracy of 85.9%, with 12.0% relative gain. By combining the features from two scales and three scales, we can boost the performance to 86.7% and 87.3%, respectively. Comparing this with the two most relevant approaches, DVAN [39] and FCAN [35], the relative accuracy gains are 7.1% and 3.7%, respectively.

This improvement mainly derives from the accurate attention localization, which is demonstrated in Figure 10.4. We can observe more significant visual cues (e.g., the heads of dogs) after gradually zooming into the attended regions at the second and third scales, which is consistent with previous research [39, 42].

Table 10.4 Comparison results on the Stanford Dogs dataset without an extra bounding box or part annotation.

Approach	Accuracy
NAC (AlexNet) [43]	68.6
PDFR (AlexNet) [37]	71.9
VGG-16 [27]	76.7
DVAN [39]	81.5
FCAN [35]	84.2
RA-CNN (scale 2) [16]	85.9
RA-CNN (scale 3) [16]	85.0
RA-CNN (scale 1+2) [16]	86.7
RA-CNN (scale 1+2+3) [16]	87.3



Figure 10.4 The learned region attention of dogs at different scales.

10.4 Deep Image Tagging for Fine-Grained Sentiment Analysis

We consider visual sentiment analysis as a binary prediction problem which classifies an image as positive or negative from its visual content [25, 44–46]. The difficulty is derived from the “affective gap” between the low-level visual content and high-level semantics [47]. Significant progress has been made by designing the sentiment-related visual sentiment ontology, which consists of more than 3000 adjective noun pairs (ANPs) with associated training images from search engines (see chapter 6, ref. [33]). Each ANP is considered to be an affective concept, and thus sentiment prediction can be conducted by determining whether an image can be classified with the affective concepts. For example, images classified with the ANP of “beautiful sky” are positive, while images classified with “terrible accident” are negative.

Although ANP-based representation has made sentiment analysis more visually detectable, general image recognition methods can only achieve limited performance. First, the same positive/negative sentiment can be reflected in different objects, which results in large intra-class variance. Second, different sentiments can be inferred from the same object, and hence visual sentiment analysis needs to detect subtle differences even in the same object class.

In this section, we introduce the deep-coupled adjective and noun networks (DCAN) for visual sentiment analysis [26]. Images with sentiment labels (positive/negative) are first fed into two sub-networks (A-net and N-net) to extract sentiment representation, as shown in Figure 10.5a–c. The outputs of both A-net and N-net are further normalized and concatenated in Figure 10.5f, and finally mapped to sentiment in Figure 10.5g. To effectively guide the training process, a weak supervision is used in Figure 10.5d and e, if noisy ANP labels are available.

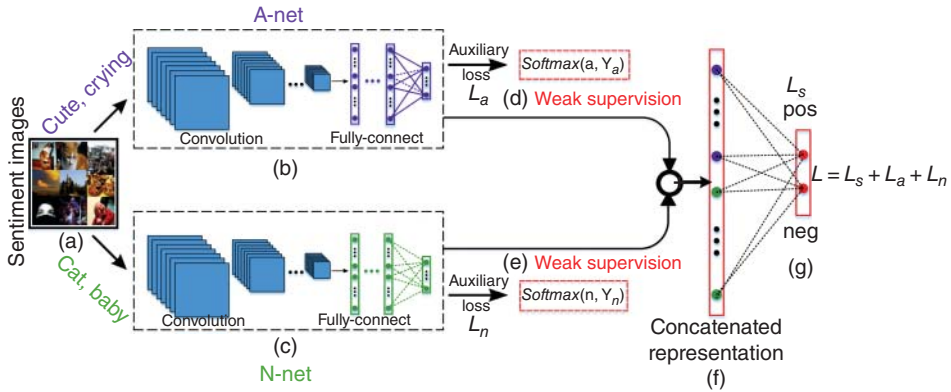


Figure 10.5 The deep-coupled adjective and noun neural network (DCAN). Images in (a) are fed into two sub-nets to extract the descriptiveness features in (b) and the objectiveness features in (c), respectively. The learnt sentiment features are further concatenated in (f) and finally mapped to sentiment in (g). Weak supervision for adjectives in (d) and for nouns in (e) are applied when noisy labels Y_a and Y_n are supplied. The network is optimized by minimizing the sentiment loss L_s and the two auxiliary losses L_a and L_n .

10.4.1 Learning Deep Sentiment Representation

Unlike traditional image recognition tasks, where images of the same object class often share highly similar visual patterns such as contour and texture, image sentiment analyses usually involve great intra-class variance. Given the sentiment supervision (positive/negative), it is hard to learn a mapping function from low-level image pixels to high-level sentiment space, even for powerful deep learning networks. Previous work has shown that a deeper network even leads to a worse result if the appropriate guidance is missing [46].

To utilize middle-level features to guide the sentiment learning, we propose learning a fine-grained sentiment representation with joint adjective/noun descriptions. Once the adjective/noun descriptions are learned, we consider them as middle-level representation to guide the learning of high-level sentiment. The network structure is shown in Figure 10.5. We divide each ANP label into an adjective and a noun, and leverage the two types of labels as weak supervision, since the ANP labels are noisy. Sentiment representation of an image is extracted by two parallel sub-networks (i.e., A-net and N-net) with convolutional and fully connected layers, under the supervision of adjectives (A) and nouns (N), respectively. The loss of each sub-network is measured by the cross entropy:

$$L_{a/n} = L(z, t; \mathbf{X}, \mathbf{W}) = -\log P(z = t | \mathbf{X}, \mathbf{W}), \quad (10.14)$$

$$P(z = t | \mathbf{X}, \mathbf{W}) = \text{softmax}(\mathbf{z}_k) = \frac{\exp(\mathbf{z}_k)}{\sum_{i=1}^K \exp(\mathbf{z}_i)}, \quad (10.15)$$

where \mathbf{z} is a K-dim network output, \mathbf{X} represents an input image, \mathbf{W} denotes network parameters, and z and t are the predicted and true labels, respectively.

The advantages of jointly learning adjective and noun networks are two-fold. First, images with the same adjective/noun label can share similar visual patterns. For example, the images of both “sunny beach” and “sunny road” show similar visual patterns of “bright”, while the images of both “angry dog” and “happy dog” share the

common dog appearance. This strategy thus enables the sub-networks to effectively learn common representation for different images under the same adjective or noun. Second, from the perspective of sample distribution, compared to the images assigned by ANP labels, images under the same adjective/noun are much richer and thus benefit from sample expansion and balance. For example, “angry dog” contains fewer than 100 images in SentiBank (see chapter 6, ref. [33]), while there are thousands of samples under “angry” or “dog”.

10.4.2 Sentiment Analysis

Since visual sentiment analysis also needs to solve the fine-grained challenge, the learned representation of A-net and N-net are further concatenated and connected to an additional fully connected layers with a softmax function for binary prediction. Adjectives are usually related to descriptiveness, while nouns represent the objectiveness of an image. Therefore, combining the two kinds of representation could be a reasonable, and easy to learn, fine-grained and discriminative sentiment predictor, which can be represented as:

$$P_s = \text{softmax}(\mathbf{W}^{(M+1)} \cdot [\mathbf{a}; \mathbf{n}]), \quad (10.16)$$

$$L_s = -\log P_s, \quad (10.17)$$

where $\mathbf{W}^{(M+1)}$ is the parameter of the last fully connected layer and L_s is the loss. To end-to-end train the whole neural network, we integrate the loss of A-net, N-net, and sentiment by a linear combination, which is given by:

$$L = w_s L_s + w_a L_a + w_n L_n, \quad (10.18)$$

where w_s , w_a , and w_n are weights. We generally set w_s to be relatively larger for faster convergence and a better result. In the optimization, we adopt the stochastic gradient descent method to train our model.

10.4.3 Experiments on SentiBank

SentiBank is widely used and contains about half a million images from Flickr using the designed ANPs as queries (see chapter 6, ref. [33]). The sentiment label of each image is decided by the sentiment polarity of the corresponding ANP. We use this dataset for weak supervision, as the noisy ANP labels are provided. The training/testing split is 90% and 10%, respectively. In SentiBank, we separate the ANPs into about 180 adjectives and 300 nouns. The input images are first resized to 227×227 and mean-subtracted before propagating to the network. We train our model using a mini-batch of 256 and weight decay of 0.0005, as suggested by [11]. The initial learning rate is 0.005 and is divided by 10 every 75 epochs until convergence. We empirically set the weight of sentiment to 2 and the weight of the sub-networks to 1 since this weighting gives the best result.

Table 10.5 summarizes the performances of our approach and other methods. Compared to the others, DCAN (Alex) uses the AlexNet structure [11] as the A-net and N-net. It gives a result of 86.1%, which is much better than PCNN [46]. DCAN (pretrain) uses the same network structure as DCAN (Alex), instead of the pretrained A-net and N-net from ImageNet. It further leads to the best performance, with more than 9.0%

Table 10.5 The precision, recall, F1, and accuracy of different approaches to the SentiBank testing set.

Methods	Prec.	Rec.	F1	Acc.
2Conv+4FC [46]	0.714	0.729	0.722	0.718
PCNN [46]	0.759	0.826	0.791	0.781
Bilinear CNN [20]	0.723	0.787	0.754	0.750
CNN(ANP) [11]	0.843	0.843	0.843	0.843
DCAN(2Conv+4FC) [26]	0.755	0.719	0.737	0.742
DCAN (Alex) [26]	0.872	0.848	0.860	0.861
DCAN (pre-train) [26]	0.866	0.883	0.874	0.873

accuracy gain compared to PCNN [46]. The performance of DCAN (2Conv+4FC) is unsatisfactory and uses the network from [46] as the A-net and N-net because the two sub-networks have few convolutions and are inadequate to capture the adjective and noun features. However, the accuracy of DCAN (2Conv+4FC) is higher than the pure CNN of 2Conv+4FC (74.2% vs. 71.8%), which shows the effectiveness of building the parallel network. Similar observation has been found in bilinear CNN [20], which shows better results than the pure CNN methods. Our network DCAN (Alex) shows better performance than bilinear CNN with the same convolutional layers, which verifies that recognizing sentiment cannot simply be formulated as a fine-grained problem. Instead, we should simultaneously consider the three challenges, i.e., large intra-class variance, fine-grained recognition, and low-scalability. The proposed network also achieves superior performance over DCAN (ANP), which uses the same network as DCAN (Alex) instead of using each ANP as the middle representation. This result suggests that learning shared features for each adjective or noun can greatly promote the performances of neural networks.

10.5 Conclusion

In this chapter we focused on fine-grained image tagging by deep learning techniques, which can recognize different bird species, dog breeds, and human sentiment. Specifically, we proposed two carefully designed deep neural network architectures, a recurrent attention convolutional neural network and a deep coupled adjective and noun network. The proposed networks can capture the subtle visual differences in fine-grained recognition tasks by recurrently amplifying the attended regions and by jointly learning representative sentiment features from both adjectives and nouns. Extensive experiments have demonstrated the superior performances of the two deep neural networks on widely used fine-grained and visual sentiment datasets.

References

- 1 Lowe, D.G. (2004) Distinctive image features from scale-invariant keypoints. *International Journal of Computing Vision*, 60 (2), 91–110.
- 2 Felzenszwalb, P.F., Girshick, R.B., McAllester, D., and Ramanan, D. (2010) Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32 (9), 1627–1645.
- 3 Oliva, A. and Torralba, A. (2006) Building the GIST of a scene: The role of global image features in recognition. *Visual Perception, Progress in Brain Research*, 155.
- 4 Sivic, J. and Zisserman, A. (2003) Video Google: A text retrieval approach to object matching in videos, in *International Conference on Computer Vision*, pp. 1470–1477.
- 5 Lazebnik, S., Cordelia, S., and Jean, P. (2006) Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories, in *Conference on Computer Vision and Pattern Recognition*, pp. 2169–2178.
- 6 Song, J., Gao, L., Nie, F., Shen, H.T., Yan, and Sebe, N. (2016) Optimized graph learning using partial tags and multiple features for image and video annotation. *Transactions on Image Processing*, 25 (11), 4999–5011.
- 7 Zhang, Y., Gong, B., and Shah, M. (2016) Fast zero-shot image tagging, in *Computer Vision and Pattern Recognition*, pp. 5985–5994, IEEE Computer Society.
- 8 Fu, J., Wang, J., Rui, Y., Wang, X.J., Mei, T., and Lu, H. (2015) Image tag refinement with view-dependent concept representations. *IEEE Transactions on Circuits and Systems for Video Technology*, 25 (28), 1409–1422.
- 9 Fu, J., Wang, J., Li, Z., Xu, M., and Lu, H. (2013) Efficient clothing retrieval with semantic-preserving visual phrases, in *Asian Conference on Computer Vision*, pp. 420–431.
- 10 Bengio, Y. (2009) Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, 2 (1), 1–127.
- 11 Krizhevsky, A., Sutskever, I., and Hinton, G.E. (2012) Imagenet classification with deep convolutional neural networks, in *Advances in Neural Information Processing Systems*, pp. 1106–1114.
- 12 He, K., Zhang, X., Ren, S., and Sun, J. (2016) Deep residual learning for image recognition, in *Conference on Computer Vision and Pattern Recognition*, pp. 770–778.
- 13 Fu, J., Mei, T., Yang, K., Lu, H., and Rui, Y. (2015) Tagging personal photos with transfer deep learning, in *Proceedings of the World Wide Web*, pp. 344–354.
- 14 Fu, J., Wu, Y., Mei, T., Wang, J., Lu, H., and Rui, Y. (2015) Relaxing from vocabulary: Robust weakly-supervised deep learning for vocabulary-free image tagging, in *IEEE International Conference on Computer Vision*, 1985–1993.
- 15 Branson, S., Horn, G.V., Belongie, S.J., and Perona, P. (2014) Bird species categorization using pose normalized deep convolutional nets, in *British Machine Vision Conference*.
- 16 Fu, J., Zheng, H., and Mei, T. (2017) Look closer to see better: Recurrent attention convolutional neural network for fine-grained image recognition, in *Conference on Computer Vision and Pattern Recognition*, 4476–4484.

- 17 Nilsback, M.E. and Zisserman, A. (2006) A visual vocabulary for flower classification, in *Conference on Computer Vision and Pattern Recognition*, pp. 1447–1454.
- 18 Reed, S.E., Akata, Z., Schiele, B., and Lee, H. (2016) Learning deep representations of fine-grained visual descriptions, in *Conference on Computer Vision and Pattern Recognition*, 49–58.
- 19 Krause, J., Jin, H., Yang, J., and Li, F.F. (2015) Fine-grained recognition without part annotations, in *Conference on Computer Vision and Pattern Recognition*, pp. 5546–5555.
- 20 Lin, T.Y., RoyChowdhury, A., and Maji, S. (2015) Bilinear CNN models for fine-grained visual recognition, in *International Conference on Computer Vision*, pp. 1449–1457.
- 21 Datta, R., Li, J., and Wang, J.Z. (2008) Algorithmic inferencing of aesthetics and emotion in natural images: An exposition, in *International Conference on Image Processing*, pp. 105–108.
- 22 Ko, E. and Kim, E.Y. (2015) Recognizing the sentiments of web images using hand-designed features, in *2015 IEEE 14th International Conference on Cognitive Informatics Cognitive Computing (ICCI*CC)*, pp. 156–161.
- 23 Siersdorfer, S., Minack, E., Deng, F., and Hare, J. (2010) Analyzing and predicting sentiment of images on the social web, in *ACM Multimedia*, pp. 715–718.
- 24 Morency, L.P., Mihalcea, R., and Doshi, P. (2011) Towards multimodal sentiment analysis: Harvesting opinions from the web, in *Proceedings of the 13th International Conference on Multimodal Interfaces*, pp. 169–176.
- 25 Yuan, J., McDonough, S., You, Q., and Luo, J. (2013) Sentribute: image sentiment analysis from a mid-level perspective, in *Workshop on Issues of Sentiment Discovery and Opinion Mining*, p. 10.
- 26 Wang, J., Fu, J., Mei, T., and Xu, Y. (2016) Beyond object recognition: Visual sentiment analysis with deep coupled adjective and noun neural networks, in *International Joint Conference on Artificial Intelligence*, 3484–3490.
- 27 Simonyan, K. and Zisserman, A. (2015) Very deep convolutional networks for large-scale image recognition, in *International Conference on Learning Representations*, pp. 1409–1556.
- 28 Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2014) Going deeper with convolutions. *arXiv preprint arXiv:1409.4842*.
- 29 Rumelhart, D., Hinton, G., and Williams, R. (1986) Learning representations by back-propagating errors. *Nature*, 323, 533–536.
- 30 Girshick, R.B. (2015) Fast R-CNN, in *International Conference on Computer Vision*, pp. 1440–1448.
- 31 Liu, Y., Fu, J., Mei, T., and Chen, C. (2017) Let your photos talk: Generating narrative paragraph for photo stream via bidirectional attention recurrent neural networks, in *Proceedings of the 31st Conference on Artificial Intelligence (AAAI), February 4–9, San Francisco, CA, USA, , pp. 1445–1452*.
- 32 Shih, K.J., Singh, S., and Hoiem, D. (2016) Where to look: Focus regions for visual question answering, in *Computer Vision and Pattern Recognition*, 4613–4621.
- 33 Yu, D., Fu, J., Mei, T., and Rui, Y. (2017) Multi-level attention networks for visual question answering, in *Conference on Computer Vision and Pattern Recognition*, 4187–4195.

- 34 Cortes, C. and Vapnik, V. (1995) Support-vector networks. *Machine Learning*, 20 (3), 273–297.
- 35 Liu, X., Xia, T., Wang, J., and Lin, Y. (2016) Fully convolutional attention localization networks: Efficient attention localization for fine-grained recognition. *Computing Research Repository*, abs/1603.06765.
- 36 Wang, D., Shen, Z., Shao, J., Zhang, W., Xue, X., and Zhang, Z. (2015) Multiple granularity descriptors for fine-grained categorization, in *International Conference on Computer Vision*, pp. 2399–2406.
- 37 Zhang, X., Xiong, H., Zhou, W., Lin, W., and Tian, Q. (2016) Picking deep filter responses for fine-grained image recognition, in *Conference on Computer Vision and Pattern Recognition*, pp. 1134–1142.
- 38 Jaderberg, M., Simonyan, K., Zisserman, A., and Kavukcuoglu, K. (2015) Spatial transformer networks, in *Advances in Neural Information Processing Systems*, pp. 2017–2025.
- 39 Zhao, B., Wu, X., Feng, J., Peng, Q., and Yan, S. (2016) Diversified visual attention networks for fine-grained object classification. *Computing Research Repository*, abs/1606.08572.
- 40 Zhang, N., Donahue, J., Girshick, R.B., and Darrell, T. (2014) Part-based R-CNNs for fine-grained category detection, in *European Conference on Computer Vision*, pp. 1173–1182.
- 41 Zhang, H., Xu, T., Elhoseiny, M., Huang, X., Zhang, S., Elgammal, A., and Metaxas, D. (2016) SPDA-CNN: Unifying semantic part detection and abstraction for fine-grained recognition, in *Conference on Computer Vision and Pattern Recognition*, pp. 1143–1152.
- 42 Xiao, T., Xu, Y., Yang, K., Zhang, J., Peng, Y., and Zhang, Z. (2015) The application of two-level attention models in deep convolutional neural network for fine-grained image classification, in *Conference on Computer Vision and Pattern Recognition*, pp. 842–850.
- 43 Simon, M. and Rodner, E. (2015) Neural activation constellations: Unsupervised part model discovery with convolutional networks, in *International Conference on Computer Vision*, pp. 1143–1151.
- 44 You, Q., Luo, J., Jin, H., and Yang, J. (2015) Joint visual-textual sentiment analysis with deep neural networks, in *ACM Multimedia*, pp. 1071–1074.
- 45 Campos, V., Salvador, A., Giro-i Nieto, X., and Jou, B. (2015) Diving deep into sentiment: understanding fine-tuned CNNs for visual sentiment prediction, in *Proceedings of the 1st International Workshop on Affect and Sentiment in Multimedia*, pp. 57–62.
- 46 You, Q., Luo, J., Jin, H., and Yang, J. (2015) Robust image sentiment analysis using progressively trained and domain transferred deep networks, in *Conference on the Association for the Advancement of Artificial Intelligence*, 381–388.
- 47 Machajdik, J. and Hanbury, A. (2010) Affective image classification using features inspired by psychology and art theory, in *ACM Multimedia*, pp. 83–92.

11

Visually Exploring Millions of Images using Image Maps and Graphs

Kai Uwe Barthel and Nico Hezel

Humans can easily understand complex pictures, but they have great difficulty dealing with large amounts of unorganized individual images. Users are confronted with large image sets when searching photos in image archives or when trying to find particular scenes in a video collection. In these cases, searching for particular images can be very time consuming. As human perception is limited, overview is quickly lost if too many images are shown at the same time. Up to now very large image and video archives have not offered visual browsing or exploration of the entire collection.

Typically, images are represented by their keywords and/or high-dimensional feature vectors to describe their look and content. Content-based image retrieval schemes compare these feature vectors to describe image similarities. While there has been a lot of effort to improve visual search, there is not much support for exploratory image search.

Sorting images by similarity can help users to view and recognize more images simultaneously. However, conventional dimensionality reduction schemes that project high-dimensional data to two dimensions cannot be used for image sorting because their projections result in unequally distributed and overlapping images. In addition, these approaches suffer from very high complexity. If images are to be sorted on a dense regular grid, only self-organizing maps (SOMs), self-sorting maps (SSMs) or discrete optimization algorithms can be used. When dealing with collections in the range of millions of images, hierarchical visualization and navigation techniques need to be used.

