



# Deep Learning for Medical Image Analysis

Edited by

S. Kevin Zhou  
Hayit Greenspan  
Dinggang Shen



# Deep Learning for Medical Image Analysis

# The Elsevier and MICCAI Society Book Series

## Advisory board

**Stephen Aylward** (*Kitware, United States*)

**David Hawkes** (*University College London, United Kingdom*)

**Kensaku Mori** (*University of Nagoya, Japan*)

**Alison Noble** (*University of Oxford, United Kingdom*)

**Sonia Pujol** (*Harvard University, United States*)

**Daniel Rueckert** (*Imperial College, United Kingdom*)

**Xavier Pennec** (*INRIA Sophia-Antipolis, France*)

**Pierre Jannin** (*University of Rennes, France*)

Also available:

Balocco, Computing and Visualization for Intravascular Imaging and Computer Assisted Stenting, 9780128110188

Wu, Machine Learning and Medical Imaging, 9780128040768

Zhou, Medical Image Recognition, Segmentation and Parsing,  
9780128025819

# Deep Learning for Medical Image Analysis

Edited by  
**S. Kevin Zhou**  
**Hayit Greenspan**  
**Dinggang Shen**



**ACADEMIC PRESS**

An imprint of Elsevier  
[elsevier.com](http://elsevier.com)

Academic Press is an imprint of Elsevier  
125 London Wall, London EC2Y 5AS, United Kingdom  
525 B Street, Suite 1800, San Diego, CA 92101-4495, United States  
50 Hampshire Street, 5th Floor, Cambridge, MA 02139, United States  
The Boulevard, Langford Lane, Kidlington, Oxford OX5 1GB, United Kingdom

Copyright © 2017 Elsevier Inc. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording, or any information storage and retrieval system, without permission in writing from the publisher. Details on how to seek permission, further information about the Publisher's permissions policies and our arrangements with organizations such as the Copyright Clearance Center and the Copyright Licensing Agency, can be found at our website: [www.elsevier.com/permissions](http://www.elsevier.com/permissions).

This book and the individual contributions contained in it are protected under copyright by the Publisher (other than as may be noted herein).

#### Notices

Knowledge and best practice in this field are constantly changing. As new research and experience broaden our understanding, changes in research methods, professional practices, or medical treatment may become necessary.

Practitioners and researchers must always rely on their own experience and knowledge in evaluating and using any information, methods, compounds, or experiments described herein. In using such information or methods they should be mindful of their own safety and the safety of others, including parties for whom they have a professional responsibility.

To the fullest extent of the law, neither the Publisher nor the authors, contributors, or editors, assume any liability for any injury and/or damage to persons or property as a matter of products liability, negligence or otherwise, or from any use or operation of any methods, products, instructions, or ideas contained in the material herein.

#### Library of Congress Cataloging-in-Publication Data

A catalog record for this book is available from the Library of Congress

#### British Library Cataloguing-in-Publication Data

A catalogue record for this book is available from the British Library

ISBN: 978-0-12-810408-8

For information on all Academic Press publications  
visit our website at <https://www.elsevier.com>



Working together  
to grow libraries in  
developing countries

[www.elsevier.com](http://www.elsevier.com) • [www.bookaid.org](http://www.bookaid.org)

*Publisher:* Joe Hayton

*Acquisition Editor:* Tim Pitts

*Editorial Project Manager:* Charlotte Kent

*Production Project Manager:* Julie-Ann Stansfield

*Designer:* Matthew Limbert

Typeset by VTeX

# Contents

Contributors .....	xv
About the Editors .....	xxi
Foreword .....	xxiii

## PART 1 INTRODUCTION

---

<b>CHAPTER 1 An Introduction to Neural Networks and Deep Learning .....</b>	<b>3</b>
Heung-II Suk	
1.1 Introduction .....	3
1.2 Feed-Forward Neural Networks .....	4
1.2.1 Perceptron .....	4
1.2.2 Multi-Layer Neural Network .....	5
1.2.3 Learning in Feed-Forward Neural Networks .....	6
1.3 Convolutional Neural Networks .....	8
1.3.1 Convolution and Pooling Layer .....	8
1.3.2 Computing Gradients .....	9
1.4 Deep Models .....	11
1.4.1 Vanishing Gradient Problem .....	11
1.4.2 Deep Neural Networks .....	12
1.4.3 Deep Generative Models .....	14
1.5 Tricks for Better Learning .....	20
1.5.1 Rectified Linear Unit (ReLU) .....	20
1.5.2 Dropout .....	20
1.5.3 Batch Normalization .....	21
1.6 Open-Source Tools for Deep Learning .....	22
References .....	22
Notes .....	24
<b>CHAPTER 2 An Introduction to Deep Convolutional Neural Nets for Computer Vision .....</b>	<b>25</b>
Suraj Srinivas, Ravi K. Sarvadevabhatla, Konda R. Mopuri, Nikita Prabhu, Srinivas S.S. Kruthiventi and R. Venkatesh Babu	
2.1 Introduction .....	26
2.2 Convolutional Neural Networks .....	27
2.2.1 Building Blocks of CNNs .....	27
2.2.2 Depth .....	29
2.2.3 Learning Algorithm .....	30

2.2.4	Tricks to Increase Performance .....	31
2.2.5	Putting It All Together: AlexNet .....	32
2.2.6	Using Pre-Trained CNNs .....	32
2.2.7	Improving AlexNet .....	33
<b>2.3</b>	CNN Flavors .....	34
2.3.1	Region-Based CNNs.....	34
2.3.2	Fully Convolutional Networks.....	35
2.3.3	Multi-Modal Networks.....	38
2.3.4	CNNs with RNNs .....	40
2.3.5	Hybrid Learning Methods .....	43
<b>2.4</b>	Software for Deep Learning .....	45
	References .....	46

---

## PART 2 MEDICAL IMAGE DETECTION AND RECOGNITION

---

<b>CHAPTER 3</b>	<b>Efficient Medical Image Parsing .....</b>	<b>55</b>
	Florin C. Ghesu, Bogdan Georgescu and Joachim Hornegger	
<b>3.1</b>	Introduction .....	55
<b>3.2</b>	Background and Motivation .....	57
3.2.1	Object Localization and Segmentation: Challenges ..	58
<b>3.3</b>	Methodology .....	58
3.3.1	Problem Formulation .....	58
3.3.2	Sparse Adaptive Deep Neural Networks .....	59
3.3.3	Marginal Space Deep Learning .....	61
3.3.4	An Artificial Agent for Image Parsing .....	64
<b>3.4</b>	Experiments.....	71
3.4.1	Anatomy Detection and Segmentation in 3D .....	71
3.4.2	Landmark Detection in 2D and 3D .....	74
<b>3.5</b>	Conclusion .....	77
	Disclaimer .....	78
	References .....	78
<b>CHAPTER 4</b>	<b>Multi-Instance Multi-Stage Deep Learning for Medical Image Recognition .....</b>	<b>83</b>
	Zhennan Yan, Yiqiang Zhan, Shaoting Zhang, Dimitris Metaxas and Xiang Sean Zhou	
<b>4.1</b>	Introduction .....	83
<b>4.2</b>	Related Work .....	85
<b>4.3</b>	Methodology .....	87
4.3.1	Problem Statement and Framework Overview .....	87
4.3.2	Learning Stage I: Multi-Instance CNN Pre-Train .....	88
4.3.3	Learning Stage II: CNN Boosting .....	90

4.3.4	Run-Time Classification .....	92
<b>4.4</b>	Results .....	93
4.4.1	Image Classification on Synthetic Data .....	93
4.4.2	Body-Part Recognition on CT Slices .....	95
<b>4.5</b>	Discussion and Future Work .....	99
	References .....	100
<b>CHAPTER 5</b>	<b>Automatic Interpretation of Carotid Intima–Media Thickness Videos Using Convolutional Neural Networks.....</b>	<b>105</b>
	Nima Tajbakhsh, Jae Y. Shin, R. Todd Hurst, Christopher B. Kendall and Jianming Liang	
<b>5.1</b>	Introduction .....	106
<b>5.2</b>	Related Work .....	107
<b>5.3</b>	CIMT Protocol .....	109
<b>5.4</b>	Method .....	109
5.4.1	Convolutional Neural Networks (CNNs) .....	109
5.4.2	Frame Selection .....	110
5.4.3	ROI Localization .....	112
5.4.4	Intima–Media Thickness Measurement .....	115
<b>5.5</b>	Experiments .....	117
5.5.1	Pre- and Post-Processing for Frame Selection .....	118
5.5.2	Constrained ROI Localization .....	118
5.5.3	Intima–Media Thickness Measurement .....	121
5.5.4	End-to-End CIMT Measurement .....	123
<b>5.6</b>	Discussion .....	124
<b>5.7</b>	Conclusion .....	128
	Acknowledgement .....	128
	References .....	128
	Notes .....	131
<b>CHAPTER 6</b>	<b>Deep Cascaded Networks for Sparsely Distributed Object Detection from Medical Images .....</b>	<b>133</b>
	Hao Chen, Qi Dou, Lequan Yu, Jing Qin, Lei Zhao, Vincent C.T. Mok, Defeng Wang, Lin Shi and Pheng-Ann Heng	
<b>6.1</b>	Introduction .....	134
<b>6.2</b>	Method .....	136
6.2.1	Coarse Retrieval Model .....	136
6.2.2	Fine Discrimination Model .....	139
<b>6.3</b>	Mitosis Detection from Histology Images .....	139
6.3.1	Background .....	139
6.3.2	Transfer Learning from Cross-Domain .....	140

6.3.3	Dataset and Preprocessing .....	140
6.3.4	Quantitative Evaluation and Comparison .....	141
6.3.5	Computation Cost .....	142
6.4	Cerebral Microbleed Detection from MR Volumes .....	143
6.4.1	Background .....	143
6.4.2	3D Cascaded Networks .....	144
6.4.3	Dataset and Preprocessing .....	146
6.4.4	Quantitative Evaluation and Comparison .....	147
6.4.5	System Implementation .....	149
6.5	Discussion and Conclusion .....	149
	Acknowledgements .....	150
	References .....	150
	Notes .....	154

## **CHAPTER 7 Deep Voting and Structured Regression for Microscopy Image Analysis..... 155**

Yuanpu Xie, Fuyong Xing and Lin Yang

7.1	Deep Voting: A Robust Approach Toward Nucleus Localization in Microscopy Images .....	156
7.1.1	Introduction .....	156
7.1.2	Methodology .....	158
7.1.3	Weighted Voting Density Estimation .....	162
7.1.4	Experiments .....	163
7.1.5	Conclusion .....	165
7.2	Structured Regression for Robust Cell Detection Using Convolutional Neural Network .....	165
7.2.1	Introduction .....	165
7.2.2	Methodology .....	166
7.2.3	Experimental Results .....	169
7.2.4	Conclusion .....	171
	Acknowledgements .....	172
	References .....	172

---

## **PART 3 MEDICAL IMAGE SEGMENTATION**

### **CHAPTER 8 Deep Learning Tissue Segmentation in Cardiac Histopathology Images..... 179**

Jeffrey J. Nirschl, Andrew Janowczyk, Eliot G. Peyster,  
Renee Frank, Kenneth B. Margulies,  
Michael D. Feldman and Anant Madabhushi

8.1	Introduction .....	180
8.2	Experimental Design and Implementation .....	183

8.2.1	Data Set Description .....	183
8.2.2	Manual Ground Truth Annotations .....	183
8.2.3	Implementation .....	183
8.2.4	Training a Model Using Engineered Features .....	185
8.2.5	Experiments .....	186
8.2.6	Testing and Performance Evaluation .....	188
<b>8.3</b>	Results and Discussion .....	188
8.3.1	Experiment 1: Comparison of Deep Learning and Random Forest Segmentation .....	188
8.3.2	Experiment 2: Evaluating the Sensitivity of Deep Learning to Training Data .....	188
<b>8.4</b>	Concluding Remarks .....	191
	Notes .....	191
	Disclosure Statement .....	191
	Funding .....	192
	References .....	192
<b>CHAPTER 9</b>	<b>Deformable MR Prostate Segmentation via Deep Feature Learning and Sparse Patch Matching .....</b>	<b>197</b>
	Yanrong Guo, Yaozong Gao and Dinggang Shen	
<b>9.1</b>	Background .....	197
<b>9.2</b>	Proposed Method .....	199
9.2.1	Related Work .....	199
9.2.2	Learning Deep Feature Representation .....	201
9.2.3	Segmentation Using Learned Feature Representation .....	206
<b>9.3</b>	Experiments .....	211
9.3.1	Evaluation of the Performance of Deep-Learned Features .....	212
9.3.2	Evaluation of the Performance of Deformable Model .....	216
<b>9.4</b>	Conclusion .....	219
	References .....	219
<b>CHAPTER 10</b>	<b>Characterization of Errors in Deep Learning-Based Brain MRI Segmentation .....</b>	<b>223</b>
	Akshay Pai, Yuan-Ching Teng, Joseph Blair, Michiel Kallenberg, Erik B. Dam, Stefan Sommer, Christian Igel and Mads Nielsen	
<b>10.1</b>	Introduction .....	224
<b>10.2</b>	Deep Learning for Segmentation .....	225
<b>10.3</b>	Convolutional Neural Network Architecture .....	226
10.3.1	Basic CNN Architecture .....	226
10.3.2	Tri-Planar CNN for 3D Image Analysis .....	227

<b>10.4</b>	Experiments . . . . .	228
10.4.1	Dataset . . . . .	228
10.4.2	CNN Parameters . . . . .	229
10.4.3	Training . . . . .	230
10.4.4	Estimation of Centroid Distances . . . . .	230
10.4.5	Registration-Based Segmentation . . . . .	231
10.4.6	Characterization of Errors . . . . .	231
<b>10.5</b>	Results . . . . .	233
10.5.1	Overall Performance . . . . .	233
10.5.2	Errors . . . . .	233
<b>10.6</b>	Discussion . . . . .	238
<b>10.7</b>	Conclusion . . . . .	240
	References . . . . .	240
	Notes . . . . .	242