In the first part of this chapter we describe different image sorting algorithms. For evaluating the quality of these sorted image arrangements, common quality measures such as the normalized cross-correlation or the normalized energy function are not useful to assess the visual sorting quality as perceived by the users. We therefore introduce a new measure, which is better suited to evaluate two-dimensional (2D) image arrangements. In addition, we present a modified SSM algorithm with improved sorting quality and reduced complexity, which allows millions of images to be sorted very quickly.

However, 2D projections in general cannot preserve the complex relationships between all images. Another important disadvantage of static regular 2D arrangements is their inability to handle changes of the image collection. Removed images will result in holes in the image map. For newly added images a new sorting process becomes necessary, which is time consuming and will change the previous order of the images.

Graph-based approaches can handle changes in the image collection. They are also able to better preserve and represent image relationships. However, it is not clear how to visualize the “high-dimensional” graph in such a way that it can easily be perceived and navigated by the user.

In the second part of this chapter we present a new graph-based approach to visually browse very large sets of varying (untagged) images. We show how high-quality image features representing the image content can be generated using transformed activations of a convolutional neural network. These features are used to model image similarities, from which a hierarchical image graph is built. We propose a new algorithm to construct such an image graph very efficiently. For the navigation and visualization of the graph we performed several experiments. We found that the best user experience for navigating the graph is achieved by dynamically projecting subgraphs onto a regular 2D image map, using the new improved sorting methods. This allows users to explore the image collection like using an interactive mapping service.

11.1 Introduction and Related Work

The increasing use of digital images has led to a growing problem of how to organize large sets of images so that they can easily be perceived by the viewer. If no specific order of the images, such as a chronological sequence, is available, it is not obvious how the images should be arranged for display. Also, for videos the set of all keyframes is far too



Figure 11.1 1024 images tagged with “flower” shown in random order. Overview is quickly lost.

large to be examined easily. For unsorted images, human perception is limited to 20 to 50 images at a time. Due to this, most websites and image management applications usually display about 20 images per page or screen. If too many images are shown, overview is quickly lost (see Figure 11.1).

If, however, images are arranged by their similarity, up to several hundred images can be perceived simultaneously (see Figures 11.2 and 11.3). The sorted arrangement helps the user to identify regions of interest more easily. The opportunity to present more images is especially interesting for e-commerce applications, stock agencies, and image management systems.

In order to arrange images by similarity, a measure has to be defined to describe the similarity between images. When dealing with tagged images, their similarities can be determined using document retrieval methods such as the Cosine similarity. Often, however, image keywords are inaccurate or may not be available at all. In this case automatic image analysis can be used to generate visual feature vectors. Depending on the type of image analysis, the meaning and/or the visual appearance of images can be described. Again, an appropriate metric is needed to compare the vectors. The dimensionality of these vectors ranges from several tens for low-level CBIR-features up to hundreds or thousands for features generated by deep-learning neural networks.

If images are represented as high-dimensional datapoints, their relationships can be expressed by appropriate visualization techniques. This is particularly important for high-dimensional data that lie on several different, but related, lower-dimensional



Figure 11.2 The same 1024 images projected by the t-SNE algorithm.

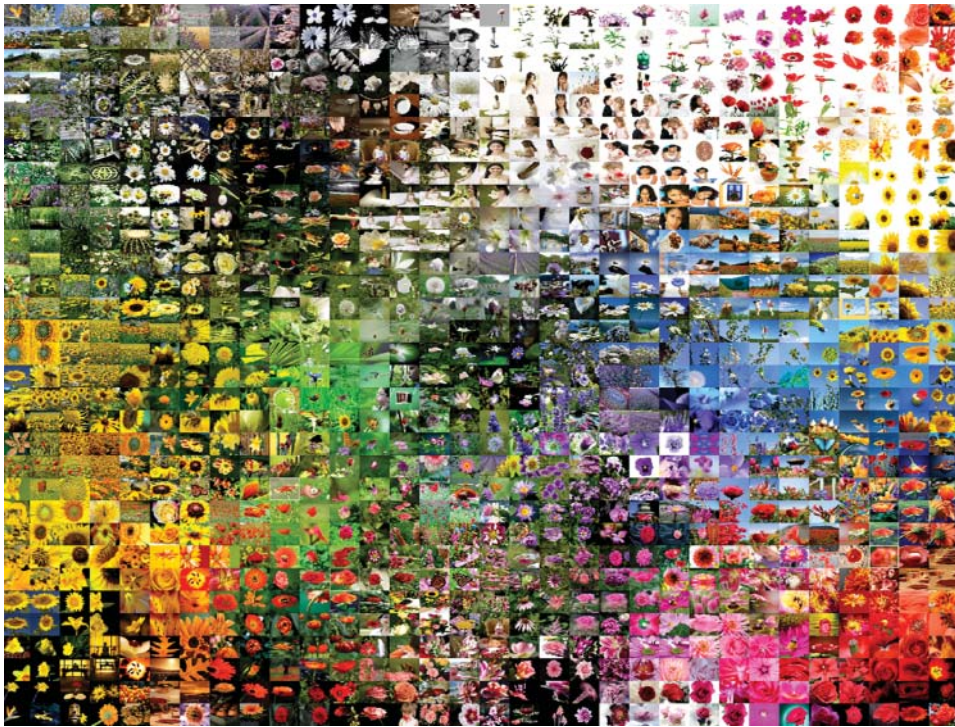


Figure 11.3 1024 “flower” images projected by the proposed image sorting algorithm.

manifolds. Over the last few decades, a variety of *dimensionality reduction techniques* have been proposed. Many of them have been reviewed by de Oliveira and Levkowitz [1]. Visualization is achieved by projecting the high dimensional data onto a 2D or 3D map. Most dimensionality reduction techniques (except principal component analysis (PCA)) [2] determine some non-linear transformation to map the high-dimensional datapoints to lower dimensional spaces. Common techniques are multidimensional scaling, Sammon mapping [3], Isomap [4], local-linear-embedding (LLE) [5], t-distributed stochastic neighbor embedding (t-SNE) [6], and auto encoders.

An overview of various visual browsing models for image exploration is given in [7, 8]. Some authors use visual attributes to split images into subsets [9]. Others [10, 11] use techniques such as Isomap or SOMs to generate visually sorted arrangements of search results. These approaches help to get a better overview, but they also suffer from unequally positioned and overlapping images. In all cases, only a few images of the entire image set are shown and there is no way to experience the relationships with other images.

Figure 11.2 shows a visualization using the similarity relations of 1024 images. The images were represented by 50-dimensional low-level feature vectors, which are described in section 11.5.1. The 2D-projection was generated using the t-SNE algorithm [6], which is known for very good visualizations. Due to the dense positions of the projected images, some images do overlap and therefore are partially hidden. The actual area used by this projection is only a fraction of the total available display area

(in this case less than 50%). The overlap of the images could be lowered by displaying them with smaller sizes. However, this would further reduce the perceptibility and the used display area.

In order to sort a set of images by similarity and to use the display area in the best possible way, three requirements need to be satisfied:

- 1) There should not be any overlap of the images.
- 2) The images should completely cover the display area without any gaps.
- 3) Neighboring images should be as similar as possible.

Dimensionality reduction techniques meet requirement 3, but they cannot be used for 2D image sorting due to the first two requirements. Theoretically these techniques could be used in a first step. A second step would then be required to place the images onto a regular 2D grid, which subsequently repositions the images in such a way that they do not overlap. However, this step does change and degrade the projection quality.

Requirements 1 and 2 are met by any arrangement of the images on a rectangular grid. However, the third requirement needs careful selection of how to arrange the images. Due to the factorial number of possible solutions, the optimal image arrangement cannot be determined, but in most cases an approximate solution will be sufficient. To generate sorted image arrangements with non-overlapping images, discrete mapping algorithms have to be used. SOMs [12] can be adapted to fulfill the mentioned requirements for non-overlapping image sorting. However, classical SOMs suffer from two problems: their complexity is rather high and their projection quality is not too good. SSMs use a much faster algorithm to arrange images without overlap by iteratively swapping image positions [13]. Another approach for generating sorted image arrangements is discrete optimization algorithms. Kernelized sorting could also be used, but its computational complexity is high, making it infeasible for very large image sets [14].

We will discuss different image-sorting methods and propose a new way to evaluate the sorting/projection quality. We show how SOMs can significantly be improved regarding speed and the achieved image sorting quality. By fusing SOMs and SSMs, very fast image sorting with high projection quality becomes possible. Figure 11.3 shows a result of the new approach using the same images as in Figures 11.1 and 11.2.

11.2 Algorithms for Image Sorting

11.2.1 Self-Organizing Maps

A self-organizing map is an artificial neural network that is trained using unsupervised learning to produce a lower dimensional, discrete representation of the input space, called a map. SOMs differ from other artificial neural networks as they are trained by competitive as opposed to error-correction learning, and they use a neighborhood function to preserve the topological properties of the input space.

A SOM consists of components called nodes. Associated with each node are a weight vector of the same dimensionality as the input data vectors and a position in the map space. The common arrangement of the SOM nodes is a 2D rectangular grid. The mapping produced by the SOM is discrete as each input vector is assigned to one particular node. After successive iterations the locations of the connected weight vectors of the SOM nodes tend to become ordered in the high-dimensional space.

In the beginning the weight vectors of all nodes are initialized with small random values. When a high-dimensional input vector X is fed to the network, its Euclidean distance to all weight vectors is computed. The node whose weight vector is most similar to the input is called the best matching unit (BMU). The weights of the BMU and the nodes close to it on the SOM grid are adjusted towards the input vector. The magnitude of the change decreases with time and distance from the BMU (within the grid). The update for a weight vector $W_{x,y}$ for a node of the grid position x, y at time $t + 1$ is

$$W_{x,y}^{t+1} = W_{x,y}^t + \alpha(t) \cdot \gamma(t, d)(X - W_{x,y}^t) \quad (11.1)$$

where $\alpha(t)$ is the adaptation step size. The distance to the BMU in the lower-dimensional map space (the SOM grid) is denoted by d . $\gamma(t, d)$ denotes the neighborhood function, which is larger for nodes close to the BMU and smaller for nodes further away. Often the following adaptation rules are used

$$\alpha(t) = \alpha_{start} \left(\frac{\alpha_{end}}{\alpha_{start}} \right)^{t/t_{max}} \quad (11.2)$$

$$\gamma(t, d) = e^{-\frac{d}{\lambda(t)}} \quad \lambda(t) = \lambda_{start} \left(\frac{\lambda_{end}}{\lambda_{start}} \right)^{t/t_{max}} \quad (11.3)$$

Typical values are $\alpha_{start} = 0.5$, $\alpha_{end} = 0.005$, $\lambda_{start} = \sqrt{n}/2$, and $\lambda_{end} = 0.01$, where n is the number of map nodes. For each iteration the assignments to the nodes are determined in the previously described way. The maximum number of iterations t_{max} is typically set in the range from 10 to 50.

To adapt SOMs to image sorting, two minor modifications are necessary. Each node must not be selected by more than one feature vector (representing an image). Allowing a multiple selection would result in overlap of the images. Obviously, because of this the number of SOM nodes has to be equal to or larger than the number of images. A useful extension of a SOM is a grid where opposite edges are connected. If these toroidal grids are used for very large SOMs, the impression of an endless map can be created. For very large SOMs, hierarchical views of the sorted images may be useful, where for each set of 2×2 images the most representative image is shown on the next higher level.

11.2.2 Self-Sorting Maps

SSMs [13] arrange images by randomly filling cells (discrete positions) with images and then splitting all cells into 4×4 blocks, each of which will be further split into four smaller blocks in the next stage. For each block the target (the mean of the feature vectors) is computed. Then even-indexed blocks are grouped with odd-indexed blocks along both x and y directions. This is followed by a shifted grouping in which odd-indexed blocks are grouped with even-indexed blocks along both directions. In each step possible cell (image) swaps are checked between grouped blocks by using the targets as guides. The feature vectors in the corresponding cells of the grouped blocks form quadruples. From the $4! = 24$ possibilities the swap of positions is selected that minimizes the sum of the squared differences between the feature vectors and the targets. After several iterations each block is split into four smaller blocks of half the size in width and height. This process is iterated until the block size has been reduced to 1.

11.2.3 Evolutionary Algorithms

Evolutionary strategy (and other discrete optimization) algorithms try to find an optimal solution from a finite set of possible solutions. Even if an exhaustive search is not feasible, the goal is to find a good approximation. Common problems are the traveling salesman problem and the minimum spanning tree problem.

The optimization problem is solved by iteration. Within each step a population of candidate solutions is evolved towards a better adopted population. Each candidate solution has a set of properties which can be changed. The evolution usually starts from a population of randomly generated individuals. In each step of this iterative process, the fitness of every individual is evaluated. The fittest individuals are selected and modified to form a new generation, which then is used in the next iteration step. The algorithm terminates when either a maximum number of iterations or a satisfactory fitness has been reached. A discrete optimization algorithm requires a representation of the solution domain and a fitness or quality function to evaluate the solutions. When using discrete optimization for image sorting, the representation of the solution is the actual mapping of the images (their positions). The function to determine the quality of the image arrangement will be defined in the next section.

A simple discrete optimization algorithm for image sorting could iteratively apply a set of random swaps of image positions. The swap with the best quality improvement is kept, and a new iteration is started. Typically, discrete optimization algorithms can find very good solutions but take a long time.

11.3 Improving SOMs for Image Sorting

When comparing different image sorting algorithms both the complexity and the sorting quality (the projection quality) have to be taken into account. We show how to significantly reduce the sorting complexity and how to improve the sorting quality of a SOM. In section 11.4 a new way to determine the projection quality of sorted image sets is presented. Section 11.5 will compare our results with conventional SOMs and other algorithms.

11.3.1 Reducing SOM Sorting Complexity

As described before, sorting images with a SOM consists of an iterative application of two steps. For each image feature vector its BMU has to be *searched*. This is done by computing the distances to all the node weight vectors and then choosing the node with the minimum distance. In the next step all the node weight vectors have to be *updated* according to Eqn (11.1). This update of the weight vectors can be seen as a *filtering* operation: the difference between the feature vector and the BMU's weight vector is convolved with the kernel of the neighborhood function $\gamma(t, d)$. The result of this convolution is then scaled by the adaptation step size $\alpha(t)$ and added to the nodes' weight vectors. The total number of operations $T(n)$ for sorting n images using a conventional SOM with n nodes therefore is

$$T(n) = I \cdot n \cdot \dim \cdot (n \cdot k_{search} + n \cdot k_{filter}) \quad (11.4)$$

where I is the number of iterations, dim is the dimensionality of the feature vectors, and k_{search} and k_{filter} are constants. Thus, the complexity of a conventional SOM is $O(n^2)$. In order to accelerate the sorting process two approaches are possible.

To speed up the process for finding the BMUs we propose a hierarchical search scheme similar to hierarchical block matching used for video coding. By successively reducing the size of the SOM by a factor of two in each dimension we construct a SOM with L hierarchy levels. The weight vector of a node of the next hierarchy level is determined by averaging the corresponding 2×2 weight vectors of the actual level.

For finding the BMU for an input feature vector, we start with a full search on the highest hierarchy level of the SOM, where only a fraction of the total number of nodes ($n/4^L$) needs to be evaluated. Then for the best matching position of the current level the search is continued on the next lower hierarchy level. However, this time only the 6×6 nodes in proximity to the previous best position of the higher level are examined. This local search is continued until the lowest level has been reached, where the best node is chosen as the BMU.

In order to guarantee that every node of the lowest level is not selected by more than one feature vector, the higher-level nodes need to be aware if unselected nodes are available on lower levels. This can be realized by adding a capacity counter to every node. Nodes on the lowest level have an initial capacity 1, the nodes of the next levels have a capacity of 4, then 16, etc. During the search only nodes with sufficient capacity need to be examined. This hierarchical approach will reduce the search complexity from $O(n^2)$ to $O(n \log(n))$. Theoretically the hierarchical search may compromise the quality of the SOM projection because not all nodes are examined, however our experiments showed that this is not the case (see section 11.5).

The other possibility to reduce SOM complexity is to simplify the update/filtering process. Instead of applying an “online” filtering mode, where for each input vector all node weights are changed directly after the assignment of a BMU, we propose a batch mode. This means: for each BMU its weight vector is replaced with the associated input feature vector. When all BMUs have been assigned, their weight vectors are simultaneously filtered with the smoothing kernel according to the neighborhood function. In order to further speed up the filtering process we evaluated different kernels. Convolutions can be computed faster if the kernels are separable and even faster if all kernel elements are equal. The actual smoothing kernel type is not critical as long its size is reduced during the iterations. We use repeated convolutions with box filters with decreasing sizes. Depending on the number of repetitions, kernels with box, pyramidal, and up to Gauss shape can be realized [15]. The advantage of a box filter is that it can be realized with constant complexity – independent of the kernel size – by using an integral image.

Applying the proposed optimizations, the number of operations for sorting n images is reduced to

$$T_{opt}(n) = I \cdot n \cdot dim \cdot (\log(n) \cdot k_{search} + k_{filter}) \quad (11.5)$$

yielding a total complexity of $O(n \log(n))$, which allows large sets of images to be sorted much faster.

11.3.2 Improving SOM Projection Quality

In this section we will describe the implementation of the proposed image sorting scheme as it is used to produce the results presented in section 11.5. For our experiments we use a square-shaped SOM with $N \times N$ nodes (equal to the number of images to be sorted). The number of hierarchy levels is determined by the number of images in such a way that the top level has at least 8×8 nodes. For our experiments we used 30 iterations. Higher numbers of iterations result in slightly improved results at the cost of longer processing time. For the first iteration ($t=0$) all BMUs are assigned randomly. For the following iterations ($t \geq 1$) the search for the BMU was performed as described in the previous section, starting at the top level. A position on a higher level corresponds to 2×2 nodes on the next lower level. For the local search on the lower levels we increase the search area by two positions in each direction (resulting in a search area of 6×6 nodes). After all BMUs have been assigned, the node weight vectors contain the copies of the image feature vectors. In the next step these weight vectors are filtered. As mentioned before we use a box filter with an initial radius of $0.2N$. Three filter repetitions result in a kernel corresponding to a simple approximation of a Gaussian kernel. For each iteration step t the size/radius is linearly reduced

$$rad(t) = \max \left(1, 0.2 \cdot N \cdot \left(1 - \frac{t}{I-2} \right) \right) \quad (11.6)$$

One problem of using this approach is that the number of available free nodes linearly decreases from the first to the last input vector. While the first vector may choose any node, for the last there is only one remaining unassigned node. To address this problem, the images are shuffled at the beginning of each iteration. Nevertheless, the projection quality is affected by this restriction.

Inspired by the discrete optimization algorithms we have developed a solution to cope with this problem. After the last iteration we apply a so-called *cleanup path*, trying to switch positions to increase the sorting quality. We randomly choose nodes which have a distance between their weight vector and those of their neighbors that is higher than the average neighbor weight vector distance. For the chosen nodes again a hierarchical search is done to find potential swap candidates. Positions are swapped if this increases the overall projection quality. Typically, a number of swap tries in the order of ten times the number of images is sufficient to find better places for most poorly assigned vectors/images.

11.3.3 Combining SOMs and SSMs

In the last section we have shown how to speed up a SOM by using a hierarchical search and filtering the SOM using an integral image. An SSM has a complexity of $n \log(n)$ as well, but in practice is faster than a hierarchical SOM. Therefore, we tried to combine the advantages of both schemes. We keep the swapping approach of the SSM but changed the target calculation. Instead of calculating the target (the mean of the feature vectors) as a constant mean vector for the entire block region, we use a sliding filter, which again can be calculated using integral images. Instead of keeping the block size constant for

several iterations we continuously reduce the block size. In [13] blocks at positions x and y are grouped with successive blocks at positions $x + 1$ and $y + 1$. For a block at positions x and y we additionally check blocks at positions $x, y + 1$ and $x + 1, y$. This doubles the number of comparisons but improves the projection quality. The effect on the computational complexity is almost negligible. To further improve the quality, the described final cleanup path for swapping image pairs can be applied after the sorting process. Timing and quality results are given in section 11.5.2.

11.4 Quality Evaluation of Image Sorting Algorithms

To compare the quality of different image sorting algorithms quality measures are needed. We discuss the drawbacks of existing measures and propose an improved method for determining the projection quality of sorted image arrangements.

11.4.1 Analysis of SOMs

To evaluate the quality of a SOM, typically two quality measures, the *U-matrix* and the *quantization error*, are used. The U-matrix represents the local differences between the node weight vectors of the SOM. The U-matrix value of a particular node at position x, y of the map is the average distance between the node weight vector and that of its closest neighbors in the map. In a rectangular map, the region R of the closest four or eight nodes is considered:

$$U_{x,y} = \frac{1}{|R|} \sum_{(i,j \in R)} W_{x,y} - W_{x+i,y+j} \quad (11.7)$$

Regions of low values represent similar weight vectors, whereas higher values indicate discontinuities in the map. The average of all neighbor distances \bar{U} describes the “evenness” of the SOM embedding in the high-dimensional space. A low \bar{U} value represents a smooth SOM embedding, which is desirable.

The other measure that is often used to evaluate SOMs is the quantization error. This indicates how well the nodes of the trained network are adapted to the input vectors. The quantization error E_q is defined by the average distance between the n data vectors X and the respective weights of their BMUs. Again, a low value of the quantization error is desirable.

$$E_q = \frac{1}{n} \sum_{k=1}^n \|X_k - W_{BMU_k}\| \quad (11.8)$$

Both previously described measures are not helpful to evaluate the SOM projection quality in the case of image sorting where the number of SOM nodes is equal to or greater than the number of images. Apparently, a low average neighbor distance and a low quantization error are indicators for a good SOM projection quality. The first condition guarantees a smooth embedding, whereas the second condition assures a good mapping of the input vectors to the SOM’s weight vectors. However, depending on the training parameters, either the first or the second requirement can easily be met. A very

wide neighborhood function γ results in a flat plane-like embedding with a \bar{U} value close to zero. On the other hand, a very high adaptation step size α will make all weight vectors identical to the input vectors, leading to $E_q = 0$. This means, if only one of these two measures has a low value, this does not necessarily imply that the SOM projection is good.

11.4.2 Normalized Cross-Correlation

One approach to measure the sorting quality is to compute the normalized cross-correlation between the distance of the projected positions $Y_k = P(X_k)$, $Y_l = P(X_l)$ and the original distances $\delta(X_k, X_l)$ of the two feature vectors of the images k and l . \bar{P} is the mean distance between any two projected image positions and $\bar{\delta}$ is the mean distance between any two feature vectors in the high-dimensional space. σ_p and σ_δ are the corresponding standard deviation values which normalize the cross correlation ρ to $-1 \dots 1$.

$$\rho = \sum_{\forall k,l} \frac{(\|P(X_k)\| - \bar{P})(\|X_k - X_l\| - \bar{\delta})}{\sigma_p \sigma_\delta} \quad (11.9)$$

One problem of normalized cross-correlation is the fact that low and high distances are treated equally. In practice it is important that very similar images stay close to each other, whereas a change of the projected distance for very different images k and l is a less serious problem. The same is true for the normalized energy function, which is also used to measure image sorting quality.

11.4.3 A New Image Sorting Quality Evaluation Scheme

The most important aspect of image sorting is to preserve the local neighborhood relations of the input feature vectors X describing the images. Therefore, a good way to determine the image sorting quality (i.e., the projection quality) is to check how well the projection P is capable of capturing the local neighborhood structure of the high-dimensional data. In the high-dimensional space, neighboring vectors obviously are more similar to each other than vectors that are further away. To evaluate the quality of image arrangements for the projected vectors on the low-dimensional map, we request that nearby images in the high-dimensional space stay close to each other on the 2D map. For each vector X_k we sort (rank) all other vectors X_l according to their high-dimensional L2 distance to X_k .

Let $r = \text{rank}(X_k, X_l)$ be the rank of X_l in the sorted list of distances to X_k . We define a function $h(r)$ to weight these distances according to their rank. $h(r)$ is larger for lower ranks (smaller distances) and smaller for higher ranks (larger distances). This means that the distances of neighboring vectors are taken into account more than those for vectors that are further away. The sum H of the weights $h(r)$ is used to normalize the weighted distances:

$$H = \sum_r h(r) \quad (11.10)$$

We compute the weighted root mean square distance $D_h(X)$ for all n vectors X in the original high-dimensional space:

$$D_h(X) = \left(\frac{1}{n} \frac{1}{H} \sum_{k=1}^n \sum_{l=1}^n h(\text{rank}(X_k, X_l)) \|X_k - X_l\|_2^2 \right)^{\frac{1}{2}} \quad (11.11)$$

The same is done for all projected vectors of the low-dimensional map, where each input vector is mapped to discrete positions. Again, for each high-dimensional vector we compute the weighted root mean square distance to all other vectors. However, this time the weighting function $h_{2D}(d)$ is based on the distances of the node positions Y_k, Y_l on the low-dimensional map. The distance $d = \text{dist}_{2D}(Y_k, Y_l)$ between any two nodes is computed as the L2 distance of their 2D map positions. The weighted root mean square distance of the projection P is

$$D_{h_{2D}}(P, X) = \left(\frac{1}{n} \frac{1}{H} \sum_{k=1}^n \sum_{l=1}^n h_{2D}(\text{dist}_{2D}(Y_k, Y_l)) \|X_k - X_l\|_2^2 \right)^{\frac{1}{2}} \quad (11.12)$$

It is important to ensure that the weighting function values are the same for the rank r and the map distance dist_{2D} . As there are multiples of identical L2 distances on the 2D map, the 2D dimensional weighting function h_{2D} has to take this into account.

The weighting function should be decreasing for larger distances of the nodes. We chose a Gaussian weighting function:

$$h(r') = e^{-(2^w r')^2} \quad \text{with } r' = \frac{\text{rank}}{\text{maxRank}} \quad w = 0, 1, 2, \dots, 9 \quad (11.13)$$

where r' is the normalized rank between 0 and 1. The factor w controls the width of the weighting function. Figure 11.4 shows the weighting function for different w factors.

Table 11.1 shows the percentage of the neighborhood taken into account for varying w values. Small percentages take only a few very close neighbors into account, whereas larger values also consider further away neighbors.

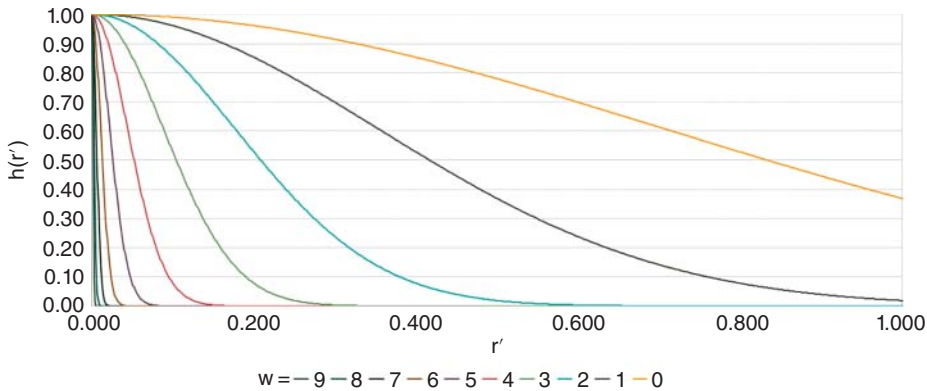


Figure 11.4 The weighting function for the normalized rank r' for different w values.

Table 11.1 Percentage of the neighborhood for varying w values.

w	9	8	7	6	5	4	3	2	1	0
%	0.2	0.4	0.7	1.4	2.8	5.6	11.1	22.2	44.2	74.8

To compare a projected mapping to a random mapping we need the root mean square distance $\bar{\delta}$ between all vectors in the high-dimensional space:

$$\bar{\delta} = \left(\frac{1}{n^2} \sum_{k=1}^n \sum_{l=1}^n \|X_k - X_l\|_2^2 \right)^{\frac{1}{2}} \quad (11.14)$$

For a random or a very bad mapping the value of $D_{h_{2D}}(P, X)$ will be close to $\bar{\delta}$. We compute the ratio of the differences between $\bar{\delta}$ and the weighted root mean square distance in the projected space $D_{h_{2D}}(P, X)$ and the original high-dimensional space $D_h(X)$, respectively. We define this ratio as the projection quality $Q(P)$:

$$Q(P) = \max \left(\frac{\bar{\delta} - D_{h_{2D}}(P, X)}{\bar{\delta} - D_h(X)}, 0 \right) \quad 0 \leq Q(P) \leq 1 \quad (11.15)$$

For a good projection preserving most of the local neighborhood relations the value of $Q(P)$ will be close to 1. For bad or random projections, the value will be close to 0. Theoretically, for a very bad projection $Q(P)$ could become a very small negative number, therefore we limit the lower value of $Q(P)$ to 0. This projection quality measure can be used to evaluate any type of image sorting scheme, as it only takes into account the weighted average neighborhood distances in the high-dimensional space and in the projected 2D space.

11.5 2D Sorting Results

11.5.1 Image Test Sets

For comparing conventional image sorting algorithms with our new sorting schemes, we used two image test sets with varying numbers of images and different feature vector dimensionalities. Table 11.2 gives an overview of the test image sets.

Table 11.2 Test image sets.

Images	Number of images	Feature vector	Dim
Stock photo flower images	1024	Low-level	50
$16 \times 16 \times 16$ RGB cube colors	4096	Lab color values	3

The first test set consists of 1024 stock photo images tagged with the keyword “flower”. This set was chosen because it allows a visual inspection of the sorting result even in the printed version. Figures 11.1–11.3 show this image set. The low-level feature vectors for representing the images are composed of a fuzzy color histogram with 20 bins, a color layout descriptor similar to MPEG-7 with 15 coefficients, an edge histogram with 6 bins, and a color texture histogram with 9 bins. This results in a total of 50 dimensions.

The second test set is somewhat artificial as we used images of constant color. The RGB color cube was equally sampled to get $16 \times 16 \times 16 = 4096$ different colors. The RGB values were transformed to the lab color space and then taken as feature vectors. In this case the task was to project a densely populated 3D space to the 2D grid of 64×64 map places. This is a task which is difficult for dimension reduction schemes as the vectors are not lying on lower-dimensional manifolds.

11.5.2 Experiments

In our experiments we compare our new algorithms with a classical SOM and a SSM implementation according to [13]. We optimized the classical SOM implementation in the way that the product of neighborhood function and the adapted step sizes and were precomputed and stored in a lookup table.

We compare three of our implementations: the described hierarchical SOM (HSOM), the improved self-swapping map (SSM2), and an identical version for which an additional cleanup process as described in section 11.3.2 was applied afterwards to correct isolated errors (SSM2 CL).

As for visual exploration schemes for very large image collections a continuous map of images is more useful from the users’ perspective, all experiments used toroidal maps where opposite edges of the map are connected. The number of map places was identical to the number of images to be sorted.

Figure 11.5 shows the results for sorting the flower image set. SSM outperforms the classical SOM considering only nearby neighbors but does not have smooth transitions

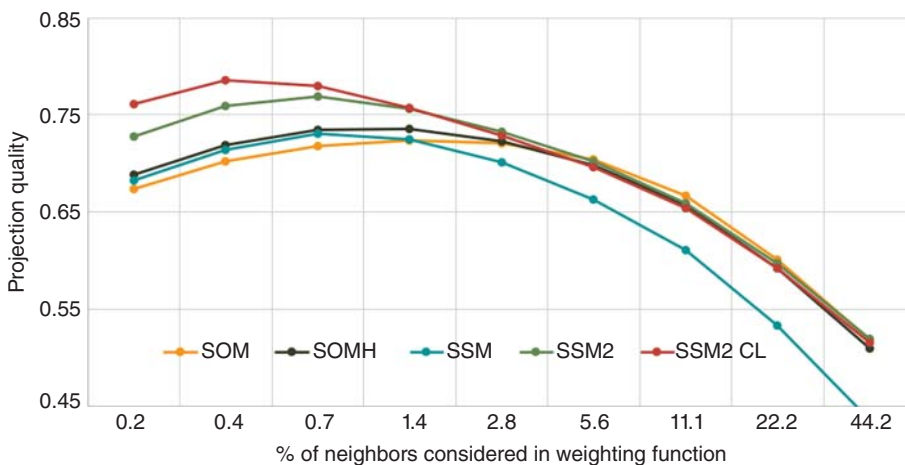


Figure 11.5 Projection quality for the 1024 flower images for different weighting functions (different percentages of closest neighbors).

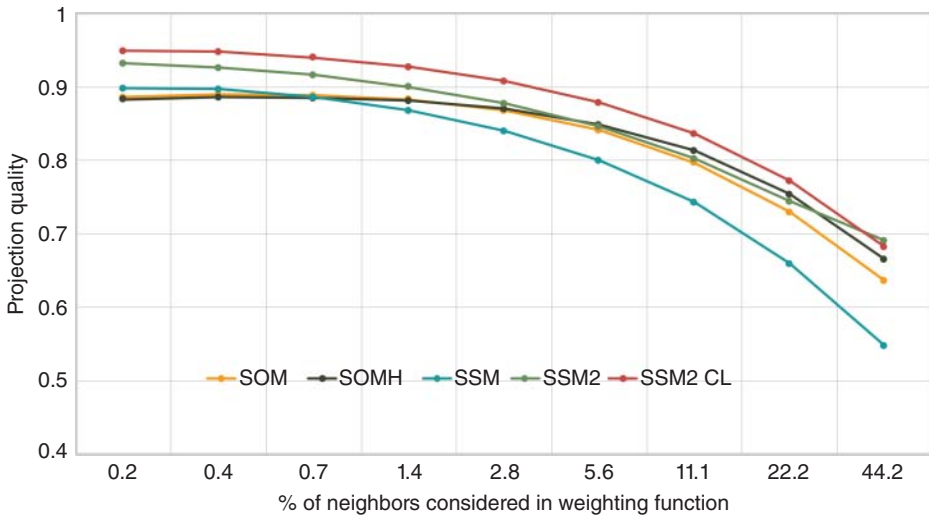


Figure 11.6 Projection quality for 4096 RGB color images for different weighting functions (different percentages of closest neighbors).

for neighbors that are further away. The SOMH is similar to the classical SOM but with better local quality at much faster processing time (see Figure 11.8). Combining SSM and SOM (SSM2) outperforms the other schemes, especially for closer neighbors. It can be seen that adding the cleanup path (SSM2 CL) improves the quality of nearby neighbors even further.

For the RGB color set the quality properties of the difference sorting algorithms are similar to the previously shown flower image set. In all cases the new proposed sorting schemes (SSM2 and SSM2 CL) are better in the sense of the projection quality $Q(P)$. Figure 11.6 shows the projection quality of the different sorting schemes. Figure 11.7 compares the sorting results for the 4096 RGB colors. It can be seen that the classical SOM has a good global arrangement but suffers from isolated bad mappings, the SSM generates small local artefacts. Both problems have been solved by our new approach (SSM2), resulting in the smoothest arrangement. Figure 11.8 compares the sorting time



Figure 11.7 Sorting results for the hierarchical SOM, original SSM, and the modified SSM2 algorithm for 4096 RGB color images.

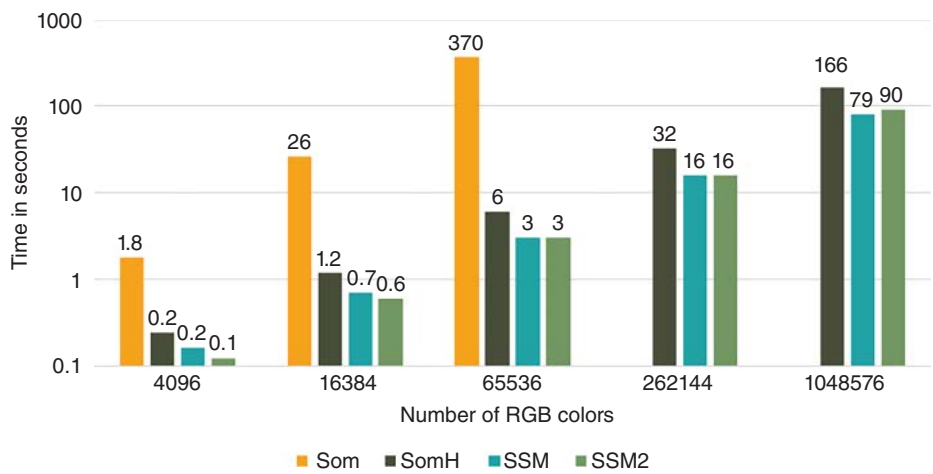


Figure 11.8 The running times of the implementations on Lab color vector datasets of varying size. One million items (3D color vectors) can be sorted in 90 seconds.

of different image set sizes (in the case of RGB colors). One million RGB colors can be sorted in 90 seconds (single-threaded 2.6 GHz Intel processor). The classical SOM needs many hours for the same task.

In summary, we have proposed a new scheme for visually sorting large sets of images. The new proposed modification of the SSM allows images to be sorted very quickly with better projection quality compared to other algorithms.

11.6 Demo System for Navigating 2D Image Maps

A demonstrator of the previously described approach using more than 1 million stock photos from Pixabay can be found at www.picsbuffet.com (Figure 11.9). The images were sorted using the SSM2. For a level of detail mechanism several hierarchical levels on top of the sorted map were created using the generalized median for each 2×2 image set from the level below. The generalized median of a set of images is the image which has the smallest sum of distances to all the other images in the set.

The user can explore the image pyramid (Figure 11.10) with the help of a web application and an interface similar to the mapping service Google Maps. Only a few hundred sorted images are shown at a time. By dragging the map other parts of the pyramid can be explored. Zooming in with the mouse provides more similar images on the next lower level, while zooming out to a higher level will give an overview of related image concepts.

As all images from Pixabay are annotated, keyword-based searches are possible. After entering a keyword query the application presents a heat map in the upper left corner and highlights the five regions of interest with little thumbnails. Clicking the heat map or the thumbnails below will change the viewport accordingly. Interesting regions can be shared with other users by sending the current URL of the website. This feature is also useful for bookmarking.

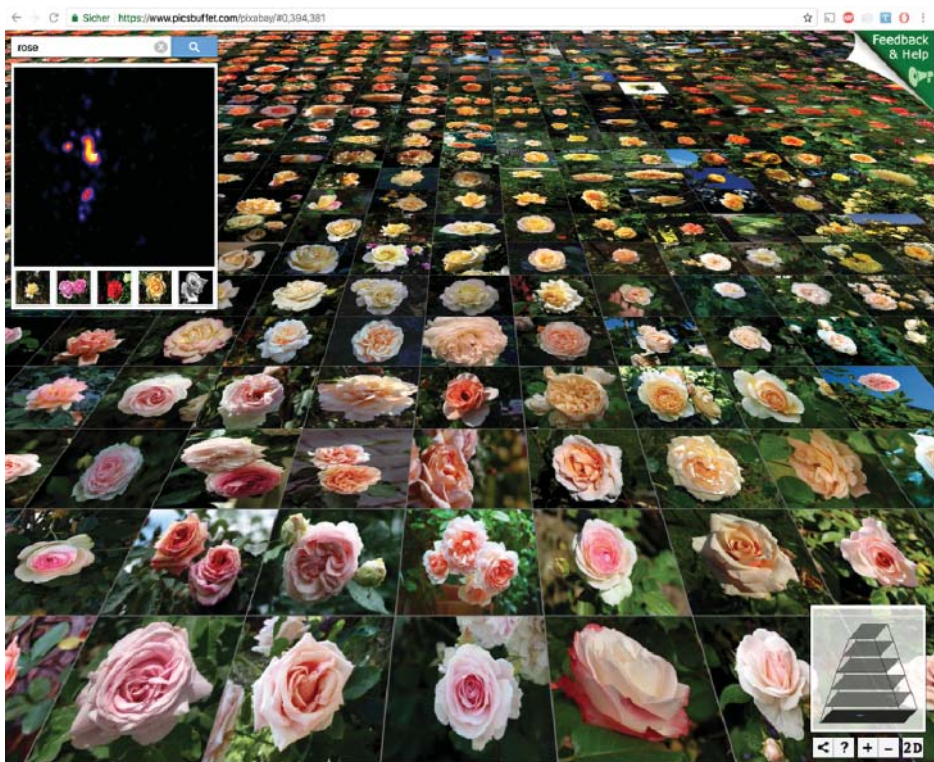


Figure 11.9 www.picsbuffet.com allows users to visually browse all images from Pixabay.

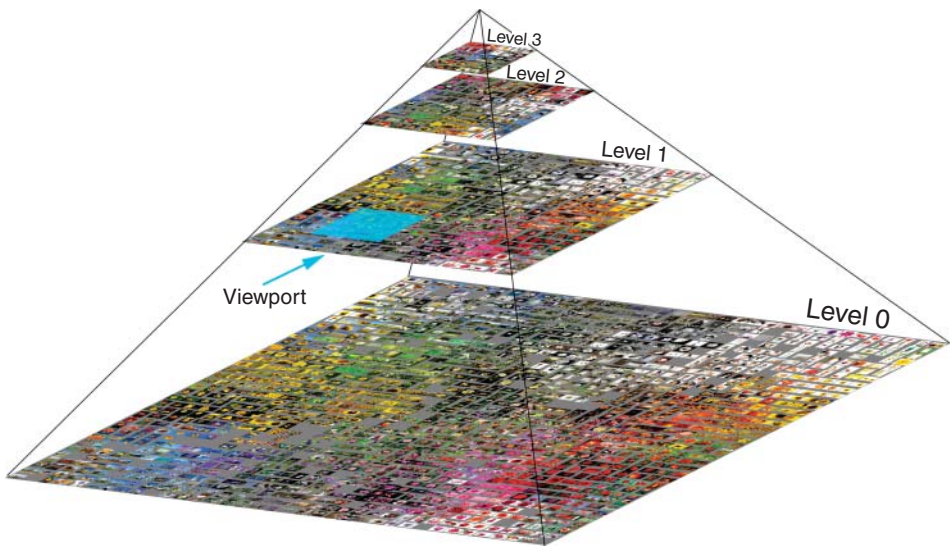


Figure 11.10 Pyramid structure of the picsbuffet image browsing scheme.

11.7 Graph-Based Image Browsing

For image collections that change over time an approach like *picsbuffet* cannot be used. At the Video Browser Showdown 2016 [16] we showcased an interactive video browsing system [17]. No keywords were available, only automatically generated image features and visual exploration techniques were allowed. Our graph-based video browser obtained the best results in this competition. We show how this idea can be extended for exploring very large archives of (untagged) images or video keyframes.

Graph-based approaches [18] are able to better preserve image relationships and can handle changes in the image archive easier. However, it is not clear how to visualize the “high-dimensional” graph in such a way that it can easily be perceived and navigated by the user. The abandoned Google Image Swirl [19] was an example of a graph-based image browsing tool. Image navigation was rather difficult and only a fraction of the available screen space was used. In this chapter we propose a graph-based image browsing approach that does not suffer from these problems. Subsets of images are successively extracted from the graph and regularly arranged onto a 2D surface while preserving the order of the previously displayed images. This creates an “endless” map and reduces the number of dimensions of the graph the user has to deal with, but still allows the complex inter-image relationships to be preserved. Visual exploration or visual browsing of a graph-based image collection requires three associated problems to be solved:

- 1) **Semantic image features:** High-quality image features are crucial to model inter-image similarities. The calculated similarities from these features need to be close to the similarities the user perceives. In addition, the size of the features should be very compact.
- 2) **A hierarchical image graph needs to be constructed.** Similar images should be connected and other images should be accessible by traversing the graph. Lower layers of the graph should connect very similar images, while higher layers should connect less related images.
- 3) **Graph visualization and navigation:** The images should be displayed such that they can easily be recognized. Navigation to other images should be easy and understandable.