## PART 4 MEDICAL IMAGE REGISTRATION

---

<b>CHAPTER 11</b>	<b>Scalable High Performance Image Registration Framework by Unsupervised Deep Feature Representations Learning . . . . .</b>	245
	Shaoyu Wang, Minjeong Kim, Guorong Wu and Dinggang Shen	
<b>11.1</b>	Introduction . . . . .	246
<b>11.2</b>	Proposed Method . . . . .	248
11.2.1	Related Research . . . . .	248
11.2.2	Learn Intrinsic Feature Representations by Unsupervised Deep Learning . . . . .	250
11.2.3	Registration Using Learned Feature Representations . . . . .	255
<b>11.3</b>	Experiments . . . . .	258
11.3.1	Experimental Result on ADNI Dataset . . . . .	259
11.3.2	Experimental Result on LONI Dataset . . . . .	260
11.3.3	Experimental Result on 7.0-T MR Image Dataset . . . . .	263
<b>11.4</b>	Conclusion . . . . .	265
	References . . . . .	265
<b>CHAPTER 12</b>	<b>Convolutional Neural Networks for Robust and Real-Time 2-D/3-D Registration . . . . .</b>	271
	Shun Miao, Jane Z. Wang and Rui Liao	
<b>12.1</b>	Introduction . . . . .	272
<b>12.2</b>	X-Ray Imaging Model . . . . .	274
<b>12.3</b>	Problem Formulation . . . . .	275

<b>12.4</b>	Regression Strategy .....	276
12.4.1	Parameter Space Partitioning .....	276
12.4.2	Marginal Space Regression .....	277
<b>12.5</b>	Feature Extraction .....	277
12.5.1	Local Image Residual .....	277
12.5.2	3-D Points of Interest .....	279
<b>12.6</b>	Convolutional Neural Network .....	280
12.6.1	Network Structure .....	280
12.6.2	Training Data .....	281
12.6.3	Solver .....	282
<b>12.7</b>	Experiments and Results .....	283
12.7.1	Experiment Setup .....	283
12.7.2	Hardware & Software .....	285
12.7.3	Performance Analysis .....	286
12.7.4	Comparison with State-of-the-Art Methods .....	288
<b>12.8</b>	Discussion .....	292
	Disclaimer .....	294
	References .....	294

## PART 5 COMPUTER-AIDED DIAGNOSIS AND DISEASE QUANTIFICATION

---

<b>CHAPTER 13</b>	Chest Radiograph Pathology Categorization via Transfer Learning .....	299
	Idit Diamant, Yaniv Bar, Ofer Geva, Lior Wolf, Gali Zimmerman, Sivan Lieberman, Eli Konen and Hayit Greenspan	
<b>13.1</b>	Introduction .....	300
<b>13.2</b>	Image Representation Schemes with Classical (Non-Deep) Features .....	303
13.2.1	Classical Filtering .....	304
13.2.2	Bag-of-Visual-Words Model .....	305
<b>13.3</b>	Extracting Deep Features from a Pre-Trained CNN Model ..	306
<b>13.4</b>	Extending the Representation Using Feature Fusion and Selection .....	309
<b>13.5</b>	Experiments and Results .....	309
13.5.1	Data .....	309
13.5.2	Experimental Setup .....	310
13.5.3	Experimental Results .....	310
<b>13.6</b>	Conclusion .....	315
	Acknowledgements .....	317
	References .....	318

<b>CHAPTER 14 Deep Learning Models for Classifying Mammogram Exams Containing Unregistered Multi-View Images and Segmentation Maps of Lesions .....</b>	<b>321</b>
Gustavo Carneiro, Jacinto Nascimento and Andrew P. Bradley	
<b>14.1 Introduction .....</b>	322
<b>14.2 Literature Review .....</b>	324
<b>14.3 Methodology .....</b>	326
14.3.1 Deep Learning Model.....	326
<b>14.4 Materials and Methods .....</b>	329
<b>14.5 Results .....</b>	331
<b>14.6 Discussion .....</b>	334
<b>14.7 Conclusion.....</b>	334
Acknowledgements .....	335
References .....	335
Note .....	339
<b>CHAPTER 15 Randomized Deep Learning Methods for Clinical Trial Enrichment and Design in Alzheimer's Disease .....</b>	<b>341</b>
Vamsi K. Ithapu, Vikas Singh and Sterling C. Johnson	
<b>15.1 Introduction .....</b>	342
<b>15.2 Background .....</b>	344
15.2.1 Clinical Trials and Sample Enrichment .....	344
15.2.2 Neural Networks .....	345
15.2.3 Backpropagation and Deep Learning .....	346
<b>15.3 Optimal Enrichment Criterion .....</b>	350
15.3.1 Ensemble Learning and Randomization .....	351
<b>15.4 Randomized Deep Networks .....</b>	352
15.4.1 Small Sample Regime and Multiple Modalities .....	353
15.4.2 Roadmap .....	354
15.4.3 rDA and rDr Training .....	356
15.4.4 The Disease Markers – rDAm and rDrm .....	358
<b>15.5 Experiments.....</b>	360
15.5.1 Participant Data and Preprocessing .....	360
15.5.2 Evaluations Setup .....	360
15.5.3 Results .....	362
<b>15.6 Discussion .....</b>	368
Acknowledgements .....	374
References .....	374
Notes .....	377

---

## PART 6 OTHERS

<b>CHAPTER 16 Deep Networks and Mutual Information Maximization for Cross-Modal Medical Image Synthesis .....</b>	<b>381</b>
Raviteja Vemulapalli, Hien Van Nguyen and S. Kevin Zhou	
<b>16.1 Introduction .....</b>	<b>382</b>
<b>16.2 Supervised Synthesis Using Location-Sensitive Deep Network .....</b>	<b>384</b>
16.2.1 Backpropagation .....	386
16.2.2 Network Simplification .....	387
16.2.3 Experiments .....	388
<b>16.3 Unsupervised Synthesis Using Mutual Information Maximization .....</b>	<b>390</b>
16.3.1 Generating Multiple Target Modality Candidates .....	392
16.3.2 Full Image Synthesis Using Best Candidates .....	393
16.3.3 Refinement Using Coupled Sparse Representation .....	396
16.3.4 Extension to Supervised Setting .....	396
16.3.5 Experiments .....	397
<b>16.4 Conclusions and Future Work .....</b>	<b>401</b>
Acknowledgements .....	401
References .....	401
Note .....	403
<b>CHAPTER 17 Natural Language Processing for Large-Scale Medical Image Analysis Using Deep Learning .....</b>	<b>405</b>
Hoo-Chang Shin, Le Lu and Ronald M. Summers	
<b>17.1 Introduction .....</b>	<b>406</b>
<b>17.2 Fundamentals of Natural Language Processing .....</b>	<b>407</b>
17.2.1 Pattern Matching .....	407
17.2.2 Topic Modeling .....	410
<b>17.3 Neural Language Models .....</b>	<b>411</b>
17.3.1 Word Embeddings .....	411
17.3.2 Recurrent Language Model .....	412
<b>17.4 Medical Lexicons .....</b>	<b>414</b>
17.4.1 UMLS Metathesaurus .....	414
17.4.2 RadLex .....	414
<b>17.5 Predicting Presence or Absence of Frequent Disease Types .....</b>	<b>414</b>
17.5.1 Mining Presence/Absence of Frequent Disease Terms .....	414
17.5.2 Prediction Result and Discussion .....	415
<b>17.6 Conclusion .....</b>	<b>419</b>