11.7.1 Generating Semantic Image Features

Convolutional neural networks (CNNs) are not only able to achieve high accuracy rates for image classification tasks, the activations of a hidden layer can also serve as good feature vectors for image retrieval [20]. The outputs of a neural network trained for a particular set of categories do not perform well in general image retrieval tasks. Early layers describe primitive visual features such as colors or patterns, while later layers represent semantic information [21]. Intermediate layers can be seen as abstract representations of the visual content and are therefore less dependent on the categories of the trained network. Existing trained networks can be reused for feature extraction. We performed an extensive evaluation to find out which CNNs, which layers, and what kind of transformations are suited to generate feature vectors that perform well for image retrieval tasks. As an exhaustive discussion would be beyond the scope and topic of this chapter, we only will briefly describe the results of our investigation.

In order to determine how to generate good feature vectors, we tested different state-of-the-art CNNs [22–25] regarding their image retrieval quality. The models were chosen by their classification accuracy and their diversity. The networks were trained using the ILSVRC2012 data set [26] or the entire ImageNet set. To evaluate the retrieval quality, we used a test set with 1000 images and 100 categories. Neither the images nor the categories were part of the training image sets. Best results in regard to the mean average precision (MAP) were obtained using a deep residual learning network [23]. A pre-trained ResNet-200 was provided by Wei Wu via GitHub [27]. It was trained with the full ImageNet dataset, except for classes with fewer than 500 images. The 2048 activations before the fully connected layer were taken as initial feature vectors. By applying an L1 normalization followed by a PCA we could improve the retrieval quality and compress the feature vectors to only 64 dimensions stored as byte values. The best metric for comparing these feature vectors was the L1 distance.

11.7.2 Building the Image Graph

The image graph has to be constructed such that similar images are connected and other related images can be reached by navigating the edges of the graph. For a non-hierarchical image graph the following requirements are to be met for each image:

- (a) Connected images should be very similar.
- (b) The number of connections should not be too high or low.
- (c) The path to any other images should be as short as possible. This implies that the graph needs to be connected.

The definition of a general optimization rule for building the graph is rather difficult and it is not clear how these requirements should be weighted. In [17] we have presented an algorithm to construct a hierarchical image graph. To fulfill requirement (a) we used an edge swapping scheme. For requirement (b) we decided to keep the number of connections constant and to connect each image with four other images. Requirement (c) was simplified to only check the connectivity of the entire graph. The image graph is built in three steps. First, all images are sorted using a torus-shaped 2D SOM. The positions of the sorted images serve as the initial non-hierarchical graph: every image is connected with its four adjacent neighbors (Figure 11.11). In the next step this graph is improved by swapping edges. Two non-touching edges are swapped if this decreases the sum of the edge distances (11.16). Each swap improves the overall quality of the image graph. Swapping is stopped when the improvement gets too small:

$$\text{swap if } \text{dist}_{AX} + \text{dist}_{EY} < \text{dist}_{AE} + \text{dist}_{XY} \quad (11.16)$$

The final step consists of generating hierarchical versions of the previously constructed non-hierarchical graph. The graph of the next hierarchy level is initialized as a copy of the actual graph. By successively removing images that are very similar to their connected neighbors a reduced graph is obtained (Figure 11.12). If image A is removed, the images B, C, D, and E are reconnected with two new edges depending on the minimum of $\text{dist}_{BC} + \text{dist}_{DE}$, $\text{dist}_{BD} + \text{dist}_{CE}$, and $\text{dist}_{BE} + \text{dist}_{CD}$. For each hierarchy level three quarters of the images are removed as this corresponds to a resolution reduction by a factor of two on a 2D map.

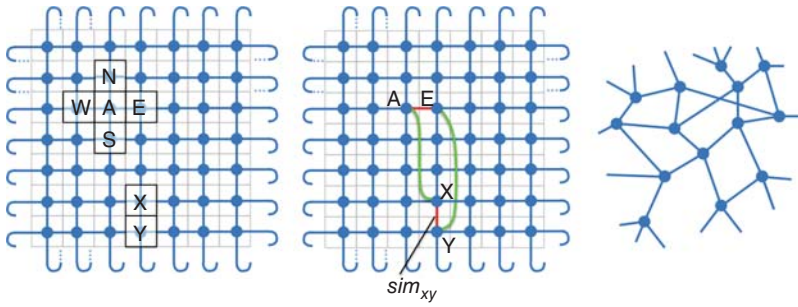


Figure 11.11 Left: initial image graph based on the image position of the 2D sorting. Right: edges are swapped if this decreases the total distance.

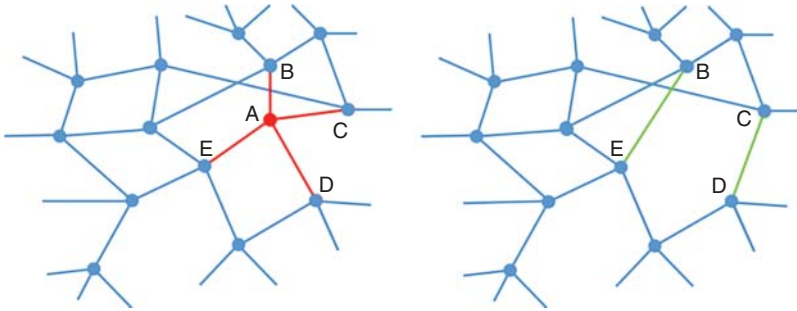


Figure 11.12 Building the graph of the next hierarchy level.

Building the image graph requires the distances between image pairs to be calculated. If the number of images is too high, only a sparse distance lookup table containing a set of the most similar images can be used. A dense lookup table would be too large due to the quadratic storage requirements. However, as our image features are rather small, the L1 distances between image feature vectors can also be computed on the fly. The memory requirements for storing the graph are linear with respect to the number of images.

Compared to the original graph-building process described above we have improved the algorithm in several aspects. Random edge swapping generates good image graphs, however the process is rather slow and the quality of the graphs varies depending on the initial sorting of the SOM and the order in which edge swaps were tested. This signifies that the optimization process easily gets stuck in local minima. One way to improve the final graph quality is to allow temporal degradations by modifying Eqn (11.16) to

$$\begin{aligned} dist_{AX} + dist_{EY} - dist_{AE} - dist_{XY} &< -\theta_t \\ \text{with } \theta_{t+1} &= \beta\theta_t \text{ and } \beta < 1 \end{aligned} \quad (11.17)$$

The initial value of the swapping threshold θ_0 is set to the average distance value. With each iteration the value of θ_t is gradually decreased towards zero by a factor β less than one. This results in a process like simulated annealing, and leads to image graphs with better quality, but the choice of the optimal factor β is not obvious. However, the main remaining problem is the slow convergence of this approach. The reason for this is the low probability of finding two edges that improve the graph quality if they are swapped.

Algorithm 11.1 Improved edge swapping

```

1  Remove a random edgekl (connecting imagek and imagel)
2  gain = distkl
3  for i = 1 to maxTries do
4      Connect imagel with imagem, which is a random image
        from the list of the most similar images of imagel
5      gain = gain - distlm
6      From imagem remove the edge with the highest distance
        (let this be edgemn and imagen)
7      gain = gain + distmn
8      if gain - distkn > 0 then
9          Connect imagen with imagek, return // improvement
10     else
11         imagel = imagen
12     end
13 end
14 Undo all changes, return // no improvement was found

```

Despite the quadratic time complexity for building the graph, we have developed an algorithm that speeds up the process significantly. Instead of trying to find pairs of edges to be swapped, we are looking for a sequence of reconnections that in sum improves the graph quality. This can be archived by removing a series of bad edges and replacing them with potential better edges. The probability of finding such sequence is much higher compared to the previous approach. Algorithm 1 shows a pseudo code implementation.

The algorithm requires a list of the most similar images for every image. For computation reasons it is favorable to limit the number of this list. In our experiments we used a sparse lookup table containing the 1000 closest neighbors per image. A step-by-step example of the algorithm is shown in Figure 11.13.

Figure 11.14 shows a comparison of the convergence of the previous and the improved edge swapping algorithm. Presorting the images with a SOM can help to speed up the graph-building process. A further speedup can be achieved by tracking the distances

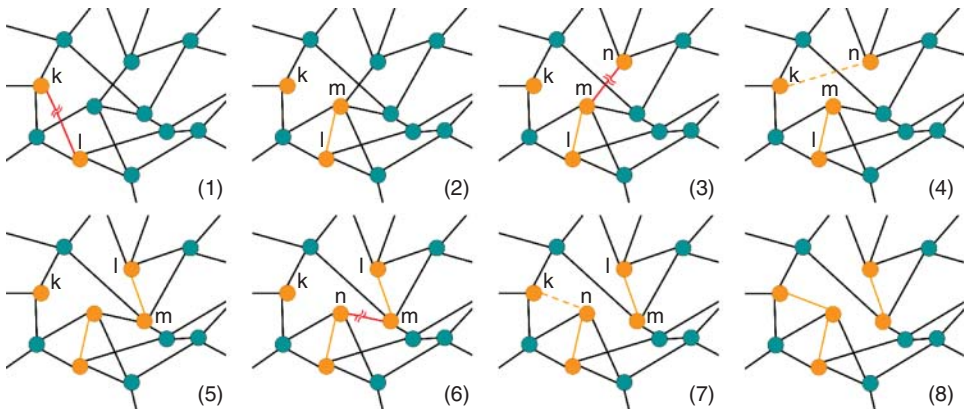


Figure 11.13 Schematic sequence of the improved edge swapping algorithm.

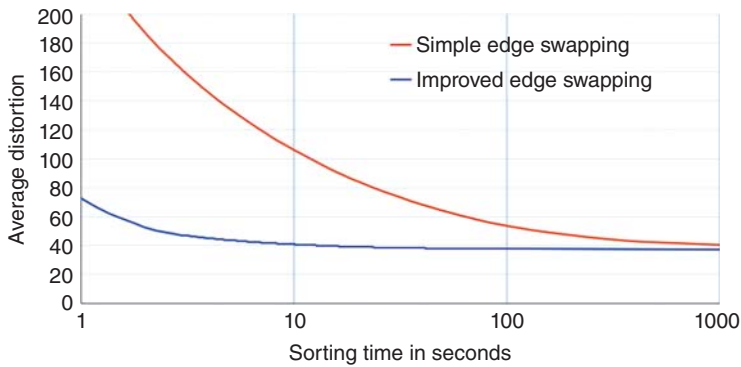


Figure 11.14 Quality over time for building a graph with 10000 RGB colors, improved edge swapping converges significantly faster.

of the connected images. Edges with higher distances than the average edge distance are chosen to be changed with higher probability. Applying these improvements allows a graph of one million images to be built up in 1 h on a single core using an Intel Xeon E3-1230 v3 CPU. Using the simple edge swapping approach, it takes over 4 days to reach similar quality.

11.7.3 Visualizing and Navigating the Graph

As described before, a 2D projection of the entire graph cannot preserve the complex image relationships and will result in similarity discontinuities. Our approach is to combine the graph navigation with a 2D projection. By dynamically querying the image graph, projecting and showing only a subgraph, it is possible to preserve the complex inter-image relationships for display and navigation.

Starting from an image in focus, its connected edges are recursively followed until the desired number of neighboring images has been retrieved. In a next step these images are sorted with the improved SSM, which maps the images according to their similarities onto a regular 2D image map. If the user does not find the desired image, an interaction may be necessary to navigate to other parts of the graph. This is achieved by dragging the 2D map such that the region where the searched images are anticipated becomes visible. Dragging the map moves some images out of the viewport and leaves empty space on the opposite side. The inverse direction of the mouse drag indicates the images of interest (Figure 11.15) and instructs the system to retrieve their neighboring vertices from the graph. Images that have already been displayed are ignored. The newly retrieved images are mapped to the most suitable empty places, defined by the highest surrounding image similarities. This sorting again is performed using the modified SSM, however the positions of the previously displayed images remain unchanged. To emphasize realistic navigation experience previous arrangements are cached.

The user may change the zoom factor to get an overview or to see more similar images. In this case the closest subgraph of the next hierarchy level needs to be selected and displayed. In order for the user not to get lost a smooth transition from the old map to the new map is performed.

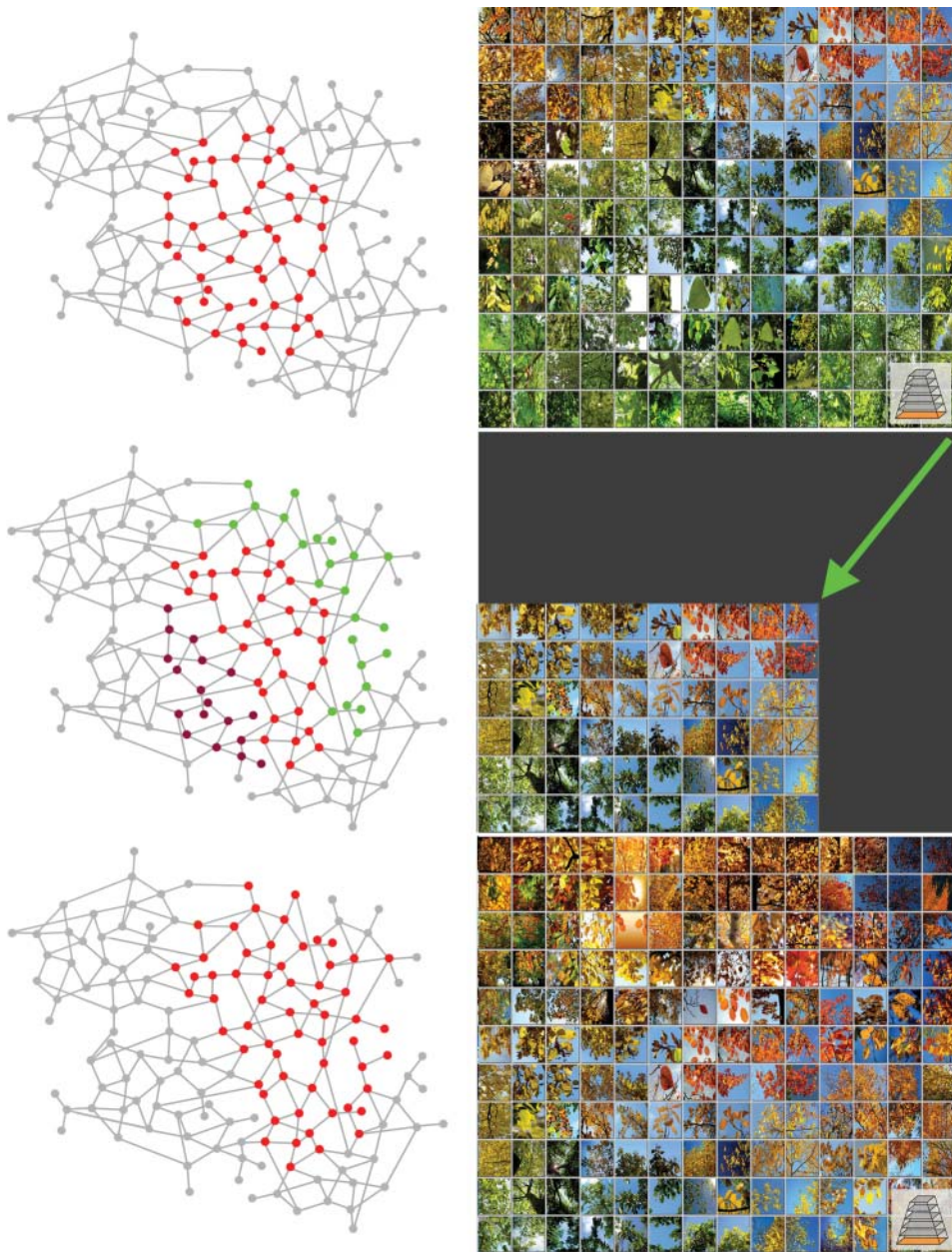


Figure 11.15 Navigating the graph. Dragging the map (indicated by the arrow) will change the subgraph (top) to be displayed. Previously displayed images that remain in the view port keep their position. Newly retrieved images are visually sorted and added to the map.

11.7.4 Prototype for Image Graph Navigation

We have implemented a prototype of the new image browsing system. The user interface and experience are very similar to our picsbuffet image browser, but there is a much less abrupt change of thematic concepts. To find a particular image a user would typically start the navigation on a higher level and then zoom to lower levels when a region of interest has been found. In addition (if keywords are available) the user may start with a keyword search and then explore similar and related images and concepts. The user always sees a canvas which is densely filled with images. Dragging the canvas will load new images which are sorted and added to the new empty spaces. Figure 11.16 shows two examples of how a successive image navigation constructs a growing image map.

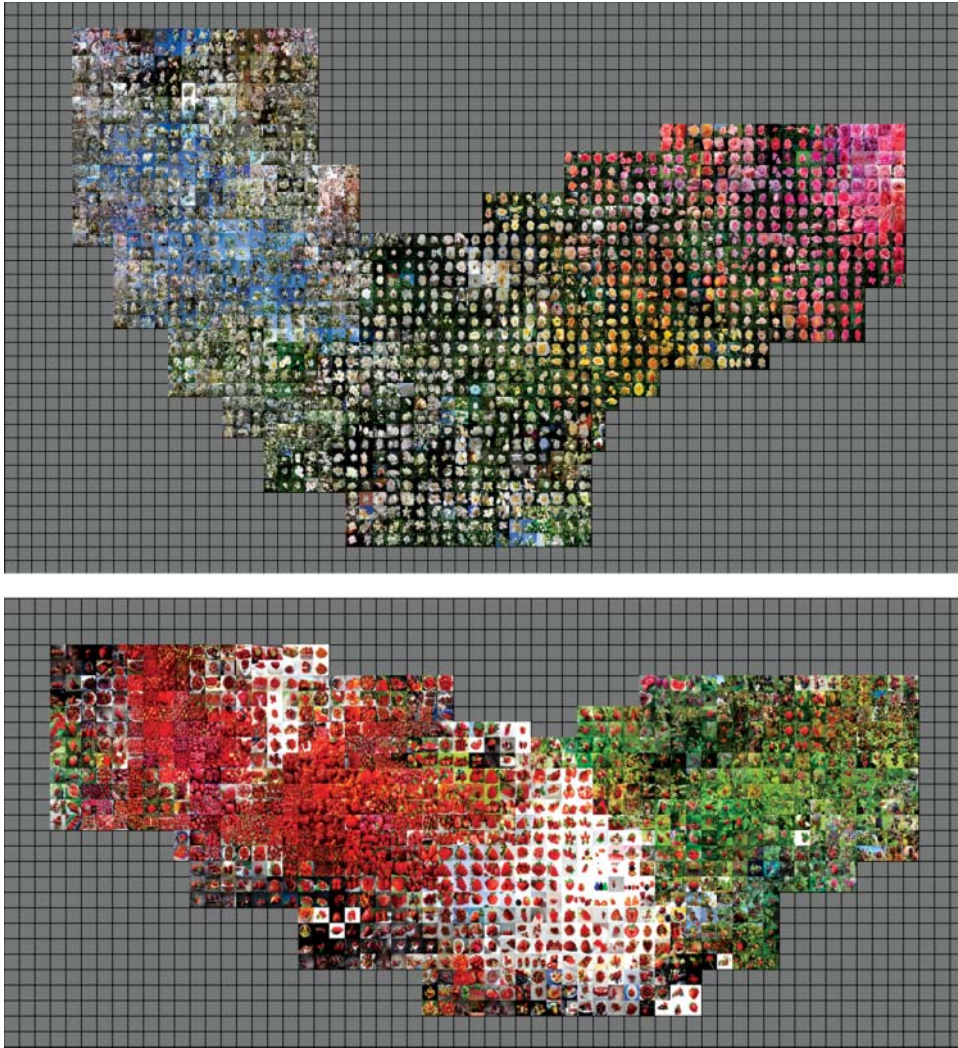


Figure 11.16 Examples of successive views of a user's visual exploration looking for flower (top) and strawberry images (below).

As the entire image graph is stored, the adaptation to a user interaction can be performed very quickly. The retrieval of the connected images and their sorting can be done very efficiently in fractions of a second.

If a new image is added to the graph, its feature vector has to be calculated, then the four most similar images will be connected with the new image. This will change the number of edges, but the visualization process can cope with varying numbers of connections per image. The same is true for removing images. For an image to be removed, all its connections have to be removed as well. If the image collection has undergone substantial changes, it might be useful to reorganize the entire image graph in order to keep the number of connections per image constant.

11.8 Conclusion and Future Work

We have shown how to speed up image sorting and how the resulting sorting quality can be improved. In addition, we have introduced a new measure to determine the projection quality of sorted image sets. While the new quality measure seems to match the perceived quality, an extensive user study will be performed to verify the claim of the improved projection quality measure.

In order to be able to deal with changing image sets, we have presented a new graph-based approach to visually explore large collections of untagged images. Organizing all images with a hierarchical graph helps to preserve inter-image relations much better than static 2D image sorting. As subgraphs are projected onto a 2D map the user can navigate the image collection similar to static interactive maps and will not be aware of the underlying graph structure. In opposition to static image maps the adaptation to changes of the image collection can be done instantly.

Future work will focus on further speeding up the graph-building process by using fast presorting techniques and better determining possible edge swap candidates. Another problem that needs to be solved is the caching of previously displayed images. If a user performs a long circular navigation path the subject of the images may have changed. In this case the previously shown image should not be shown again.

As a further development we will enable visual image exploration using keywords even if the images are not tagged. For collections with untagged images, feature vectors of typical keywords are to be determined, which can be used as the search query.

An implementation of this graph-based image exploration tool and other demonstrations is available at www.visual-computing.com.

References

- 1 Ferreira de Oliveira, M.C. and Levkowitz, H. (2003). From visual data exploration to visual data mining: A survey. *IEEE Transactions on Visualization and Computer Graphics* 9: 378–394.
- 2 Pearson, K. (1901). On lines and planes of closest fit to points in space. *Philosophical Magazine* 2: 559–572.
- 3 Sammon, J. (1969). A nonlinear mapping for data structure analysis. *IEEE Transactions on Computers* 18: 401–409.

- 4 Tenenbaum, J.B., Silva, V., and Langford, J.C. (2000). A global geometric framework for nonlinear dimensionality reduction. *Science* 290: 2319–2323.
- 5 Roweis, S.T. and Saul, L.K. (2000). Nonlinear dimensionality reduction by locally linear embedding. *Science* 290: 2323–2326.
- 6 van der Maaten, L. and Hinton, G.E. (2008). Visualizing high-dimensional data using t-SNE. *Journal of Machine Learning Research* 9: 2579–2605.
- 7 Fan, J., Gao, Y., Luo, H. et al. (2008). A novel approach to enable semantic and visual image summarization for exploratory image search. *Multimedia Information Retrieval* 1: 358–365.
- 8 Heesch, D. (2008). A survey of browsing models for content based image retrieval. *Multimedia Tools and Applications* 40: 261–284.
- 9 Qiu, G., Morris, J., and Fan, X. (2007). Visual guided navigation for image retrieval. *Pattern Recognition* 40: 1711–1721.
- 10 Strong, G., Hoque, E., Gong, M., and Hoeber, O. (2010). Organizing and browsing image search results based on conceptual and visual similarities. In: *Advances in Visual Computing*, vol. 2, 481–490.
- 11 Wang, J., Jia, L., and Hua, X.-S. (2011). Interactive browsing via diversified visual summarization for image search results. *Multimedia Systems* 17 (5): 379–391.
- 12 Kohonen, T. (1982). Self-organized formation of topologically correct feature maps. *Biological Cybernetics* 43: 59–69.
- 13 Strong, G. and Gong, M. (2014). Self-sorting map: an efficient algorithm for presenting multimedia data in structured layouts. *IEEE Transactions on Multimedia* 16 (4): 1045–1058.
- 14 Quadrianto, N., Smola, A.J., Song, L., and Tuytelaars, T. (2010). Kernelized sorting. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32 (10): 1809–1821.
- 15 Kovesi, P. (2010). Fast almost-Gaussian filtering. *Digital Image Computing: Techniques and Applications* 121–125.
- 16 Schoeffmann, K., Ahlstrom, D., Bailer, W., and Cobarzan, C. (2013). The video browser showdown: a live evaluation of interactive video search tools. *International Journal of Multimedia Information Retrieval* 1–15.
- 17 Barthel, K.U., Hezel, N., and Mackowiak, R. (2016). Navigating a graph of scenes for exploring large video collections. *MultiMedia Modeling* 22: 418–423.
- 18 Qiu, S., Wang, X., and Tang, X. (2013). Visual semantic complex network for web images. in *IEEE International Conference on Computer Vision* 3623–3630.
- 19 Jing, Y., Rowley, H., Wang, J. et al. (2012). Google Image Swirl: A large-scale content-based image visualization system. in *International Conference on World Wide Web* 21: 539–540.
- 20 Krizhevsky, A., Sutskever, I., and Hinton, G.E. (2012). ImageNet classification with deep convolutional neural networks. in *International Conference on Neural Information Processing Systems* 25: 1097–1105.
- 21 Razavian, A.S., Azizpour, H., Sullivan, J. and Carlsson, S. (2014) CNN features off-the-shelf: An astounding baseline for recognition, in *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 512–519.
- 22 He, K., Zhang, X., Ren, S. and Sun, J. (2016) Deep residual learning for image recognition, in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778.

- 23 He, K., Zhang, X., Ren, S. and Sun, J. (2016) Identity mappings in deep residual networks, in *European Conference on Computer Vision, Volume 14*, pp. 630–645.
- 24 Ioffe, S. and Szegedy, C. (2015) Batch normalization: Accelerating deep network training by reducing internal covariate shift, in *International Conference on Machine Learning*, Volume 32, pp. 448–456.
- 25 Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S.E., Anguelov, D., Erhan, D., Vanhoucke, V. and Rabinovich, A. (2015) Going deeper with convolutions, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- 26 Russakovsky, O., Deng, J., Su, H. et al. (2015). ImageNet large scale visual recognition challenge. *International Journal of Computer Vision* 115 (3): 211–252.
- 27 Wei, W. GitHub ResNet [Online]. Available: <https://github.com/tornadomeet/ResNet> (accessed 10 8 2017).

12

Medical Decision Support Using Increasingly Large Multimodal Data Sets

Henning Müller and Devrim Ünay

Medical decision support has traditionally been using model-based (or handcrafted) approaches and small data sets for evaluation. As medical data are sometimes hard to obtain and annotations by clinicians required for learning-based approaches are very expensive they are often acquired in small amounts. Applications for such decision support include detection, classification, and retrieval in several clinical scenarios. Each of these applications provides or modifies information that is essential in decision making.

This chapter analyses recent trends and techniques that make use of increasingly large data sets and thus more data-driven approaches to medical decision support that have in some areas replaced the more traditional rule-based (or model-based) approaches. Besides techniques and available data sets, the text describes scientific challenges that have helped to make data and annotations available that really are the basis for advances in the field. Challenge infrastructures and approaches are explained as they are often essential to access data, and application scenarios give examples of existing applications and objectives. An outlook on ideas on how to overcome current limitations and how to tackle the upcoming challenges of image-based decision support for digital medicine ends the chapter.

12.1 Introduction

Digital medicine has become a reality over the past few years and hospitals in basically all Western countries have been using electronic medical records for several years now, making all the clinical data available in digital form [1]. This has created a major revolution in medicine, where experience is increasingly complemented by evidence based on data that are often complex to integrate manually [2, 3]. Other authors highlight the cost savings that are possible when using big data in medicine [4]. In addition to reducing costs, big data initiatives in health care can save lives and improve patient outcomes [5]. Particularly personalized or precision medicine really requires much data on individuals for optimizing treatment outcomes based on past results [6]. It is specifically stated in [7] that the prospect of applying precision medicine is dramatically improved by the recent development of large-scale biomedical databases (such as the human genome

data), powerful methods for characterizing patients (such as proteomics, metabolomics, genomics, and cellular assays), and computational tools for analyzing large sets of data. In order to fulfill this great potential, big data analytics has to overcome some challenges like variations in the data, privacy constraints, security, and accessibility [8]. An overview of big data for health is given in [9].

Much has been written on big data analysis in the medical field and medical imaging is estimated to occupy up to 30% of world storage [10]. Storage of genomic data has also increased exponentially in recent years and in cancer care many hospitals now have full genome sequencing for all patients, leading to massive amounts of data that become available for data analysis. The amount of data produced in modern imaging protocols (which can contain tens of thousands of images per patient or extremely large images, such as in histopathology) and of genomic data means that a fully manual analysis has sometimes simply become impossible. Hence, these huge heterogeneous data need to be efficiently processed (stored, distributed, indexed, and analyzed) by big data analytics to improve medical decision support [11, 12] states that there is much potential in delivering more targeted, wide-reaching and cost-efficient health care by exploiting current big data trends and technologies.

In general, the electronic health record has as an objective to bring the right information to the right people in the right moment and the right format [1]. With massive amounts of data to be interpreted it also means that the right tools are required in this context to make meaningful use of the supplied data and integrated interactions between the data. Historically, medical decision support has been rule-driven and decision trees have been developed based on medical knowledge for many years, where rules are derived from the data manually. In recent years data-driven approaches have aimed

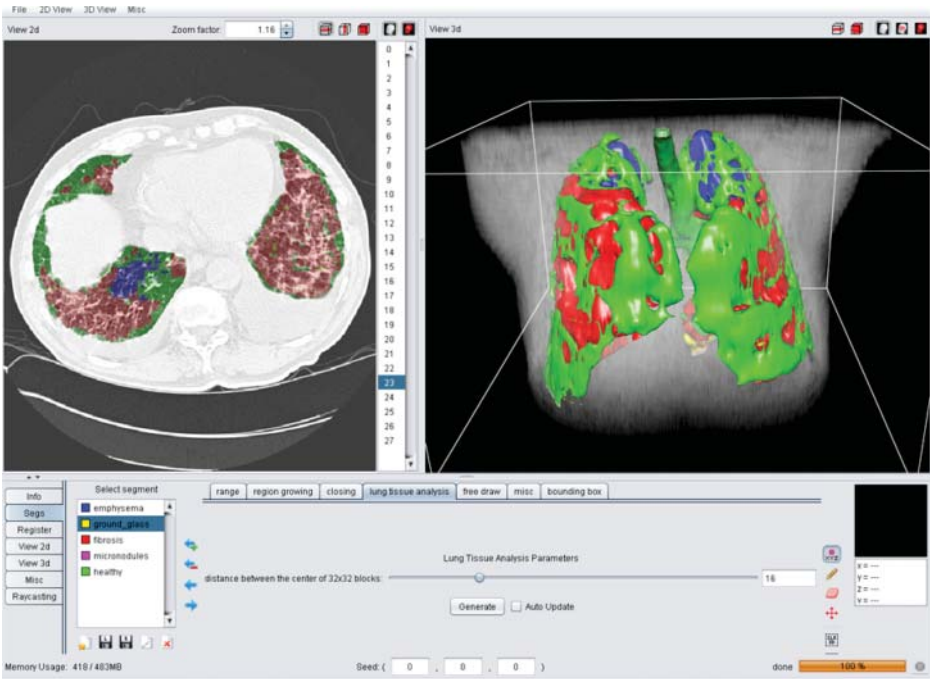


Figure 12.1 Screen shot of a tool for 3D lung tissue categorization (details in [13]).

at complementing this in areas such as radiomics [14], where connections in the data are learned and then often used for prediction. For visual medical decision support three main approaches have been used besides simple quantitative measures: *detection*, *classification*, and *retrieval*. In detection the objective is to find specific regions in volumes, for example nodules in the lung [15] or specific lesions [16]. Classification has looked more into determining the specific type of an entire image or image region (region of interest). Figure 12.1 gives an example of the classification of lung tissue into a limited number of classes and its visualization. Many similar tools have been proposed and developed.

Medical image retrieval has been another active area of research with review articles describing the main approaches [17, 18]. The idea is that physicians can take images of a current case and clinical data, and use these to formulate queries to find similar cases or lesions [19, 20]. An example for a retrieval application can be seen in Figure 12.2, where queries can be formulated using text and images combined.

Whereas many early approaches relied on single sources of data, images, or text or structured data, it has become clear that multimodal data and fusion approaches can lead to much better results [22, 23]. An exemplary architecture of such a retrieval system can be seen in Figure 12.3. Most systems using visual data do not take into account any other data types. This is also shown by the relatively small number of papers found in this survey. The survey concentrates fully on multimodal approaches, thus only papers that include more than a single modality are included in the detailed analysis. It also concentrates on data-driven approaches and thus does not include rule-based approaches.



Figure 12.2 Screen shot of an image retrieval application that allows for text search and also image similarity search in medical data, taken from [21].

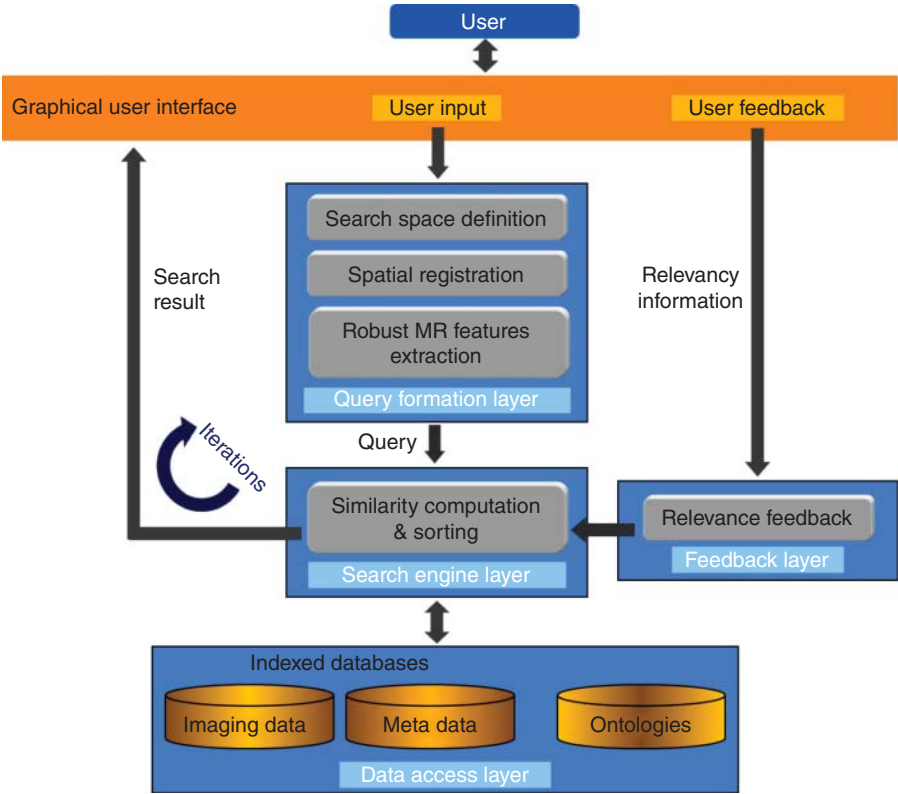


Figure 12.3 Architecture of an image retrieval system that exploits multimodal information for improved similarity search in medical data, taken from [24].

12.2 Methodology for Reviewing the Literature in this chapter

This chapter aims to review the literature of mainly the past five years (2011–2016) in the field of medical visual decision support and highlights the use of multimodal data and data-driven approaches, even though not all applications use multiple modalities at the moment. Several older papers are cited as well, mainly when we considered them important or landmark papers in the domain. We aimed to cite journal articles over conference papers when similar topics were available and to have a variety of authors for the various topics, even though it is possible that papers have been missed, as the field is extremely large and consists of many communities. We concentrate is on multimodal (visual + non-visual) data usage, which is surprisingly small, with most researchers focusing on single media, in our case only visual data without taking any other information on the patients. Many non-visual and visual-only areas are left out, as this was not considered a main objective of the text.

For the papers included in this survey we systematically searched through the Pubmed, IEEE, and Google Scholar databases and conferences such as the Medical

Image Computing and Computer Assisted Interventions (MICCAI), the IEEE International Symposium on Biomedical Imaging (ISBI), the International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), and the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) by using the search terms “multimodal medical decision support”, “large scale medical diagnosis”, “medical deep learning”, “big data analytics in healthcare”, “medical multimodal diagnosis”, “large scale multimodal medical retrieval”, and the like. The set of resulting references was then manually controlled and analyzed to keep only the articles corresponding to our criteria.

12.3 Data, Ground Truth, and Scientific Challenges

This section describes the main challenges in medical data access. Medical data have specific requirements, as they cannot simply be shared for research use but have to be under the control of ethics committees and strong rules regarding the data sharing [25]. This often results in relatively small data sets and data that can rarely be shared for research use, even though data sharing is regarded as a main driver for innovation [26]. Scientific challenges have managed to make large data sets available and this is described here in detail. To ease the analysis, the papers included are grouped by the size of the data sets they use as small ($<1\text{K}$ images/patients), medium ($1\text{K}–100\text{K}$), or large ($>100\text{K}$).

12.3.1 Data Annotation and Ground Truthing

Usually, imaging data without annotations is difficult to use for evaluating image analysis algorithms. Thus, annotation of images is an important aspect of many image-related research projects [27] to create a (sometimes subjective) ground truth. Machine learning algorithms most often require global image annotations or, even better, labeling of precise image regions. This is expensive and hard to obtain on a large scale, particularly in medicine where domain experts (physicians) are required, as the labeling is highly specific. Some efforts exist to make large annotated data sets available [28] and also to generate annotations on a large scale based on small manually annotated data [29]. Sometimes when enough globally annotated data are available then local regions with abnormalities can be derived as described in [30]. The type of annotations available also has a strong influence on the metrics that can be used for the evaluation and for comparing different approaches [31]. For evaluating detection, detailed regions are required and for segmentation evaluation, fully annotated structures. For global classification, simple global labels are sufficient.

12.3.2 Scientific Challenges and Evaluation as a Service

Historically, medical image analysis has been done on very small collections of private data, and it was thus impossible to compare any two algorithms as baselines can be chosen fairly arbitrarily [32]. Over the years, data sharing was often promoted but only rarely put into practice on a larger scale. One of the first medical evaluation campaigns

with a publicly shared image data set was the ImageCLEF medical task in 2004 [33], where several groups compared their algorithms on image retrieval. A medical task has been held almost every year since 2004 on retrieval or classification.

At the MICCAI conference the first challenge was on liver segmentation using a fairly modest data set [34] in 2007, with an on-site challenge and a limited time of a few hours for computation. This has now led to many more challenges that have had an increasingly important impact at other conferences [35, 36]. The web page on the Grand Challenges in Medical Image Analysis¹ aims to collect all active and past challenges in medical image analysis and these have increased over the years.

Scientific challenges have improved the comparability of algorithms and made medical data accessible for many researchers, which is an important impact. Scholarly and financial impact are also high [37, 38], but there are difficulties in distributing data when the data sets become extremely large (download impossible, sending hard disks not practical), when they are confidential (cannot be shared), and when they change quickly over time (requirement to always work on the latest data), as is the case with medical routine data [39]. Approaches that bring the algorithms to the data have thus been developed to solve most of these challenges. In a larger context this is also called evaluation as a service (EaaS) [40]. Different approaches have been implemented for this, including submissions of source code, virtual machines, and Docker containers.

12.3.3 Other Medical Data Resources Available

Funding organizations have realized over the past 20 years that a large part of the research budget and time in projects is often used for acquiring and cleaning the required data but many of the data sets are only exploited in a short project lifetime and by a very small number of people. This has led to data-sharing policies in funding organizations. For example, the National Institutes of Health (NIH) require funded projects to make the data available after the project end. This has led to several large archives of data that remain available for research use and are frequently employed for research, such as The Cancer Imaging Archive (TCIA)² and The Cancer Genome Atlas (TCGA)³, which include many images and other clinical data or documents. The data can then be used for several years even though sometimes imaging protocols change and clinical impact on current care may be limited with very old data. This allows many research groups to access image data sets and work on the data for research without the requirement to go through ethics approval and tedious data collection in collaboration with a clinical institution. Sharing the data acquisition can also lead to bigger and better annotated data sets if costs can be shared.

In the European Union similar ideas are underway to make sure that data created with public money can also be used by other researchers and have a maximum impact. Guidelines are made available to push for more open-access publishing and data sharing, not limited to medical data.⁴

1 <https://grand-challenge.org>.

2 <http://www.cancerimagingarchive.net>.

3 <https://cancergenome.nih.gov>.

4 http://ec.europa.eu/research/participants/docs/h2020-funding-guide/cross-cutting-issues/open-access-data-management/open-access_en.htm.

12.4 Techniques used for Multimodal Medical Decision Support

This section summarizes the techniques employed in the papers we found through a detailed literature search, with most being published from 2011 to 2016.

Table 12.1 presents an overview of the papers included in this survey, while Figure 12.4 provides a breakdown of these papers into year and data set size, anatomy, application, and imaging modality. This gives a good global overview of topics that were covered in the past five years and the tendencies for multimodal data usage.

12.4.1 Visual and Non-Visual Features Describing the Image Content

The literature on multimodal medical decision support consists of work exploiting various types of image features, such as intensity [43, 46, 49, 50, 56, 60, 64, 65, 69, 70], color/grey level histogram analysis [50, 52, 56, 60], texture [43, 51, 65, 69, 70], shape [24, 41, 50, 59, 60, 65, 69, 70], and volumetric information of objects of interest [42, 44–49, 64, 65, 70]. Local binary patterns [52], and the popular scale invariant feature transform (SIFT) [55, 57] have also been used.

With the recent popularity of deep learning, several articles made use of features based on the output of deep neural networks for multimodal medical decision support [53, 54, 61–63, 66–68].

Augmenting information mined from images with that gained from non-imaging data has been a popular track to improve medical decision support. To this end, researchers benefit from demographic indicators such as age and gender [24, 49, 58, 63–66, 68–70], medical history of subjects [24], clinical findings like biopsy results, cerebrospinal fluid (CSF) biomarkers, physical/blood measurements, tumor descriptors [24, 44–51, 58, 59, 63–65, 68], natural language processing based analysis of clinical/radiology reports [53, 54, 56, 61, 62, 67] and biomedical journal articles [52, 55], analysis of biosignals such as electro-encephalogram (EEG), electro-cardiogram (ECG), and audio recordings [41, 42, 60], and genetic test results [43, 48, 59].