Acknowledgements .....	419
References .....	419
Index .....	423

# Contributors

**R. Venkatesh Babu**

Indian Institute of Science, Bangalore, India

**Yaniv Bar**

Tel-Aviv University, Ramat-Aviv, Israel

**Joseph Blair**

University of Copenhagen, Copenhagen, Denmark

**Andrew P. Bradley**

University of Queensland, Brisbane, QLD, Australia

**Gustavo Carneiro**

University of Adelaide, Adelaide, SA, Australia

**Hao Chen**

The Chinese University of Hong Kong, Hong Kong, China

**Erik B. Dam**

Biomediq A/S, Copenhagen, Denmark

**Idit Diamant**

Tel-Aviv University, Ramat-Aviv, Israel

**Qi Dou**

The Chinese University of Hong Kong, Hong Kong, China

**Michael D. Feldman**

University of Pennsylvania, Philadelphia, PA, United States

**Renee Frank**

University of Pennsylvania, Philadelphia, PA, United States

**Yaozong Gao**

University of North Carolina at Chapel Hill, Chapel Hill, NC, United States

**Bogdan Georgescu**

Siemens Medical Solutions USA, Inc., Princeton, NJ, United States

**Ofer Geva**

Tel-Aviv University, Ramat-Aviv, Israel

**Florin C. Ghesu**

Siemens Medical Solutions USA, Inc., Princeton, NJ, United States;  
Friedrich-Alexander University Erlangen–Nürnberg, Erlangen, Germany

**Hayit Greenspan**

Tel-Aviv University, Ramat-Aviv, Israel

**Yanrong Guo**

University of North Carolina at Chapel Hill, Chapel Hill, NC, United States

**Pheng-Ann Heng**

The Chinese University of Hong Kong, Hong Kong, China

**Joachim Hornegger**

Friedrich-Alexander University Erlangen–Nürnberg, Erlangen, Germany

**R. Todd Hurst**

Mayo Clinic, Scottsdale, AZ, United States

**Christian Igel**

University of Copenhagen, Copenhagen, Denmark

**Vamsi K. Ithapu**

University of Wisconsin–Madison, Madison, WI, United States

**Andrew Janowczyk**

Case Western Reserve University, Cleveland, OH, United States

**Sterling C. Johnson**

William S. Middleton Memorial Hospital, Madison, WI, United States;  
University of Wisconsin–Madison, Madison, WI, United States

**Michiel Kallenberg**

Biomediq A/S, Copenhagen, Denmark

**Christopher B. Kendall**

Mayo Clinic, Scottsdale, AZ, United States

**Minjeong Kim**

University of North Carolina at Chapel Hill, Chapel Hill, NC, United States

**Eli Konen**

Sheba Medical Center, Tel-Hashomer, Israel

**Srinivas S.S. Kruthiventi**

Indian Institute of Science, Bangalore, India

**Jianming Liang**

Arizona State University, Scottsdale, AZ, United States

**Rui Liao**

Siemens Medical Solutions USA, Inc., Princeton, NJ, United States

**Sivan Lieberman**

Sheba Medical Center, Tel-Hashomer, Israel

**Le Lu**

National Institutes of Health Clinical Center, Bethesda, MD, United States

**Anant Madabhushi**

Case Western Reserve University, Cleveland, OH, United States

**Kenneth B. Margulies**

University of Pennsylvania, Philadelphia, PA, United States

**Dimitris Metaxas**

Rutgers University, Piscataway, NJ, United States

**Shun Miao**

Siemens Medical Solutions USA, Inc., Princeton, NJ, United States

**Vincent C.T. Mok**

The Chinese University of Hong Kong, Hong Kong, China

**Konda R. Mopuri**

Indian Institute of Science, Bangalore, India

**Jacinto Nascimento**

Instituto Superior Técnico, Lisbon, Portugal

**Hien Van Nguyen**

Uber Advanced Technology Center, Pittsburgh, PA, United States

**Mads Nielsen**

Biomediq A/S, Copenhagen, Denmark;  
University of Copenhagen, Copenhagen, Denmark

**Jeffrey J. Nirschl**

University of Pennsylvania, Philadelphia, PA, United States

**Akshay Pai**

Biomediq A/S, Copenhagen, Denmark;  
University of Copenhagen, Copenhagen, Denmark

**Eliot G. Peyster**

University of Pennsylvania, Philadelphia, PA, United States

**Nikita Prabhu**

Indian Institute of Science, Bangalore, India

**Jing Qin**

The Hong Kong Polytechnic University, Hong Kong, China

**Ravi K. Sarvadevabhatla**

Indian Institute of Science, Bangalore, India

**Dinggang Shen**

University of North Carolina at Chapel Hill, Chapel Hill, NC, United States

**Lin Shi**

The Chinese University of Hong Kong, Hong Kong, China

**Hoo-Chang Shin**

National Institutes of Health Clinical Center, Bethesda, MD, United States

**Jae Y. Shin**

Arizona State University, Scottsdale, AZ, United States

**Vikas Singh**

University of Wisconsin–Madison, Madison, WI, United States

**Stefan Sommer**

University of Copenhagen, Copenhagen, Denmark

**Suraj Srinivas**

Indian Institute of Science, Bangalore, India

**Heung-II Suk**

Korea University, Seoul, Republic of Korea

**Ronald M. Summers**

National Institutes of Health Clinical Center, Bethesda, MD, United States

**Nima Tajbakhsh**

Arizona State University, Scottsdale, AZ, United States

**Yuan-Ching Teng**

University of Copenhagen, Copenhagen, Denmark

**Raviteja Vemulapalli**

University of Maryland, College Park, MD, United States

**Defeng Wang**

The Chinese University of Hong Kong, Hong Kong, China

**Jane Z. Wang**

University of British Columbia, Vancouver, BC, Canada

**Shaoyu Wang**

University of North Carolina at Chapel Hill, Chapel Hill, NC, United States;  
Donghua University, Shanghai, China

**Lior Wolf**

Tel-Aviv University, Ramat-Aviv, Israel

**Guorong Wu**

University of North Carolina at Chapel Hill, Chapel Hill, NC, United States

**Yuanpu Xie**

University of Florida, Gainesville, FL, United States

**Fuyong Xing**

University of Florida, Gainesville, FL, United States

**Zhennan Yan**

Rutgers University, Piscataway, NJ, United States

**Lin Yang**

University of Florida, Gainesville, FL, United States

**Lequan Yu**

The Chinese University of Hong Kong, Hong Kong, China

**Yiqiang Zhan**

Siemens Healthcare, Malvern, PA, United States

**Shaoting Zhang**

University of North Carolina at Charlotte, Charlotte, NC, United States

**Lei Zhao**

The Chinese University of Hong Kong, Hong Kong, China

**S. Kevin Zhou**

Siemens Healthineers Technology Center, Princeton, NJ, United States

**Xiang Sean Zhou**

Siemens Healthcare, Malvern, PA, United States

**Gali Zimmerman**

Tel-Aviv University, Ramat-Aviv, Israel

This page intentionally left blank

# About the Editors

**S. Kevin Zhou**, PhD, is currently a Principal Key Expert at Siemens Healthineers Technology Center, leading a team of full-time research scientists and students dedicated to researching and developing innovative solutions for medical and industrial imaging products. His research interests lie in computer vision and machine/deep learning and their applications to medical image analysis, face recognition, and modeling, etc. He has published over 150 book chapters and peer-reviewed journal and conference papers, registered over 250 patents and inventions, written two research monographs, and edited three books. He has won multiple technology, patent, and product awards, including the R&D 100 Award and Siemens Inventor of the Year. He is an editorial board member for the Medical Image Analysis and IEEE Transactions on Medical Imaging journals and a fellow of the American Institute of Medical and Biological Engineering (AIMBE).

**Hayit Greenspan** is a Professor at the Biomedical Engineering Department, Faculty of Engineering, Tel Aviv University. She was a visiting Professor at the Radiology Department of Stanford University, and is currently affiliated with the International Computer Science Institute (ICSI) at Berkeley. Dr. Greenspan's research focuses on image modeling and analysis, deep learning, and content-based image retrieval. Research projects include brain MRI research (structural and DTI), CT and X-ray image analysis – automated detection to segmentation and characterization. Dr. Greenspan has over 150 publications in leading international journals and conference proceedings. She has received several awards and is a coauthor on several patents. Dr. Greenspan is a member of several journal and conference program committees, including SPIE Medical Imaging, IEEE\_ISBI, and MICCAI. She is an Associate Editor for the IEEE Transactions on Medical Imaging (TMI) journal. Recently, in May 2016, she was the lead guest editor for an IEEE-TMI special issue on "Deep Learning in Medical Imaging."

**Dinggang Shen** is a Professor of Radiology at the Biomedical Research Imaging Center (BRIC), and Computer Science, and Biomedical Engineering Departments in the University of North Carolina at Chapel Hill (UNC-CH). He is currently directing the Center for Image Informatics and Analysis, the Image Display, Enhancement, and Analysis (IDEA) Lab in the Department of Radiology, and also the medical image analysis core in the BRIC. He was a tenure-track assistant professor at the University of Pennsylvania (UPenn), and a faculty member at Johns Hopkins University. Dr. Shen's research interests include medical image analysis, computer vision, and pattern recognition. He has published more than 700 papers in international journals and conference proceedings. He serves as an editorial board member for six international journals. He has served on the Board of Directors of The Medical Image Computing and Computer Assisted Intervention (MICCAI) Society in 2012–2015.

This page intentionally left blank

# Foreword

Computational Medical Image Analysis has become a prominent field of research at the intersection of Informatics, Computational Sciences, and Medicine, supported by a vibrant community of researchers working in academics, industry, and clinical centers.

During the past few years, Machine Learning methods have brought a revolution to the Computer Vision community, introducing novel efficient solutions to many image analysis problems that had long remained unsolved. For this revolution to enter the field of Medical Image Analysis, dedicated methods must be designed which take into account the specificity of medical images.

Indeed, medical images capture the anatomy and physiology of patients through the measurements of geometrical, biophysical, and biochemical properties of their living tissues. These images are acquired with algorithms that exploit complex medical imaging processes whose principles must be well understood as well as those governing the complex structures and functions of the human body.

The book *Deep Learning for Medical Image Analysis* edited by S. Kevin Zhou, Hayit Greenspan, and Dinggang Shen, top-notch researchers from both academia and industry in designing machine learning methods for medical image analysis, covers state-of-the-art reviews of deep learning approaches for medical image analysis, including medical image detection/recognition, medical image segmentation, medical image registration, computer aided diagnosis and disease quantification, to name some of the most important addressed problems. The book, which starts with an introduction to Convolutional Neural Networks for Computer Vision presents a set of novel deep learning methods applied to a variety of clinical problems and imaging modalities operating at various scales, including X-ray radiographies, Magnetic Resonance Imaging, Computed Tomography, microscopic imaging, ultrasound imaging, etc.

This impressive collection of excellent contributions will definitely serve and inspire all the researchers interested in the development of new machine learning methods in the rapidly evolving field of medical image analysis.

Nicholas Ayache, PhD  
*Inria, Sophia Antipolis, France*  
September 1, 2016

This page intentionally left blank

PART

Introduction

1

This page intentionally left blank

# An Introduction to Neural Networks and Deep Learning

# 1

Heung-Il Suk

*Korea University, Seoul, Republic of Korea*

## CHAPTER OUTLINE

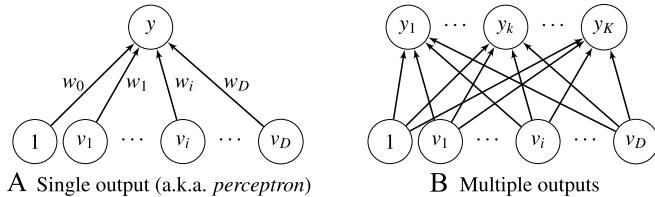
---

1.1	<b>Introduction</b>	3
1.2	<b>Feed-Forward Neural Networks</b>	4
1.2.1	Perceptron	4
1.2.2	Multi-Layer Neural Network	5
1.2.3	Learning in Feed-Forward Neural Networks	6
1.3	<b>Convolutional Neural Networks</b>	8
1.3.1	Convolution and Pooling Layer	8
1.3.2	Computing Gradients	9
1.4	<b>Deep Models</b>	11
1.4.1	Vanishing Gradient Problem	11
1.4.2	Deep Neural Networks	12
1.4.2.1	Auto-Encoder	12
1.4.2.2	Stacked Auto-Encoder	13
1.4.3	Deep Generative Models	14
1.4.3.1	Restricted Boltzmann Machine	15
1.4.3.2	Deep Belief Network	17
1.4.3.3	Deep Boltzmann Machine	18
1.5	<b>Tricks for Better Learning</b>	20
1.5.1	Rectified Linear Unit (ReLU)	20
1.5.2	Dropout	20
1.5.3	Batch Normalization	21
1.6	<b>Open-Source Tools for Deep Learning</b>	22
	<b>References</b>	22
	<b>Notes</b>	24

---

## 1.1 INTRODUCTION

A brain or biological neural network is considered as the most well-organized system that processes information from different senses such as sight, hearing, touch, taste, and smell in an efficient and intelligent manner. One of the key mechanisms for

**FIGURE 1.1**

An architecture of a single-layer neural network.

information processing in a human brain is that the complicated high-level information is processed by means of the collaboration, i.e., connections (called synapses), of a large number of the structurally simple elements (called neurons). In machine learning, artificial neural networks are a family of models that mimic the structural elegance of the neural system and learn patterns inherent in observations.

## 1.2 FEED-FORWARD NEURAL NETWORKS

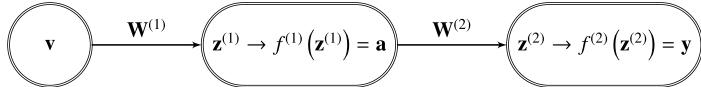
This section introduces neural networks that process information in a feed-forward manner. Throughout the chapter, matrices and vectors are denoted as boldface uppercase letters and boldface lowercase letters, respectively, and scalars are denoted as normal italic letters. For a transpose operator, a superscript  $\top$  is used.