12.4.2 General Machine Learning and Deep Learning

Machine learning is an indispensable part of medical decision support, especially in diagnostic and categorization applications. In order to mine multimodal information, the literature made use of various machine learning techniques such as ensemble learners [42, 50], support vector machines [44–46, 48, 51, 55, 63], naive Bayes classifiers [48, 51], nearest-neighbor classifiers [51, 56], decision trees [49], and random forests [65, 70]. Depending on the applications all these classifiers have their advantages and inconveniences. To combine the results of the different modalities, early fusion employs all features in a single space whereas late fusion combines classifier outcomes. More on fusion techniques between visual features and text can be found in [71].

Deep learning has had an important influence in computer vision over the past ten years [72]. The medical field has also had a strong increase in using neural networks for decision support. Several review articles have been published, such as [73] for health

Table 12.1 Overview of papers using multimodal data sets for medical decision support.

Ref.	Modality		Application	Anatomy	Data set	Learning
	Image	Non-image				
[41]	US	Bs	Cardiac decision support	Cardiac	Small	n.a.
[42]	MRI, PET	Bs	Alzheimer diagnosis	Brain	Small	Ensemble
[43]	Microscopy	Ge	Cancer diagnosis	Brain	Small	n.a.
[44]	MRI	CF	Alzheimer diagnosis	Brain	Small	SVM
[45]	MRI, PET	CF	Alzheimer diagnosis	Brain	Small	SVM
[46]	MRI, PET	CF	Alzheimer diagnosis	Brain	Small	SVM
[47]	MRI	CF	Alzheimer diagnosis	Brain	Small	n.a.
[48]	MRI	CF, Ge	Alzheimer diagnosis	Brain	Small	SVM, NB, LR
[49]	MRI	De, CF	Survival prediction	Brain	Small	DT
[24]	MRI	De, MH, CF	Dementia diagnosis	Brain	Small	n.a.
[50]	X-ray, CT	CF	Osteoporosis diagnosis	Bone	Small	Ensemble
[51]	US	CF	Disease staging	Liver	Small	SVM, NN, NB
[52]	Jpeg	Re	Case-based retrieval	Multiple	Large	n.a.
[53]	OCT	Re	Tissue segmentation	Retina	Small	CNN
[54]	CT, MRI	Re	Image annotation	Multiple	Large	CNN
[55]	Mixed	Re	Case-based retrieval	Multiple	Large	SVM
[55]	Mixed	Re	Modality classification	Multiple	Large	SVM
[56]	CT	Re	Case-based retrieval	Lung	Small	NN
[57]	CT	Re	Case-based retrieval	Lung	Small	n.a.
[57]	MRI, PET	Re	Case-based retrieval	Brain	Small	n.a.
[58]	MRI	De, CF	Alzheimer diagnosis	Brain	Small	n.a.
[59]	US	CF, Ge	Cancer diagnosis	Thyroid	Small	n.a.
[60]	US	Bs	Disease monitoring	Cardiac	n.a.	n.a.
[61]	X-ray	Re	Image annotation	Multiple	Medium	CNN, RNN
[62]	CT, MRI	Re	Image categorization	Multiple	Large	CNN
[63]	MRI	De, CF	Survival prediction	Brain	Small	SVM, CNN
[64]	MRI	De, CF	Survival prediction	Brain	Small	n.a.
[65]	MRI	De, CF	Survival prediction	Brain	Small	RF
[66]	X-ray	De	Pathology retrieval	Chest	Small	CNN
[67]	US	Re	Image annotation	Cardiac	Small	CNN
[68]	Cervicogram	De, CF	Cancer diagnosis	Uterus	Small	CNN
[69]	X-ray	De	Cancer diagnosis	Breast	Large	CNN
[70]	MRI	De	Survival prediction	Brain	Small	RF

Data sets grouped as small (<1K patients/images), medium (1K–100K), or large (>100K). MRI, magnetic resonance imaging; PET, positron emission tomography; US, ultrasound; CT, computed tomography; OCT, optical coherence tomography. Bs, biosignals; Ge, genetics; CF, clinical findings; De, demographics; MH, medical history; Re, reports; SVM, support vector machines; NB, naive bayes; LR, logistic regression; DT, decision tree; NN, nearest neighbor; RF, random forests; CNN, convolutional neural networks; RNN, recurrent neural networks; n.a., not available.

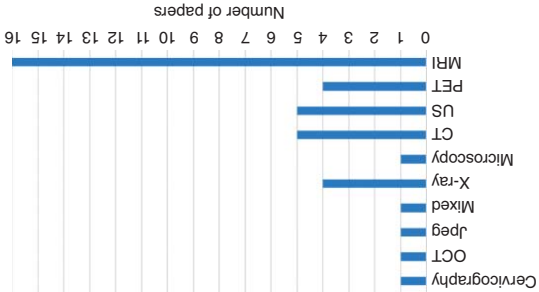
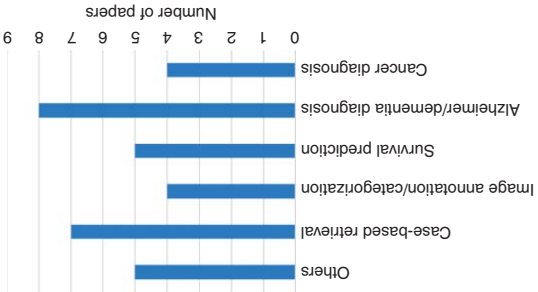
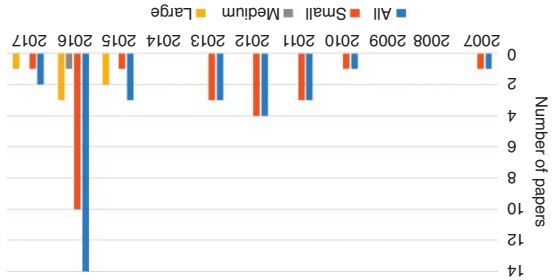


Figure 12.4 Breakdown of papers included in the survey by year and data set size, anatomy, application, and imaging modality.

informatics in a more general sense and [74] focusing on medical image analysis. Studies employing multimodal analysis of medical data covered in this survey generally employ convolutional neural networks for analysis of images and non-image data at the same time [53, 54, 62, 63, 66–69]. One exception is [61], where a dual-network architecture is used with convolutional neural networks for image analysis and recurrent neural networks for extraction of text features from radiology reports.

In a guest editorial on deep learning in medical image analysis [75] the authors also highlight all the opportunities that these techniques have for the future of digital medicine. However, in terms of the exact network architectures for combining several complementary sources of information there is still much work that is required.

12.5 Application Types of Image-Based Decision Support

This section categorizes the papers included in this survey by their applications scenarios. We separate these into several types of applications: (i) *localization* has as objective to find a region of interest in images with specific patterns, (ii) *segmentation* aims to find exact borders of an organ, lesion or other structure, (iii) *classification* aims to attach a finite number of class labels to images or regions of interest, (iv) *prediction* aims to predict a future event, for example treatment response or survival time, (v) *retrieval* aims to respond to the information needs of users or find similar images or regions, and (vi) *automatic image annotation* aims to attach key words or semantic concepts to images or image regions. The underlying techniques (visual characteristics, machine learning, fusion) for all these application areas are similar and related, but they also have differences in their exact objectives.

12.5.1 Localization

Many medical imaging tasks require the detection and localization of anatomical/pathological regions as a first step, so finding a small region that bears the important information of an image. In an exemplary multimodal work, computer-aided detection/localization of mammographic lesions from a large X-ray data set as accomplished by training a convolutional neural network with image features and demographic information [69]. The difficulty of localization is that there is much anatomic variation in medical images and anomalies are often focused on very small parts in the multidimensional data.

12.5.2 Segmentation

Exact delineation of regions of interest (lesions, organs) from medical images is a crucial step for building accurate decision support. To this end, researchers exploit non-image information for improved image segmentation. For example, in [53] semantic concepts mined from clinical reports are used to improve segmentation of optical coherence tomography images of the retina into intraretinal cystoid fluid, subretinal fluid, and normal retinal tissue by convolutional neural networks. Several segmentation challenges of data from single modalities was provided by the VISCERAL project [28].

12.5.3 Classification

Classification of multimodal medical data into “healthy vs. disease”, “normal vs. pathology”, or “different disease stages” is an important task to improve medical decision support. Classification means to add each image, volume or region into one or several of a set of classes. Accordingly, most of the papers included in this survey focus on the classification task. Among these, many target diagnoses of Alzheimer’s from small data sets (the ADNI data sets are publicly available, easing this application domain) by augmenting image features extracted from MRI only [44, 47, 48, 58] or MRI+PET positron emission tomography [42, 45, 46] with non-image features such as demographic indicators [58], clinical findings like CSF biomarkers [44–48, 58], biosignals [42], and genetic test results [48]. They then feed these multimodal features into machine learning algorithms such as naive Bayes [48], support vector machines [44–46, 48] or ensemble learners [42].

Cancer diagnosis is another focus of interest, where examples include glioblastoma diagnosis using microscopy images and genetic test results [43], thyroid cancer diagnosis from ultrasound images and biopsy findings [59], and cervical dysplasia diagnosis from cervicograms, demographic indicators, and clinical findings [68], all realized using small data sets.

Besides the aforementioned articles, [50] performed osteoporosis diagnosis over a small data set by combining information extracted from CT and X-ray images with physical and blood measurement results in an ensemble learner framework, while [51] realized staging of chronic liver disease over a small database by exploiting ultrasound images and laboratory/clinical findings via machine learning.

Other articles include a large-scale modality classification application where images and texts from journal articles are employed [55], and a deep learning application of image categorization into clinically/semantically relevant clusters by using CT and MRI images together with radiology reports [62].

12.5.4 Prediction

Prediction aims to predict a future temporal event based on past data, for example predicting treatment outcome or estimating survival time. The added-value of using multimodal information in medical decision support is shown in the application of cancer survival prediction. Several small-scale studies merge information extracted from structural and functional images like diffusion/perfusion MRI with demographic indicators of subjects and clinical manifestations of tumors. Existing papers typically realize survival prediction by using machine learning or deep learning [49, 63–65, 70].

12.5.5 Retrieval

Retrieval aims to supply information to fulfill an information need. With images this often means finding similar images or similar cases to an example. Case-based retrieval of similar patients is playing an increasingly important role in improving decision support, diagnosis, treatment planning, and physician’s training. In this context, [41] presented a small-scale cardiac decision support system using ultrasound images with electroencephalography data and audio recordings [24] proposed a small-scale retrieval-based dementia diagnosis system where MRI data are used together with

demographic indicators, medical history of patients, and clinical findings. Recently, [56] realized case-based retrieval over a small interstitial lung disease database where CT images are employed with radiology reports [57] implemented a small-scale case-based retrieval system by employing latent semantic topics together with MRI and PET images for Alzheimer's and CT images for lung nodules [66] proposed and evaluated a pathology retrieval system on a small data set of chest X-rays where image information is augmented by demographic indicators.

Besides these, [52] realized medical information discovery from literature by mining JPEG images and texts of a large number of PubMed articles [55] adapted case-based retrieval and modality classification of multiple diseases using images and texts of a large data set of journal articles.

12.5.6 Automatic Image Annotation

Automatic image annotation aims to attach keywords or semantic labels to visual data so they can be searched or classified in an easier way. Semantic annotation of medical images is a prerequisite for building comprehensive archives that can be used for not only evidence-based diagnosis but also biomedical research and education of physicians. As such, recent efforts focus on exploiting multimodal data and deep learning for this problem [54] used MRI and CT images together with clinical reports, while [61] employed X-ray images with radiology reports to annotate large databases of multiple diseases. On the other hand, [67] combined information from Doppler ultrasound images and radiology reports to annotate a small cardiac image database.

12.5.7 Other Application Types

Besides the aforementioned articles, [76] presented a clinical use-case evaluation study for effectiveness of web-based decision support tools for coronary artery disease, where perfusion scintigraphy images, electrocardiography data, the patient's medical history, and clinical findings of a small data set were used [60], on the other hand, proposed a cardiac monitoring system employing ultrasound images with electrocardiography data. These application types are different from the categories described above.

12.6 Discussion on Multimodal Medical Decision Support

The larger number of more recent papers using multimodal (visual, plus structured data or free text) medical data for decision support shows the increasing activity in this research domain. However, most of these use small data sets. Many techniques have been employed and in recent years an increasing number of approaches have used convolutional neural networks and similar techniques. Many variations also try to combine traditional approaches with deep learning, which often leads to good results. A big challenge remains the accessibility of large annotated data sets. Some large data sets have become available, so data itself is slowly becoming less of a problem but annotating data is critical and leveraging small amounts of manual annotations is essential. For this reason, active learning techniques and silver corpus approaches are often required to make the data really usable and useful. Learning from weakly annotated data is another

direction that has started in very focused domains. Having many research data sets that are small and private also creates problems with reproducibility of the research results.

Evaluation campaigns have shown a tremendous impact in many areas [38, 77], and the medical data analysis domain is no exception. Such benchmarking campaigns are now part of all major conferences. This has made data and annotations available with a clear evaluation scenario, so it becomes possible to compare results obtained with these techniques. Still, the size of data sets is nothing compared to what is being made available in terms of web images and maybe relatively new initiatives such as Medical ImageNET⁵ can help in the future when they are fully employed.

12.7 Outlook or the Next Steps of Multimodal Medical Decision Support

Deep learning has had an enormous impact on computer vision over the past five years and in medical image analysis and multimodal data processing the current tools such as Tensorflow⁶ or Caffe⁷ are frequently used. With larger data sets becoming available the full potential of these techniques will likely be seen in a few years. It seems important to adapt the tools to the scenario of medical data where very small parts of an image are usually relevant for the decision making and not the entire image or volume, as is often the case in stock photography. There are often 2000–4000 gray levels in the image and high resolutions, which is not foreseen in many neural network architectures and has to be added to leverage on the full image information, or at least the right level/window levels need to be chosen depending on the organ to be analysed. New networks for visual and multimodal data that include the specific aspects of medical data are expected to be developed and also a systematic approach towards choosing the optimal networks for a specific scenario and type of data.

Fully training networks on medical data is often not possible as the amounts of data available are often too small and the variety in the data is enormous. Thus, transfer learning is used and for images often the ImageNET pretrained networks are employed. This can be expected to change. Larger data sets have now become available and it is a question of how to best obtain annotations and limit the amount of manual work. This requires learning from larger but weakly annotated data sets. It also pushes for active learning approaches to focus annotation efforts on the most informative examples.

Another activity that will likely gain traction is the evaluation of tools in clinical practice and the evaluation of impact on clinical care. Most current tools cited in this paper are research prototypes and few have been evaluated in clinical scenarios even on a modest scale. To show performance in real situations clinical impact needs to be shown and tested in large-scale studies. With an increasing amount of data available on individuals the need to use tools will increase and there will likely also be a difference in quality that can be reached. Even though some people predict that radiologists may not be necessary in the future for evaluating images we think that tools will improve the efficiency of the clinical workflow by automating simple cases, and concentrate the clinician's work on the difficult cases and on interpreting the data and thus also the effectiveness.

5 <http://langlotzlab.stanford.edu/projects/medical-image-net/>.

6 <https://www.tensorflow.org/>.

7 <http://caffe.berkeleyvision.org/>.

It has been shown that it is impossible to read all the literature in even a small focused medical domain [78], so access to recent medical knowledge will remain an important aspect of making decision evidence-based and personalized. The half-life of medical knowledge is estimated to be between 5 and 10 years, and this highlights why decision support is needed: it is important to always include the latest knowledge in precision medicine.

With the ever-increasing amount of medical multimodal data available, big data analytics is becoming more important every day. If the associated challenges — heterogeneity, fragmentation, availability, privacy and security of data — are properly addressed [79], big data analytics has the potential to improve medical decision support by exploiting large multimodal data sets in the near future and eventually transform healthcare. This is only achievable with good data management and computerized decision support.

References

- 1 Haux, R. (2005) Hospital information systems — past, present, future. *International Journal of Medical Informatics*, 75, 268–281.
- 2 Topol, E. (2013) *The creative destruction of medicine – how the digital revolution will create better healthcare*, Basic Books.
- 3 Anthony Celi, L., Mark, R.G., Stone, D.J., and Montgomery, R.A. (2013) “big data” in the intensive care unit. closing the data loop. *American Journal of Respiratory and Critical Care Medicine*, 187 (11), 1157–1160.
- 4 Bates, D.W., Saria, S., Ohno-Machado, L., Shah, A., and Escobar, G. (2014) Big data in health care: Using analytics to identify and manage high-risk and high-cost patients. *Health Affairs*, 33 (7), 1123–1131.
- 5 Groves, P., Kayyali, B., Knott, D., and Kuiken, S.V. (2013) The ‘big data’ revolution in healthcare: Accelerating value and innovation, *Technical Report*, McKinsey & Company, Center for US Health System Reform, Business Technology Office.
- 6 Mirnezami, R., Nicholson, J., and Darzi, A. (2012) Preparing for precision medicine. *New England Journal of Medicine*, 366 (6), 489–491.
- 7 Collins, F.S. and Varmus, H. (2015) A new initiative on precision medicine. *New England Journal of Medicine*, 372 (9), 793–795.
- 8 Kruse, C.S., Goswamy, R., Raval, Y., and Marawi, S. (2016) Challenges and opportunities of big data in health care: A systematic review. *Journal of Medical Internet Research Medical Informatics*, 4 (4), e38.
- 9 Andreu-Perez, J., Poon, C.C.Y., Merrifield, R.D., Wong, S.T.C., and Yang, G.Z. (2015) Big data for health. *IEEE Journal of Biomedical and Health Informatics*, 19 (4), 1193–1208.
- 10 (2010), Riding the wave: How Europe can gain from the rising tide of scientific data, Submission to the European Commission, <http://cordis.europa.eu/fp7/ict/e-infrastructure/docs/hlg-sdi-report.pdf>.
- 11 Wang, L., Ranjan, R., Kolodziej, J., Zomaya, A., and Alem, L. (2015) Software tools and techniques for big data computing in healthcare clouds. *Generation Computer Systems*, 43 (C), 38–39.

- 12 Heinrich, A. et al. (2016) Big data technologies in healthcare: Needs, opportunities and challenges, *Technical Report*, Big Data Value Association: TF7 Healthcare Subgroup.
- 13 Depeursinge, A., Van De Ville, D., Platon, A., Geissbuhler, A., Poletti, P.A., and Müller, H. (2012) Near-affine-invariant texture learning for lung tissue analysis using isotropic wavelet frames. *IEEE Transactions on Information Technology in BioMedicine*, 16 (4), 665–675.
- 14 Aerts, H.J., Velazquez, E.R., Leijenaar, R.T., Parmar, C., Grossmann, P., Carvalho, S., Bussink, J., Monshouwer, R., Haibe-Kains, B., Rietveld, D. et al. (2014) Decoding tumour phenotype by noninvasive imaging using a quantitative radiomics approach. *Nature Communications*, 5 (1), 4006.
- 15 Gurcan, M.N., Sahiner, B., Petrick, N., Chan, H.P., Kazerooni, E.A., Cascade, P.N., and Hadjiiski, L. (2002) Lung nodule detection on thoracic computed tomography images: Preliminary evaluation of computer-aided diagnosis system. *Medical Physics*, 29 (1), 2552–2558.
- 16 Smith, J.T., Hawkins, R.M., Guthrie, J.A., Wilson, D.J., Arnold, P.M., Boyes, S., and Robinson, P.J. (2010) Effect of slice thickness on liver lesion detection and characterisation by multidetector CT. *Journal of Medical Imaging and Radiation Oncology*, 54 (3), 188–193.
- 17 Müller, H., Michoux, N., Bandon, D., and Geissbuhler, A. (2004) A review of content-based image retrieval systems in medicine-clinical benefits and future directions. *International Journal of Medical Informatics*, 73 (1), 1–23.
- 18 Akgül, C., Rubin, D., Napel, S., Beaulieu, C., Greenspan, H., and Acar, B. (2011) Content-based image retrieval in radiology: Current status and future directions. *Journal of Digital Imaging*, 24 (2), 208–222.
- 19 Depeursinge, A., Vargas, A., Gaillard, F., Platon, A., Geissbuhler, A., Poletti, P.A., and Müller, H. (2012) Case-based lung image categorization and retrieval for interstitial lung diseases: clinical workflows. *International Journal of Computer Assisted Radiology and Surgery*, 7 (1), 97–110.
- 20 Jimenez-del-Toro, O., Hanbury, A., Langs, G., Foncubierta-Rodríguez, A., and Müller, H. (2015) Overview of the VISCERAL Retrieval Benchmark 2015, in *Multimodal Retrieval in the Medical Domain (MRMD) 2015, Lecture Notes in Computer Science*, vol. 9059, Springer.
- 21 Schaer, R. and Müller, H. (2014) A modern web interface for medical image retrieval, in *Annual Congress SGMI 2014*, Volume 30.
- 22 May, M. (2014) Life science technologies: Big biological impacts from big data. *Science*, 344 (6189), 1298.
- 23 Jimenez-del-Toro, O., Cirujeda, P., and Müller, H. (2017) Combining radiology images and clinical meta-data for multimodal medical case-based retrieval, in *Cloud-Based Benchmarking of Medical Image Analysis*, Springer International Publishing.
- 24 Soydemir, M. and Unay, D. (2013) Context-aware medical image retrieval for improved dementia diagnosis, in *Intelligent Multimedia Technologies for Networking Applications: Techniques and Tools* (ed. D. Kanellopoulos), IGI Global, Hershey, PA, USA, chap. 18, pp. 434–448.
- 25 Elger, B., Iavindrasana, J., Lo Iacono, L., Müller, H., Roduit, N., Summers, P., and Wright, J. (2010) Strategies for health data exchange for secondary,