### 1.2.1 PERCEPTRON

The simplest learnable artificial neural model, known as *perceptron* [1], is structured with input visible units  $\{v_i\}_{i=1}^D$ , trainable connection weights  $\{w_i\}_{i=1}^D$ , and a bias  $w_0$ , and an output unit  $y$  as shown in Fig. 1.1A. Since the perceptron model has a single layer of an output unit, not counting the input visible layer, it is also called a single-layer neural network. Given an observation<sup>1</sup> or datum  $\mathbf{v} \in \mathbb{R}^D$ , the value of the output unit  $y$  is obtained from an activation function  $f(\cdot)$  by taking the weighted sum of the inputs as follows:

$$y(\mathbf{v}; \Theta) = f \left( \sum_{i=1}^D v_i w_i + w_0 \right) = f \left( \mathbf{w}^\top \mathbf{v} + w_0 \right) \quad (1.1)$$

where  $\Theta = \{\mathbf{w}, w_0\}$  denotes a parameter set,  $\mathbf{w} = [w_i]_{i=1}^D \in \mathbb{R}^D$  is a connection weight vector, and  $w_0$  is a bias. Let us introduce a pre-activation variable  $z$  that is determined by the weighted sum of the inputs, i.e.,  $z = \mathbf{w}^\top \mathbf{v} + w_0$ . As for the activation function  $f(\cdot)$ , a ‘logistic sigmoid’ function, i.e.,  $\sigma(z) = \frac{1}{1+\exp(-z)}$ , is commonly used for a binary classification task.

**FIGURE 1.2**

An architecture of a two-layer neural network in simplified representation.

Regarding a multi-output task, e.g., multi-class classification or multi-output regression, it is straightforward to extend the perceptron model by adding multiple output units  $\{y_k\}_{k=1}^K$  (Fig. 1.1B), one for each class, with their respective connection weights  $\{W_{ki}\}_{i=1,\dots,D; k=1,\dots,K}$  as follows:

$$y_k(\mathbf{v}; \Theta) = f \left( \sum_{i=1}^D v_i W_{ki} + w_{k0} \right) = f \left( \mathbf{w}_k^\top \mathbf{v} + w_{k0} \right) \quad (1.2)$$

where  $\Theta = \{\mathbf{W} \in \mathbb{R}^{K \times D}\}$ ,  $W_{ki}$  denotes a connection weight from  $v_i$  to  $y_k$ . As for the activation function, it is common to use a ‘softmax’ function  $s(z_k) = \frac{\exp(z_k)}{\sum_{l=1}^K \exp(z_l)}$  for multi-class classification, where the output values can be interpreted as probability.

### 1.2.2 MULTI-LAYER NEURAL NETWORK

One of the main limitations of the single-layer neural network comes from its linear separation for a classification task, despite the use of nonlinear activation function. This limitation can be circumvented by introducing a so-called ‘hidden’ layer between the input layer and the output layer as shown in Fig. 1.2, where the double circles denote a vectorization of the units in the respective layers for simplification. Note that in Fig. 1.2, there can exist multiple units in each layer and units of the neighboring layers are fully connected to each other, but no connections in the same layer. For a two-layer neural network, which is also known as *multi-layer perceptron*, we can write its composition function as follows:

$$y_k(\mathbf{v}; \Theta) = f^{(2)} \left( \sum_{j=1}^M W_{kj}^{(2)} f^{(1)} \left( \sum_{i=1}^D W_{ji}^{(1)} v_i \right) \right) \quad (1.3)$$

where the superscript denotes a layer index,  $M$  is the number of hidden units, and  $\Theta = \{\mathbf{W}^{(1)} \in \mathbb{R}^{M \times D}, \mathbf{W}^{(2)} \in \mathbb{R}^{K \times M}\}$ . Hereafter, the bias term is omitted for simplicity. It is possible to add a number of hidden layers ( $L - 1$ ) and the corresponding estimation function is defined as

$$y_k = f^{(L)} \left( \sum_l W_{kl}^{(L)} f^{(L-1)} \left( \sum_m W_{lm} f^{(L-2)} \left( \dots f^{(1)} \left( \sum_i W_{ji}^{(1)} x_i \right) \right) \right) \right). \quad (1.4)$$

While, in theory, it is possible to apply different types of activation functions for different layers or even different units, it is common to apply the same type of an activation function for the hidden layers in the literature. However, it should be a nonlinear function; otherwise, the function will be represented by a single-layer neural network with a weight matrix, equal to the resulting matrix of multiplying weight matrices of hidden layers. In convention, the activation function  $f(\cdot)$  is commonly defined with a sigmoidal function such as a ‘*logistic sigmoid*’ function, i.e.,  $\sigma(z) = \frac{1}{1+\exp[-z]}$ , or a ‘*hyperbolic tangent*’ function, i.e.,  $\tanh(z) = \frac{\exp[z]-\exp[-z]}{\exp[z]+\exp[-z]}$ , due to their nonlinear and differentiable characteristics. Their difference lies in the range of output values squashed from real values to the range of  $[0, 1]$  for logistic sigmoid function and  $[-1, 1]$  for hyperbolic tangent function.

### 1.2.3 LEARNING IN FEED-FORWARD NEURAL NETWORKS

In terms of network learning, there are two fundamental problems, namely, network architecture learning and network parameters learning. While the network architecture learning still remains an open question,<sup>2</sup> there exists an efficient algorithm for network parameters learning as circumstantiated below.

The problem of learning parameters of a feed-forward neural network can be formulated as error function minimization. Given a training data set  $\{\mathbf{x}_n, \mathbf{t}_n\}_{n=1}^N$ , where  $\mathbf{x}_n \in \mathbb{R}^D$  denotes an observation and  $\mathbf{t}_n \in \{0, 1\}^K$  denotes a class indicator vector with one-of- $K$  encoding, i.e., for a class  $k$ , only the  $k$ th element in a vector  $\mathbf{t}_n$  is 1 and all the other elements are 0. For  $K$ -class classification, it is common to use a cross-entropy cost function defined as follows:

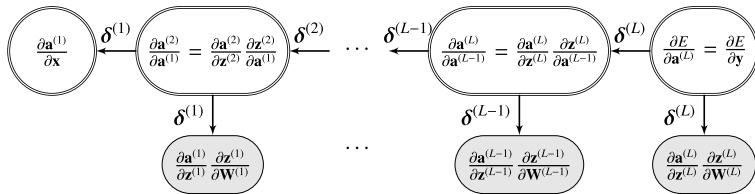
$$E(\Theta) = -\frac{1}{2} \sum_{n=1}^N \sum_{k=1}^K t_{nk} \ln y_{nk} \quad (1.5)$$

where  $t_{kn}$  denotes the  $k$ th element of the target vector  $\mathbf{t}_n$  and  $y_{kn}$  is the  $k$ th element of the prediction vector  $\mathbf{y}_n$  for  $\mathbf{x}_n$ . For a certain parameter set  $\Theta$ , the prediction vector  $\mathbf{y}_n$  from an  $L$ -layer neural network can be obtained by Eq. (1.4).

The error function in Eq. (1.5) is highly nonlinear and non-convex. Thus, there is no analytic solution of the parameter set  $\Theta$  that minimizes Eq. (1.5). Instead, we resort to a gradient descent algorithm by updating the parameters iteratively. To utilize a gradient descent algorithm, one requires a way to compute a gradient  $\nabla E(\Theta)$  evaluated at the parameter set  $\Theta$ .

For a feed-forward neural network, the gradient can be efficiently evaluated by means of error backpropagation [2]. The key idea of backpropagation algorithm is to propagate errors from the output layer back to the input layer by a chain rule. Specifically, in an  $L$ -layer neural network, the derivative of an error function  $E$  with respect to the parameters for  $l$ th layer, i.e.,  $\mathbf{W}^{(l)}$ , can be estimated as follows:

$$\frac{\partial E}{\partial \mathbf{W}^{(l)}} = \frac{\partial E}{\partial \mathbf{a}^{(L)}} \frac{\partial \mathbf{a}^{(L)}}{\partial \mathbf{a}^{(L-1)}} \cdots \frac{\partial \mathbf{a}^{(l+2)}}{\partial \mathbf{a}^{(l+1)}} \frac{\partial \mathbf{a}^{(l+1)}}{\partial \mathbf{a}^{(l)}} \frac{\partial \mathbf{a}^{(l)}}{\partial \mathbf{z}^{(l)}} \frac{\partial \mathbf{z}^{(l)}}{\partial \mathbf{W}^{(l)}} \quad (1.6)$$

**FIGURE 1.3**

Graphical representation of a backpropagation algorithm in an  $L$ -layer network.

where  $\mathbf{z}^{(l)}$  and  $\mathbf{a}^{(l)}$  respectively denote the pre-activation vector and the activation vector of the layer  $l$  and  $\mathbf{a}^{(L)} = \mathbf{y}$ . Note that  $\frac{\partial E}{\partial \mathbf{a}^{(L)}}$ , or equivalently  $\frac{\partial E}{\partial \mathbf{y}}$ , corresponds to the error computed at the output layer. For the estimation of the gradient of an error function  $E$  with respect to the parameter  $\mathbf{W}^{(l)}$ , it utilizes the error propagated from the output layer through the chains in the form of  $\frac{\partial \mathbf{a}^{(k+1)}}{\partial \mathbf{a}^{(k)}}$ ,  $k = l, l + 1, \dots, L - 1$ , along with  $\frac{\partial \mathbf{a}^{(l)}}{\partial \mathbf{z}^{(l)}} \frac{\partial \mathbf{z}^{(l)}}{\partial \mathbf{W}^{(l)}}$ . The fraction  $\frac{\partial \mathbf{a}^{(k+1)}}{\partial \mathbf{a}^{(k)}}$  can be also computed in a similar way as

$$\frac{\partial \mathbf{a}^{(l+1)}}{\partial \mathbf{a}^{(l)}} = \frac{\partial \mathbf{a}^{(l+1)}}{\partial \mathbf{z}^{(l+1)}} \frac{\partial \mathbf{z}^{(l+1)}}{\partial \mathbf{a}^{(l)}} \quad (1.7)$$

$$= f'(\mathbf{z}^{(l)}) (\mathbf{W}^{(l+1)})^\top \quad (1.8)$$

where  $f'(\mathbf{z}^{(l)})$  denotes a gradient of an activation function  $f^{(l)}$  with respect to the pre-activation vector  $\mathbf{z}^{(l)}$ .

**Fig. 1.3** illustrates a graphical representation of estimating the gradients of an error function with respect to parameters in an  $L$ -layer neural network based on Eq. (1.6) and Eq. (1.7). Basically, it depicts the error propagation mechanism in a backpropagation algorithm. Specifically, computations are performed at each node by passing error messages,  $\delta^{(l)}$ , through arrows from tail (source) to head (destination), starting from the rightmost double circle node. When a double circle node of the layer  $l$  receives an error message  $\delta^{(l+1)}$  from the double circle node of the layer  $l + 1$ , it first updates the error message  $\delta^{(l)}$  as

$$\delta^{(l)} = \frac{\partial \mathbf{a}^{(l+1)}}{\partial \mathbf{z}^{(l+1)}} \odot \left\{ \frac{\partial \mathbf{z}^{(l+1)}}{\partial \mathbf{a}^{(l)}} \cdot \delta^{(l+1)} \right\} \quad (1.9)$$

$$= f'(\mathbf{z}^{(l)}) \odot \left\{ (\mathbf{W}^{(l+1)})^\top \cdot \delta^{(l+1)} \right\} \quad (1.10)$$

where  $\odot$  denotes an element-wise multiplication. After updating the error message, the double circle node of the layer  $l$  sends the error message to the double circle node of the layer  $l - 1$  and also the shaded node directly connected for computing  $\frac{\partial E}{\partial \mathbf{W}^{(l)}}$ . It should be noted that the double circle nodes can send their message to the neighboring nodes, once they receive an error message from its source node.

Once we obtain the gradient vector of all the layers, i.e., the shaded nodes at the bottom of the graph in Fig. 1.3, the parameter set  $\mathbf{W} = [\mathbf{W}^{(1)} \dots \mathbf{W}^{(l)} \dots \mathbf{W}^{(L)}]$  is updated as follows:

$$\mathbf{W}^{(\tau+1)} = \mathbf{W}^{(\tau)} - \eta \nabla E (\mathbf{W}^{(\tau)}) \quad (1.11)$$

where  $\nabla E (\mathbf{W}) = \left[ \frac{\partial E}{\partial \mathbf{W}^{(1)}} \dots \frac{\partial E}{\partial \mathbf{W}^{(l)}} \dots \frac{\partial E}{\partial \mathbf{W}^{(L)}} \right]$  is obtained via backpropagation,  $\eta$  is a learning rate, and  $\tau$  denotes an iteration index. The update process repeats until convergence or the predefined number of iterations is reached.

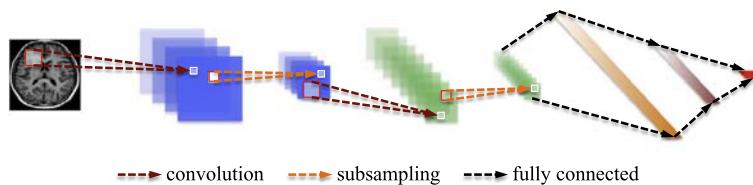
As for the parameter update in Eq. (1.11), there are two different approaches depending on the timing of parameter update, namely, batch gradient descent and stochastic gradient descent. The batch gradient descent updates the parameters based on the gradients  $\nabla E$  evaluated over the whole training samples. Meanwhile, the stochastic gradient descent sequentially updates weight parameters by computing gradient on the basis of one sample at a time. When it comes to large scale learning such as deep learning, it is advocated to apply stochastic gradient descent [3]. As a trade-off between batch gradient and stochastic gradient, a mini-batch gradient descent method, which computes and updates the parameters on the basis of a small set of samples, is commonly used in the literature [4].

## 1.3 CONVOLUTIONAL NEURAL NETWORKS

In conventional multi-layer neural networks, the inputs are always in vector form. However, for (medical) images, the structural or configural information among neighboring pixels or voxels is another source of information. Hence, vectorization inevitably destroys such structural and configural information in images. A Convolutional Neural Network (CNN) that typically has convolutional layers interspersed with pooling (or sub-sampling) layers and then followed by fully connected layers as in a standard multi-layer neural network (Fig. 1.4) is designed to better utilize such spatial and configuration information by taking 2D or 3D images as input. Unlike the conventional multi-layer neural networks, a CNN exploits extensive weight-sharing to reduce the degrees of freedom of models. A pooling layer helps reduce computation time and gradually build up spatial and configural invariance.