- cross-institutional clinical research. *Computer Methods and Programs in Biomedicine*, 99 (3), 230–251.
- 26 Vannier, M.W. and Summers, R.M. (2003) Sharing images. *Radiology*, 228, 23–25.
 - 27 Grünberg, K., Jimenez-del-Toro, O., Jakab, A., Langs, G., Salas, T., Winterstein, M., Weber, M.A., and Krenn, M. (2017) Annotating medical image data, in *Cloud-Based Benchmarking of Medical Image Analysis*, Springer International Publishing.
 - 28 Jimenez-del-Toro, O., Müller, H., Krenn, M., Gruenberg, K., Taha, A.A., Winterstein, M., Eggel, I., Foncubierta-Rodríguez, A., Goksel, O., Jakab, A., Kontokotsios, G., Langs, G., Menze, B., Salas Fernandez, T., Schaer, R., Walleyo, A., Weber, M.A., Dicente Cid, Y., Gass, T., Heinrich, M., Jia, F., Kahl, F., Kechichian, R., Mai, D., Spanier, A.B., Vincent, G., Wang, C., Wyeth, D., and Hanbury, A. (2016) Cloud-based evaluation of anatomical structure segmentation and landmark detection algorithms: VISCERAL Anatomy Benchmarks. *IEEE Transactions on Medical Imaging*, 35 (11), 2459–2475.
 - 29 Krenn, M., Dorfer, M., Jimenez-del-Toro, O., Müller, H., Menze, B., Weber, M.A., Hanbury, A., and Langs, G. (2014) Creating a large-scale silver corpus from multiple algorithmic segmentations, in *MICCAI Workshop on Medical Computer Vision: Algorithms for Big Data, MCV 2015*, vol. 8848, Springer, pp. 163–170.
 - 30 Hofmanninger, J., Krenn, M., Holzer, M., Schlegl, T., Prosch, H., and Langs, G. (2016) Unsupervised identification of clinically relevant clusters in routine imaging data, in *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, Springer International Publishing.
 - 31 Aziz Taha, A. and Hanbury, A. (2015) Metrics for evaluating 3D medical image segmentation: analysis, selection, and tool. *BMC Medical Imaging*, 15 (1).
 - 32 Armstrong, T.G., Moffat, A., Webber, W., and Zobel, J. (2009) Improvements that don't add up: ad-hoc retrieval results since 1998, in *Proceedings of the 18th ACM Conference on Information and Knowledge Management (CIKM '09)*, ACM, pp. 601–610, doi: <http://doi.acm.org/10.1145/1645953.1646031>.
 - 33 Kalpathy-Cramer, J., García Seco de Herrera, A., Demner-Fushman, D., Antani, S., Bedrick, S., and Müller, H. (2015) Evaluating performance of biomedical image retrieval systems: Overview of the medical image retrieval task at ImageCLEF 2004–2014. *Computerized Medical Imaging and Graphics*, 39 (0), 55 –61.
 - 34 Heimann, T., Van Ginneken, B., Styner, M., Arzhaeva, Y., Aurich, V., Bauer, C., Beck, A., Becker, C., Beichel, R., Bekes, G. et al. (2009) Comparison and evaluation of methods for liver segmentation from CT datasets. *IEEE Transactions on Medical Imaging*, 28 (8), 1251–1265.
 - 35 Menze, B.H., Jakab, A., Bauer, S., Kalpathy-Cramer, J., Farahani, K., Kirby, J., Burren, Y., Porz, N., Slotboom, J., Wiest, R., Lanczi, L., Gerstner, E., Weber, M.A., Arbel, T., Avants, B.B., Ayache, N., Buendia, P., Collins, D.L., Cordier, N., Corso, J.J., Criminisi, A., Das, T., Delingette, H., Demiralp, C., Durst, C.R., Dojat, M., Doyle, S., Festa, J., Forbes, F., Geremia, E., Glocker, B., Golland, P., Guo, X., Hamamci, A., Iftekharuddin, K.M., Jena, R., John, N.M., Konukoglu, E., Lashkari, D., Mariz, J.A., Meier, R., Pereira, S., Precup, D., Price, S.J., Raviv, T.R., Reza, S.M.S., Ryan, M., Sarikaya, D., Schwartz, L., Shin, H.C., Shotton, J., Silva, C.A., Sousa, N., Subbanna, N.K., Szekely, G., Taylor, T.J., Thomas, O.M., Tustison, N.J., Unal, G., Vasseur, F., Wintermark, M., Ye, D.H., Zhao, L., Zhao, B., Zikic, D., Prastawa, M., Reyes, M.,

- and Van Leemput, K. (2015) The Multimodal Brain Tumor Image Segmentation Benchmark (BRATS). *IEEE Transactions on Medical Imaging*, 34 (10), 1993–2024.
- 36 Jimenez-del-Toro, O., Goksel, O., Menze, B., Müller, H., Langs, G., Weber, M.A., Eggel, I., Gruenberg, K., Holzer, M., Kotsios-Kontokotsios, G., Krenn, M., Schaer, R., Taha, A.A., Winterstein, M., and Hanbury, A. (2014) VISCERAL – VISual Concept Extraction challenge in RadioLogY: ISBI 2014 challenge organization, in *Proceedings of the VISCERAL Challenge at ISBI* (ed. O. Goksel), Beijing, China, no. 1194 in CEUR Workshop Proceedings, pp. 6–15.
 - 37 Rowe, B.R., Wood, D.W., Link, A.N., and Simoni, D.A. (2010) Economic impact assessment of NIST text retrieval conference (TREC) program, *Technical report project number 0211875*, National Institute of Standards and Technology.
 - 38 Tsikrika, T., García Seco de Herrera, A., and Müller, H. (2011) Assessing the scholarly impact of ImageCLEF, in *CLEF 2011*, Springer Lecture Notes in Computer Science, pp. 95–106.
 - 39 Hanbury, A., Müller, H., Langs, G., Weber, M.A., Menze, B.H., and Fernandez, T.S. (2012) Bringing the algorithms to the data: cloud-based benchmarking for medical image analysis, in *CLEF Conference*, Springer Lecture Notes in Computer Science.
 - 40 Hanbury, A., Müller, H., Balog, K., Brodt, T., Cormack, G.V., Eggel, I., Gollub, T., Hopfgartner, F., Kalpathy-Cramer, J., Kando, N., Krithara, A., Lin, J., Mercer, S., and Potthast, M. (2015) Evaluation-as-a-service: Overview and outlook. *ArXiv*, 1512.07454.
 - 41 Syeda-Mahmood, T., Wang, F., Beymer, D., Amir, A., Richmond, M., and Hashmi, S.N. (2007) Aalim: Multimodal mining for cardiac decision support. *2007 Computers in Cardiology*, pp. 209–212.
 - 42 Polikar, R., Tilley, C., Hillis, B., and Clark, C.M. (2010) Multimodal EEG, MRI and PET data fusion for Alzheimer's disease diagnosis. *2010 Annual International Conference of the IEEE Engineering in Medicine and Biology*, pp. 6058–6061.
 - 43 Kong, J., Cooper, L.A.D., Wang, F., Gutman, D.A., Gao, J., Chisolm, C., Sharma, A., Pan, T., Van Meir, E.G., Kurc, T.M., Moreno, C.S., Saltz, J.H., and Brat, D.J. (2011) Integrative, multimodal analysis of glioblastoma using TCGA molecular data, pathology images, and clinical outcomes. *IEEE Transactions on Biomedical Engineering*, 58 (12), 3469–3474.
 - 44 Davatzikos, C., Bhatt, P., Shaw, L.M., Batmanghelich, K.N., and Trojanowski, J.Q. (2011) Prediction of MCI to AD conversion, via MRI, CSF biomarkers, and pattern classification. *Neurobiology of Aging*, 32 (12), 2322.e19–2322.e27.
 - 45 Zhang, D., Wang, Y., Zhou, L., Yuan, H., and Shen, D. (2011) Multimodal classification of Alzheimer's disease and mild cognitive impairment. *NeuroImage*, 55 (3), 856–867.
 - 46 Zhang, D. and Shen, D. (2012) Multi-modal multi-task learning for joint prediction of multiple regression and classification variables in Alzheimer's disease. *NeuroImage*, 59 (2), 895–907.
 - 47 Westman, E., Muehlboeck, J.S., and Simmons, A. (2012) Combining MRI and CSF measures for classification of Alzheimer's disease and prediction of mild cognitive impairment conversion. *NeuroImage*, 62 (1), 229–238.
 - 48 Mattila, J., Koikkalainen, J., Virkki, A., van Gils, M. and Lötjönen, J. for the Alzheimer's Disease Neuroimaging Initiative (2012) Design and application of a

- generic clinical decision support system for multiscale data. *IEEE Transactions on Biomedical Engineering*, 59 (1), 234–240.
- 49 Zacharaki, E.I., Morita, N., Bhatt, P., O'Rourke, D.M., Melhem, E.R., and Davatzikos, C. (2012) Survival analysis of patients with high-grade gliomas based on data mining of imaging variables. *American Journal of Neuroradiology*, 33 (6), 1065.
 - 50 Tay, W.L., Chui, C.K., Ong, S.H., and Ng, A.C.M. (2013) Ensemble-based regression analysis of multimodal medical data for osteopenia diagnosis. *Expert Systems with Applications*, 40 (2), 811–819.
 - 51 Ribeiro, R.T., Marinho, R.T., and Sanches, J.M. (2013) Classification and staging of chronic liver disease from multimodal data. *IEEE Transactions on Biomedical Engineering*, 60 (5), 1336–1344.
 - 52 Mourão, A., Martins, F., and Magalhães, J. (2015) Multimodal medical information retrieval with unsupervised rank fusion. *Computerized Medical Imaging and Graphics*, 39, 35–45.
 - 53 Schlegl, T., Waldstein, S.M., Vogl, W.D., Schmidt-Erfurth, U., and Langs, G. (2015) *Predicting Semantic Descriptions from Medical Images with Convolutional Neural Networks*, Springer International Publishing, Cham, pp. 437–448.
 - 54 Shin, H.C., Lu, L., Kim, L., Seff, A., Yao, J., and Summers, R.M. (2015) Interleaved text/image deep mining on a large-scale radiology database. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1090–1099.
 - 55 Kitanovski, I., Strezoski, G., Dimitrovski, I., Madjarov, G., and Loskovska, S. (2016) *Multimodal medical image retrieval system. Multimedia Tools and Applications*, pp. 1–24.
 - 56 Ramos, J., Kockelkorn, T.T.J.P., Ramos, I., Ramos, R., Grutters, J., Viergever, M.A., van Ginneken, B., and Campilho, A. (2016) Content-based image retrieval by metric learning from radiology reports: Application to interstitial lung diseases. *IEEE Journal of Biomedical and Health Informatics*, 20 (1), 281–292.
 - 57 Zhang, F., Song, Y., Cai, W., Hauptmann, A.G., Liu, S., Pujol, S., Kikinis, R., Fulham, M.J., Feng, D.D., and Chen, M. (2016) Dictionary pruning with visual word significance for medical image retrieval. *Neurocomputing*, 177, 75–88.
 - 58 Rhodius-Meester, H.F.M., Koikkalainen, J., Mattila, J., Teunissen, C.E., Barkhof, F., Lemstra, A.W., Scheltens, P., Lotjonen, J., and van der Flier, W.M. (2016) Integrating biomarkers for underlying alzheimer's disease in mild cognitive impairment in daily practice: Comparison of a clinical decision support system with individual biomarkers. *Journal of Alzheimers Disease*, 50 (1), 261–270.
 - 59 Kim, T.H., Jeong, D.J., Hahn, S.Y., Shin, J.H., Oh, Y.L., Ki, C.S., Kim, J.W., Jang, J.Y., Cho, Y.Y., Chung, J.H., and Kim, S.W. (2016) Triage of patients with AUS/FLUS on thyroid cytopathology: effectiveness of the multimodal diagnostic techniques. *Cancer Medicine*, 5 (5), 769–777.
 - 60 Jatrniko, W., Arsa, D.M.S., Wisesa, H., Jati, G., and Ma'sum, M.A. (2016) A review of big data analytics in the biomedical field. *2016 International Workshop on Big Data and Information Security (IWBIS)*, pp. 31–41.
 - 61 Shin, H., Roberts, K., Lu, L., Demner-Fushman, D., Yao, J., and Summers, R.M. (2016) Learning to read chest x-rays: Recurrent neural cascade model for automated image annotation. *Computing Research Repository*, abs/1603.08486.
 - 62 Wang, X., Lu, L., Shin, H., Kim, L., Nogues, I., Yao, J., and Summers, R.M. (2016) Unsupervised category discovery via looped deep pseudo-task optimization

- using a large scale radiology image database. *Computing Research Repository*, abs/1603.07965.
- 63 Nie, D., Zhang, H., Adeli, E., Liu, L., and Shen, D. (2016) *3D Deep Learning for Multi-modal Imaging-Guided Survival Time Prediction of Brain Tumor Patients*, Springer International Publishing, Cham, pp. 212–220.
 - 64 Burth, S., Kickingeder, P., Eidel, O., Tichy, D., Bonekamp, D., Weberling, L., Wick, A., Löw, S., Hertenstein, A., Nowosielski, M., Schlemmer, H.P., Wick, W., Bendszus, M., and Radbruch, A. (2016) Clinical parameters outweigh diffusion- and perfusion-derived MRI parameters in predicting survival in newly diagnosed glioblastoma. *Neuro-Oncology*, 18 (12), 1673–1679.
 - 65 Chang, K., Zhang, B., Guo, X., Zong, M., Rahman, R., Sanchez, D., Winder, N., Reardon, D.A., Zhao, B., Wen, P.Y., and Huang, R.Y. (2016) Multimodal imaging patterns predict survival in recurrent glioblastoma patients treated with bevacizumab. *Neuro-Oncology*, 18 (12), 1680–1687.
 - 66 Anavi, Y., Kogan, I., Gelbart, E., Geva, O., and Greenspan, H. (2016) Visualizing and enhancing a deep learning framework using patients age and gender for chest x-ray image retrieval, vol. 9785, pp. 978510–978510–6.
 - 67 Moradi, M., Guo, Y., Gur, Y., Negahdar, M., and Syeda-Mahmood, T. (2016) *A Cross-Modality Neural Network Transform for Semi-automatic Medical Image Annotation*, Springer International Publishing, Cham, pp. 300–307.
 - 68 Xu, T., Zhang, H., Huang, X., Zhang, S., and Metaxas, D.N. (2016) *Multimodal Deep Learning for Cervical Dysplasia Diagnosis*, Springer International Publishing, Cham, pp. 115–123.
 - 69 Kooi, T., Litjens, G., van Ginneken, B., Gubern-Mérida, A., Sánchez, C.I., Mann, R., den Heeten, A., and Karssemeijer, N. (2017) Large scale deep learning for computer aided detection of mammographic lesions. *Medical Image Analysis*, 35, 303–312.
 - 70 Zhang, B., Chang, K., Ramkissoon, S., Tanguturi, S., Bi, W.L., Reardon, D.A., Ligon, K.L., Alexander, B.M., Wen, P.Y., and Huang, R.Y. (2017) Multimodal MRI features predict isocitrate dehydrogenase genotype in high-grade gliomas. *Neuro-Oncology*, 19 (1), 109–117.
 - 71 Depeursinge, A. and Müller, H. (2010) Fusion techniques for combining textual and visual information retrieval, in *ImageCLEF, The Springer International Series On Information Retrieval*, vol. 32 (eds H. Müller, P. Clough, T. Deselaers, and B. Caputo), Springer, Berlin, Heidelberg, pp. 95–114.
 - 72 LeCun, Y., Bengio, Y., and Hinton, G. (2015) Deep learning. *Nature*, 521 (7553), 436–444.
 - 73 Ravi, D., Wong, C., Deligianni, F., Berthelot, M., Andreu-Perez, J., Lo, B., and Yang, G.Z. (2017) Deep learning for health informatics. *IEEE Journal of Biomedical and Health Informatics*, 21 (1), 4–21.
 - 74 Litjens, G.J.S., Kooi, T., Bejnordi, B.E., Setio, A.A.A., Ciompi, F., Ghafoorian, M., van der Laak, J.A.W.M., van Ginneken, B., and Sánchez, C.I. (2017) A survey on deep learning in medical image analysis. *Computing Research Repository*, abs/1702.05747.
 - 75 Greenspan, H., van Ginneken, B., and Summers, R.M. (2016) Guest editorial deep learning in medical imaging: Overview and future promise of an exciting new technique. *IEEE Transactions on Medical Imaging*, 35 (5), 1153–1159.

- 76 Lin, F.Y., Dunning, A.M., Narula, J., Shaw, L.J., Gransar, H., Berman, D.S., and Min, J.K. (2013) Impact of an automated multimodality point-of-order decision support tool on rates of appropriate testing and clinical decision making for individuals with suspected coronary artery disease: A prospective multicenter study. *Journal of the American College of Cardiology*, 62 (4), 308–316.
- 77 Thornley, C.V., Johnson, A.C., Smeaton, A.F., and Lee, H. (2011) The scholarly impact of TRECVID (2003–2009). *Journal of the American Society for Information Science and Technology*, 62 (4), 613–627.
- 78 Fraser, A.G. and Dunstan, F.D. (2010) On the impossibility of being expert. *British Medical Journal*, 341.
- 79 Kayadjanian, N., Barros, H., Billig, H., Capron, M., Cohen, A., Dubochet, G., Frackowiak, R., Grüters-Kieslich, A., Holgate, S., Kuster, S., Pozzan, T., Vingron, M., Wolff-Boenisch, B., and Radwanska, M. (2014) How to transform big data into better health: Envisioning a health big data ecosystem for advancing biomedical research and improving health outcomes in europe, *Technical Report*, Science Europe: Medical Sciences Committee.

Conclusions and Future Trends

In this book we have discussed the most important state-of-the-art approaches for big data multimedia retrieval and analysis focusing on audiovisual content. Starting from multimedia representation we discussed feature extraction, high level concept, and event-based indexing. We then presented data analysis algorithms for multimedia based on machine learning and discussed the social dimension of multimedia and privacy challenges. Subsequently, we provided an insight into big data storage, retrieval structures, and algorithms. Finally, we presented applications of big multimedia data search focusing on the user interface and on applications in different domains, such as health.

As discussed, mining, analysis and search of big data multimedia content is an important challenge in itself, mainly due to three attributes of big data (volume, velocity and variety), and the very multimodal nature of the multimedia. It is important for the analysis algorithms to be scalable in all the three respects right from the feature extraction step. The machine learning approaches rely heavily on the representation of the data in the feature space, with only the relevant discriminative features retained and all of the redundant or irrelevant features removed. Deep learning algorithms are able to learn the most relevant features or even the feature transforms implicitly, which makes them a promising candidate for scalability in terms of both volume and variety. In addition, deep learning approaches can model both the temporal and the spatial, and their intertwined correlations as useful features, demonstrating scalability in terms of variety. With respect to velocity, the advantage due to data and process parallelizations is significant, especially when the machine learning approach uses the same set of sub-functions iteratively at different hierarchical levels. With respect to big data indexing and retrieval an important conclusion is that while media storage and processing are well served by big data systems, this is not yet the case for scalable query processing over all multimedia metadata.

With respect to future directions and trends, important research challenges in big multimedia data analysis are still open in several fields such as infrastructure, storage and databases, cloud computing, privacy, deep learning, and in particular unsupervised methods, as well as access and presentation.

Regarding infrastructure, GPU processing is being used along with CPU, and the cloud is increasingly used. It is expected that in the multimedia big data era, more stable and powerful computer processors will be required. In addition, with the penetration of Internet of Things (IoT), the quantity of Internet multimedia data will further increase,

which will require large distributed data warehouses. Multimedia big data will require scalable systems with a set of algorithms and mechanisms to load, extract, transform, integrate, and analyse the heterogeneous data by taking advantage of the massively parallel processing technology. The design and development of fast algorithms, including optimization, approximation, statistics, and applied mathematics, that are scalable to massive data with a high dimensionality is also a future research direction. With respect to big multimedia data storage, on the one hand scaling relational databases requires substantial processing power and expensive commodity hardware and storage. On the other hand, although NoSQL stores have been designed for big data storage, elasticity and durability are still challenges for NoSQL databases that need to be configured for future systems. Also, cloud computing has emerged to be one of the most robust and flexible multimedia big data techniques. However, there are specific disadvantages of cloud computing that introduce important challenges in future research, such as the cost of large-scale data storage and transfer, control over the distributed environment, data privacy and security. Specifically, it is critical to provide the support of privacy control from different levels of the mobile system to ensure data security. The potential mechanisms include security storage and management, multi-granularity access control, and privacy-aware data mining and analysis. A multimedia big data platform should find the balance between secure access control and processing convenience. Furthermore, special care should be taken with regard to the security of multimedia content, such as the encryption of multimedia data. With respect to large-scale data analysis the application of deep learning still has several open research directions. For instance, multimodal deep learning techniques are required to analyse different modalities of data, while handling high-dimensional, heterogeneous, and unlabeled multimedia data has great potential for future deep learning research. Finally, visualization tools are crucial to multimedia big data in order to allow efficient data access.

Data is considered by many as the new oil, the fuel of a new economy and society, and as such big data analysis and sciences are the pillar technologies for future multimedia and artificial intelligence applications.

Index

a

accelerated generalised subclass
 discriminant analysis (AGSDA) 32,
 35, 46
 active learning, probabilistic fusion
 incorporating informativeness 139
 Oracle's confidence 139–140
 $P(S|V, T)$, 138
 adjacency matrix (A) 67
 adjective noun pairs (ANPs) 281
 AlexNet 78–80
 data augmentation 7–8
 dropout 8
 learning model aspects 6
 ReLU nonlinearity 6–7
 Alluxio 227–228
 animated GIFs
 data annotation 112–114
 data collection 111–112
 Apache Flink 168
 Apache Hadoop 167
 Apache Samza 168
 Apache Spark 167–168
 Apache Storm 168
 aspect model 68
 auDeep 80
 audio data privacy 194–195
 autoencoder 71–72
 automatically distributed computing
 frameworks (ADCFs) 213
 automatic image annotation 328

b

back-propagation (BP) 272
 baseline CBCF (BCBCF) 36
 batch media processing
 ADCFs 223
 comparison 226
 Map-Reduce and Hadoop 224–225
 Spark 225
 batch-processing frameworks 167–168
 big data
 automatically distributed computing
 frameworks 213
 distributed file systems 213
 NoSQL systems 213
 privacy 192–194
 3vs dimensions 63
 bounding box (bbox) 278

c

CAP theorem 221
 cascade architecture
 algorithm 39–40
 development of 37
 threshold assignment and stage 38–39
 causal convolutional layers 75
 CC_WEB_VIDEO dataset 99, 100
 Celery 169
 Ceph 222
 CNNs *see* convolutional neural networks
 (CNNs)
 codebook 16, 17

- Columbia Consumer Video (CCV) 102–103
- combined dataset 99–101
- comparison algorithms 100
- complexity analysis 96–97
- compression encoder 71–72
- computational complexity method 47–48
- concept-based video search
 - cascade architecture 37–40
 - computational complexity 47–48
 - dataset and experimental setup 42–43
 - exploiting label relations 41–42
 - learning areas for 33
 - multi-task learning 40–41
 - results 43–46
 - video concept detection 35–36
- concept detectors, training 36
- concept language model (CLM) 51
- content authentication 244
- content-based video hashing 92
- content distribution networks (CDN) 228
- content identification 244
- context memory model (CMM) 107, 108
- convolutional filters 10–11
- convolutional neural networks (CNNs)
 - AlexNet 6–8
 - architectures of 78–79
 - based feature extraction 77–78
 - data-driven algorithms 3
 - deep learning-based feature learning 72–73
 - feature extraction 5
 - GoogLeNet 11–13
 - graph-based image 306, 307
 - ImageNet subset 5–6
 - neural networks 4
 - NIN 8–10
 - process parallelization 80–81
 - ResNet 13–15
 - VGG 10–11
- convolutional recurrent neural networks 75
- correlation-based pruning of stacked binary relevance models (BSBRM) 36

- correlation component manifold space learning (CCMSL) 104
- Cosine similarity 291
- cross-modal fusion 124
- CUB-200–2011 277–280
- curse of dimensionality 62
- cybercasing 184

d

- data augmentation 7–8
- data-driven 3, 63
- data-mining 62
- data parallelization 64–65
- data privacy
 - audio data 194–195
 - multimedia data 202–203
 - multimodal threats 195–196
 - privacy level 194
 - private information retrieval 201
 - visual data 195
- deep convolutional neural networks (DCNNs) 31, 34, 92
- deep image tagging
 - fine-grained sentiment analysis
 - learning deep sentiment representation 282–283
 - SentiBank, experiments 283–284
 - sentiment analysis 283
 - fine-grained visual analysis
 - classification and ranking 275–276
 - CUB-200–2011, experiments 277–280
 - implementation 276–277
 - learning process 279
 - multi-scale joint representation 276
 - proposal network attention 274–275
 - Stanford dogs, experiments 280–281
- deep learning
 - approach 69
 - general machine learning 323, 326, 329
 - models 270–272
- deep learning-based feature learning
 - autoencoder 72
 - convolutional neural networks 72–73
 - modular approach to scalability 74–75
 - multilayer perceptron model 70
 - recurrent neural networks 73–74

- degree matrix (D) 67
- digital medicine 317
- dimensionality reduction techniques 34–35, 292, 293
- dimension reduction 12–13
- dimensions of big data 62
- discriminative model fusion (DMF) 36
- distributed data storage
 - CAP theorem 221
 - Ceph 222
 - distributed hash tables 221
 - Hadoop distributed file system 221–222
 - PACELC formulation 221
 - storage hierarchy 217–218
- distributed processing frameworks 168–169
- document stores 229
- dropout, AlexNet 8
- dynamic data 216
- e**
 - electronic health record 318
 - end-to-end learning 76
 - event-based video search
 - experimental results 53
 - multimedia event detection 52–53
 - textual descriptors 50–52
 - visual information 50
 - zero-example learning 53–54
 - zero-shot learning 49
 - event language model (ELM) 51
 - event video mashup (EVM) 104
 - evolutionary algorithms 295
 - extreme learning machines (ELMs) 75
- f**
 - feature engineering
 - defined 64
 - feature reduction, spatial transformations 66
 - handcrafted features 65–66
 - Laplacian matrix 66–68
 - latent Dirichlet allocation 68
 - feature extraction 3, 16, 69
 - fine-grained discriminative regions 272
 - fine-grained sentiment analysis, deep image tagging
 - learning deep sentiment representation 282–283
 - SentiBank, experiments 283–284
 - sentiment analysis 283
 - fine-grained visual analysis
 - basic deep learning models 270–272
 - deep image tagging
 - classification and ranking 275–276
 - CUB-200–2011, experiments 277–280
 - implementation 276–277
 - learning process 279
 - multi-scale joint representation 276
 - proposal network attention 274–275
 - Stanford dogs, experiments 280–281
 - Fisher vector (FV) 34, 240
 - five-minute rule 218–219
- g**
 - gated recurrent unit (GRU) 74
 - General Data Protection Regulation (GDPR) 189
 - generalised subclass discriminant analysis (GSDA) 34
 - generative adversarial networks (GANs) 75
 - geographical content, social multimedia 170
 - GIF animation
 - data annotation 112–114
 - data collection 111–112
 - GIST 241
 - global average pooling 9–10
 - GoogLeNet
 - dimension reduction 12–13
 - inception modules 11–12
 - learning model aspects 11
 - graph-based approaches 290
 - graph-based event video 109–111
 - graph-based fusion 124–126
 - graph-based image browsing
 - generating semantic image features 306–307
 - image graph construction 307–310
 - prototype for 312–313
 - visualization and navigation 310–311

- graph-based video model
 - context association model 107–109
 - experiments 114–116
 - graph-based event video 109–111
 - optimized graph learning 104–107
 - overview 103–104
 - TGIF 111–114
- graph stores 229

h

- Hadoop distributed file system (HDFS) 221–222
- Hamming distance 241
- hard disk drives (HDDs) 218
- hash bit selection 258–260
- hash generation 241
- hash value 239, 240
- hash verification 241
- hierarchical late fusion 35
- histogram of oriented gradients (HOG) 269

i

- image-based decision support application
 - automatic image annotation 328
 - classification 327
 - localization 326
 - prediction 327
 - retrieval 327–328
 - segmentation 326
- image content analysis, Twitter Firehose
 - classification types 161–163
 - top concepts 163–164
- ImageNet 4, 144
- ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) 4
- image retrieval system 319, 320
- image sorting algorithms
 - evolutionary algorithms 295
 - experiments 302–304
 - quality evaluation of 298–301
 - self-organizing maps 293–294
 - self-sorting maps 294
 - test image sets 301–302
- inception modules
 - with dimension reduction 13
 - GoogLeNet 11–12

- naive version of 12
- inferotemporal cortex (IT) 5
- information privacy 185
- intellectual property protection 199
- intentional privacy 187, 190
- iterative quantization (ITQ) 246–247

k

- Karush-Kuhn-Tucker approach 107
- kernel-based supervised hashing (KSH) 252–253
- kernelized locality sensitive hashing (KLSH) 249–250
- key-value stores 229
- K*-means hashing (KMH) 247–249

l

- label powerset (LP) 41
- Lambda architecture 214–215
- Laplacian matrix (L) 66–68
- large and small data
 - CNNs
 - AlexNet 6–8
 - GoogLeNet 11–13
 - network in network 8–10
 - ResNet 13–15
 - VGG 10–11
 - feature extraction 3
 - transfer representation learning
 - experimental results 18–23
 - observations 23
 - specification methods 17–18
- large-scale process, multimedia *see also* social multimedia
 - batch-processing frameworks 167–168
 - distributed processing frameworks 168–169
 - geographical content analysis 170
 - stream-processing frameworks 168
 - textual content analysis 169–170
 - user content analysis 170
 - visual content analysis 169
- large-scale video
 - graph-based model
 - context association model 107–109
 - experiments 114–116
 - graph-based event video 109–111

- optimized graph learning 104–107
 - overview 103–104
 - TGIF 111–114
 - video retrieval with hashing
 - experiments 99–103
 - multiple feature hashing 93–97
 - overview 91–93
 - submodular video hashing 97–99
 - large video collections
 - concept detection and concept-based search
 - cascade architecture 37–40
 - computational complexity 47–48
 - dataset and experimental setup 42–43
 - exploiting label relations 41–42
 - multi-task learning 40–41
 - results 43–46
 - video concept detection 35–36
 - event detection and event-based search
 - decision function 49–50
 - experimental results 53
 - multimedia event detection 52–53
 - textual descriptors 50–52
 - zero-example learning 53–54
 - zero-shot learning 49
 - video preprocessing and machine learning
 - dimensionality reduction 34–35
 - video representation 33–34
 - Lasso multi-task learning 40–41
 - late fusion methods 35
 - rank fusion 133
 - score fusion 132–133
 - support vector machine 130
 - latent Dirichlet allocation 68
 - latent semantic indexing (LSI) 68
 - learning deep sentiment representation 282–283
 - linear discriminant analysis (LDA) 66, 67
 - linear support vector machines (LSVMs) 34
 - local data storage 219–220
 - logistic regression (LR) 36
 - long short-term memory (LSTMs) 74, 92
 - low-level visual patterns 269
- m**
- machine learning 323
 - algorithm 64
 - Map-Reduce framework 224–225
 - max-pooling 271
 - mean average precision (MAP) 53
 - CC_WEB_VIDEO 100
 - combined dataset 101
 - mean extended inferred average precision (MXInfAP) 43–45
 - mean inferred average precision (mInfAP) 53, 54
 - mean-pooling 271
 - media data storage
 - distributed storage 220–222
 - storage hierarchy
 - five-minute rule 218–219
 - local storage 219–220
 - main memory 218
 - secondary storage 218
 - Media Event Detection (MED)
 - dataset 102–103
 - zero-example event detection 49
 - media processing
 - batch processing
 - comparison 226
 - Map-Reduce and Hadoop 224–225
 - Spark 225
 - metadata extraction 223
 - stream processing 226
 - medical data access
 - data annotation and ground truthing 321
 - data resources 322
 - scientific challenges 321–322
 - medical decision support 317
 - medical image retrieval 319
 - Memcached 227
 - memory hierarchy
 - five-minute rule 218–219
 - local storage 219–220
 - main memory 218
 - secondary storage 218
 - metadata extraction 223
 - metadata storage
 - architecture 214–216
 - dynamic data 216

metadata storage (*contd.*)

 Lambda architecture 214–215

 processing layer 216

 serving layer 216

 storage layer 215–216

mlpconv layer 9

model-centric algorithm 3

multi-cue fusion (MCF) method 36

multi feature hashing (MFH)

 algorithm 96

 framework based 93

 hashing methods 92

 objective function of 93–95

 solution of 95–97

multi-label classification algorithm 41

multilayer perceptron (MLP)

 convolutional layer 9

 data-driven algorithms 3

 volume and velocity 70

multimedia big data mining

 benchmark studies

 CNN 77–79

 convolutional neural network 80–81

 dataset 76–77

 process parallelization 79–80

 spectrogram 77

 deep learning-based feature learning

 autoencoder 72

 convolutional neural networks 72–73

 modular approach to scalability
74–75

 multilayer perceptron model 70

 recurrent neural networks 73–74

 feature engineering

 feature reduction, spatial
transformations 66

 Laplacian matrix 66–68

 latent Dirichlet allocation 68

 parallelization 64–65

multimedia classification

 active learning, probabilistic fusion
137–141

 incorporating informativeness 139

 Oracle's confidence 139–140

$P(S|V, T)$, 138

 experiment comparison

 datasets 141–142

 existing methods 148–151

 fusion approaches 147

 parameter sensitivity investigation
147–148

 positive, negative/both expanding
143–145

 sample selection approaches
145–147

 settings 142–143

 problem formulation 136–137

 SALIC 134, 135

multimedia data storage, management

 applications and scale 212

 big data management 213

 data popularity 230

 media processing

 batch processing 223–226

 metadata extraction 223

 stream processing 226

 media storage

 distributed storage 220–222

 storage hierarchy 217–220

 metadata storage architecture 214–216

 multimedia delivery

 distributed in-memory buffering 227

 metadata retrieval and NoSQL systems
228–229

 system architecture 213–214

multimedia event detection (MED) 52–53

multimedia opinion data mining, large-scale

 overview 171

 social media data crawler 171–173

 social multimedia analysis 173

 visual content analysis

 near-duplicate detection 174–175,
177–178

 synthetic image filter 174–177

multimedia privacy, audiovisual content

 big data 192–194

 challenge research 202–204

 content masking method 198–199

 data privacy 200–201

 data privacy threats 194–196

 filter analysis 196–198

 information privacy 185

 multimedia data dark side 184

 protecting user privacy

- sensitive information 189
 - threat models 191–192
 - reorientation research 204–205
 - security and trust 199–200
 - multimedia retrieval
 - experiment 128–130
 - late fusion methods
 - rank fusion 133
 - score fusion 132–133
 - support vector machine 130
 - notations and definitions 123
 - partial least squares regression 127–128
 - unsupervised fusion
 - cross-modal fusion 124
 - linear and non-linear similarity fusion 123–124
 - randomwalks and graph-based fusion 124–126
 - unifying graph-based model 126–127
 - multimedia search, perceptual hashing 243
 - multimodal fusion
 - multimedia classification
 - active learning, probabilistic fusion 137–141
 - experiment 141–151
 - problem formulation 136–137
 - SALIC 134, 135
 - multimedia retrieval
 - experimental comparison 128–130
 - late fusion methods 130–133
 - partial least squares regression 127–128
 - unsupervised fusion 123–127
 - multimodal medical decision support 329–330
 - evaluation campaigns 329
 - techniques for 323–326
 - multi-task formulation 274
 - multi-task learning (MTL)
 - algorithms 40–41
 - defined 32–33
 - exploiting label relations 41–42
 - video concept detection 36
- n**
- near-duplicate detection images 174–175, 177–178
 - near-duplicate video retrieval (NDVR) 96
 - network in network (NIN)
 - global average pooling 9–10
 - learning model aspects 8
 - MLP convolutional layer 9
 - neural networks 4
 - non-hierarchical image graph 307
 - non-linear discriminant analysis (NDA) 34
 - non-volatile memories (NMV) 220
- o**
- openXBOW 68
 - optimized graph learning (OGL)
 - framework 104
 - iterative optimization 106–107
 - overview of 105
 - SSL 105–106
 - terms and notations 104–105
 - Oracle's confidence 139–140
- p**
- PACELC theorem 221
 - parallelization 64–65
 - partial least squares regression 127–128
 - perceptual hashing
 - advantages 240
 - algorithms, evaluation of 244–245
 - applications of 243–244
 - discrimination 242
 - goal of 242
 - multimedia search 243
 - property of 239
 - robustness 242
 - schematic diagrams of 241
 - personal identifiable information 188
 - personal information 188, 189
 - picking deep filter response (PDFR) 278, 280
 - Pioneer neural network models 4
 - PixelCNN 75
 - prefrontal cortex (PFC) 5
 - principal component analysis (PCA) 7, 66, 67

Privacy and Big Data, 185
 privacy-enhancing technologies (PETs) 200
 probabilistic latent semantic indexing 68
 process parallelization 64–65, 79–80
 proposal network, attention
 attention localization and amplification 274–275
 multi-task formulation 274

q

quality evaluation, of image sorting
 algorithms 299–301
 normalized cross-correlation 299
 self-organizing maps, analysis of 298–299
 quantization error 298

r

radial basis function (RBF) 17
 random edge swapping 308
 rank fusion 133
 receptive field kernel 72
 reciprocal rank fusion (RRF) 133
 rectified linear units (ReLUs) 6–7, 271
 recurrent attention convolutional neural network (RA-CNN) 272
 recurrent autoencoders 75
 recurrent neural network (RNN) 73–74, 92
 Redis 227
 region proposal network (RPN) 274
 Residual learning 13–14
 resilient distributed dataset (RDD) 225
 ResNet
 identitymapping, shortcuts 14–15
 learning model aspects 13
 observations 15
 residual learning 13–14
 restricted Boltzmann machine
 (RBM)-based hashing 253–255

s

SALIC *see* Social Active Learning for Image Classification (SALIC)
 scalable machine learning 63

scale-invariant feature transform (SIFT) 3, 240, 269
 score fusion 132–133
 self-organizing maps (SOMs) 293–294
 improving 295–298
 projection quality 297
 reducing sorting complexity 295–296
 and self-sorting maps 297–298
 self-sorting maps (SSMs) 294
 self-supervised temporal hashing (SSTH) 92
 semantic features 198
 semi-supervised hashing 250–252
 semi-supervised learning (SSL) 105–106
 SentiBank 283–284
 single-task learning (STL) 36, 44
 Social Active Learning for Image Classification (SALIC) 134, 142
 social media data crawler 171–173
 social multimedia
 defined 157–158
 large-scale process
 batch-processing frameworks 167–168
 distributed processing frameworks 168–169
 geographical content analysis 170
 stream-processing frameworks 168
 textual content analysis 169–170
 user content analysis 170
 visual content analysis 169
 opinion mining system
 overview 171
 social media data crawler 171–173
 social multimedia analysis 173
 visual content analysis 174–178
 social multimedia streams 158–159
 Twitter Firehose
 dataset 160
 geographic analysis 164–166
 image content analysis 161–164
 linked resource analysis 160–161
 textual analysis 166–167
 softmax function 283
 solid-state drives (SSDs) 218
 Spark 225
 spectral hashing (SH) 245–246

- spectrogram 77
 - static regular 2D image arrangements 289
 - stochastic gradient descent (SGD) 7
 - storage layer 215–216
 - stream-processing frameworks 168
 - strides 72
 - submodular video hashing (SVH)
 - based hashing methods 92
 - evaluation of 101–103
 - framework 97
 - video pooling 97–98
 - for video retrieval 98–99
 - supervised perceptual hash algorithms
 - kernel-based supervised hashing 252–253
 - restricted Boltzmann machine-based hashing 253–255
 - semi-supervised hashing 250–252
 - supervised semantic-preserving deep hashing 255–257
 - supervised semantic-preserving deep hashing 255–257
 - support vector machine (SVM) 17, 64, 276
 - symmetric positive semidefinite (SPSD) 35
 - synthetic image filter
 - dataset 175–176
 - network fine-tuning vs. handcrafted features 176
 - transferability of features 176–177
 - visual content 174
- t**
- Tensorflow 71
 - textual content analysis 169–170
 - threat models 191–192
 - 3D sensing technology 63
 - transfer representation learning
 - CNN 15–16
 - experimental 18–19
 - melanoma results 20–21
 - observations 23
 - otitis media results 19–20
 - specification methods 17–18
 - visualization 21–23
 - TRECVID 42, 43
 - TRECVID Multimedia Event Detection 101
 - tumblr GIF (TGIF)
 - data annotation 112–114
 - data collection 111–112
 - Twitter-API 160
 - Twitter Firehose
 - dataset 160
 - geographic analysis 164–166
 - image content analysis
 - classification types 161–163
 - top concepts 163–164
 - linked resource analysis 160–161
 - textual analysis 166–167
 - Twitter's historical powertrack (HPT) 171
 - 2D image maps navigation 304–305
 - two-step hashing (TSH) 257–258
- u**
- U-matrix 298
 - universal approximation theorem 70
 - Universal Declaration of Human Rights (UDHR) 190
 - unsupervised fusion model
 - cross-modal fusion 124
 - linear and non-linear similarity fusion 123–124
 - randomwalks and graph-based fusion 124–126
 - unifying graph-based model 126–127
 - unsupervised perceptual hash algorithms
 - iterative quantization 246–247
 - kernelized locality sensitive hashing 249–250
 - K-means hashing 247–249
 - spectral hashing 245–246
 - user content analysis 170
 - user intent 198
 - user privacy protection
 - function creep 190
 - personal identifiable information 188
 - personal information 189
 - sensitive information 189
 - threat models 191–192
- v**
- variety data 62–63

- vector of locally aggregated descriptors (VLAD) 241
 - velocity data 62–63
 - ventral visual pathway 5
 - VGG19 78–80
 - video concept detection 32, 35
 - video datasets 99
 - video pooling 97–98
 - video retrieval, with hashing
 - experiments
 - CC_WEB_VIDEO dataset 100
 - combined dataset 100–101
 - comparison algorithms 100
 - SVH 101–103
 - video datasets 99
 - visual features 99–100
 - multi feature hashing
 - algorithm 96
 - framework based 93
 - objective function of 93–95
 - solution of 95–97
 - overview 91–93
 - proposed framework for 93
 - submodular video hashing
 - framework 97
 - video pooling 97–98
 - Viridis 77
 - visual concepts 128
 - visual content 112
 - near-duplicate detection 174–175, 177–178
 - social multimedia 169
 - synthetic image filter 174–177
 - visual data privacy 195
 - visual descriptors 128
 - Visual Geometry Group (VGG)
 - learning model aspects 10
 - multi-scale training 11
 - very small convolutional filters 10–11
 - visual information 5
 - machine learning 33–35
 - types 50
 - visualization
 - melanoma 23
 - OM symptoms 22
 - visual question answering (VQA) 275
 - volume data 62–63
- W**
- WaveNet 75
 - wide column stores 229
- X**
- XPRIZE tricorder 5
- Z**
- zero-example event detection 49
 - zero-shot learning (ZSL) 49