### 1.3.1 CONVOLUTION AND POOLING LAYER

The role of a convolution layer is to detect local features at different positions in the input feature maps with learnable kernels  $k_{ij}^{(l)}$ , i.e., connection weights between the feature map  $i$  at the layer  $l - 1$  and the feature map  $j$  at the layer  $l$ . Specifically, the units of the convolution layer  $l$  compute their activations  $A_j^{(l)}$  based only on a spatially contiguous subset of units in the feature maps  $A_i^{(l-1)}$  of the preceding layer

**FIGURE 1.4**

An architecture of a convolutional neural network.

$l - 1$  by convolving the kernels  $k_{ij}^{(l)}$  as follows:

$$\mathbf{A}_j^{(l)} = f \left( \sum_{i=1}^{M^{(l-1)}} \mathbf{A}_i^{(l-1)} * k_{ij}^{(l)} + b_j^{(l)} \right) \quad (1.12)$$

where  $M^{(l-1)}$  denotes the number of feature maps in the layer  $l - 1$ ,  $*$  denotes a convolution operator,  $b_j^{(l)}$  is a bias parameter, and  $f(\cdot)$  is a nonlinear activation function. Due to the local connectivity and weight sharing, we can greatly reduce the number of parameters compared to a fully connected neural network, and thus it is possible to avoid overfitting. Further, when the input image is shifted, the activation of the units in the feature maps are also shifted by the same amount, which allows a CNN to be equivariant to small shifts, as illustrated in Fig. 1.5. In the figure, when the pixel values in the input image are shifted by one-pixel right and one-pixel down, the outputs after convolution are also shifted by one-pixel right and one-pixel down.

A pooling layer follows a convolution layer to downsample the feature maps of the preceding convolution layer. Specifically, each feature map in a pooling layer is linked with a feature map in the convolution layer, and each unit in a feature map of the pooling layer is computed based on a subset of units in its receptive field. Similar to the convolution layer, the receptive field that finds a maximal value among the units in its receptive field is convolved with the convolution map but with a stride of the size of the receptive field so that the contiguous receptive fields are not overlapped commonly. The role of the pooling layer is to progressively reduce the spatial size of the feature maps, and thus reduce the number of parameters and computation involved in the network. Another important function of the pooling layer is for translation invariance over small spatial shifts in the input. In Fig. 1.5, while the bottom leftmost image is a translated version of the top leftmost image by one-pixel right and one-pixel down, their outputs after convolution and pooling operations are the same, especially for the units in green.

### 1.3.2 COMPUTING GRADIENTS

Assume that a convolution layer is followed by a pooling layer. In such a case, units in a feature map of a convolution layer  $l$  are connected to a single unit of the corre-

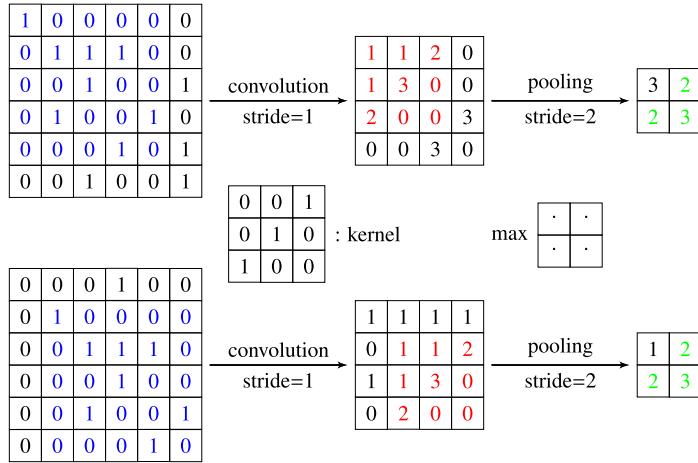
**FIGURE 1.5**

Illustration of translation invariance in convolutional neural network. The bottom leftmost input is a translated version of the upper leftmost input image by one-pixel right and one-pixel down.

sponding feature map in the pooling layer  $l + 1$ . By up-sampling the feature maps of the pooling layer to recover the reduced size of maps, all we need to do is multiply with the derivative of activation function evaluated at the convolution layer's pre-activations  $\mathbf{Z}_j^{(l)}$  as follows:

$$\Delta_j^{(l)} = f'(\mathbf{Z}_j^{(l)}) \odot \text{up}(\Delta_j^{(l+1)}) \quad (1.13)$$

where  $\text{up}(\cdot)$  denotes an up-sampling operation.

For the case when a current layer, whether it is a pooling layer or a convolution layer, is followed by a convolution layer, we must figure out which patch in the current layer's feature map corresponds to a unit in the next layer's feature map. The kernel weights multiplying the connections between the input patch and the output unit are exactly the weights of the convolutional kernel. The gradients for the kernel weights are computed by the chain rule similar to backpropagation. However, since the same weights are now shared across many connections, we need to sum the gradients for a given weight over all the connections using the kernel weights as follows:

$$\frac{\partial E}{\partial k_{ij}^{(l)}} = \sum_{u,v} \Delta_{j;(u,v)}^{(l)} \mathbf{P}_{i;(u,v)}^{(l-1)} \quad (1.14)$$

where  $\mathbf{P}_{i;(u,v)}^{(l-1)}$  denotes the patch in the  $i$ th feature map of the layer  $l - 1$ , i.e.,  $\mathbf{A}_i^{(l-1)}$ , which was multiplied by  $k_{ij}^{(l)}$  during convolution to compute the element at  $(u, v)$  in the output feature map  $\mathbf{A}_j^{(l)}$ .

## 1.4 DEEP MODELS

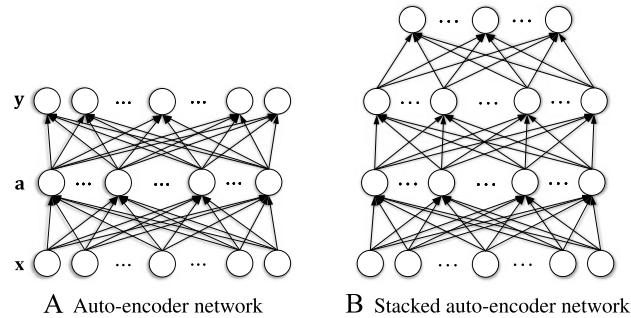
Under a mild assumption on the activation function, a two-layer neural network with a finite number of hidden units can approximate any continuous function [5]. Thus, it is regarded as a universal approximator. However, it is also possible to approximate complex functions to the same accuracy using a ‘deep’ architecture (i.e., multiple layers) with much fewer number of neurons in total. Hence, we can reduce the number of weight parameters, thus allowing us to train with a relatively small-sized dataset [6]. More importantly, compared to shallow architecture that needs a ‘good’ feature extractor, mostly designed by engineering skills or domain expert knowledge, deep models are good at discovering good features automatically in a hierarchical manner, i.e., fine-to-abstract [7].

### 1.4.1 VANISHING GRADIENT PROBLEM

While there have been earlier attempts to deploy deep neural networks, the lack of training samples and the limited computational power prohibited them from successful use for applications. Recently, thanks to the huge amount of data available online and the advances in hardware such as multi-core Central Processing Units (CPUs) and Graphics Processing Units (GPUs), those difficulties have been resolved.

However, so-called ‘vanishing gradient’ problem has remained as one of the main challenges in deep learning. That is, for a deep network, the backpropagated error messages  $\delta^{(l)}$  in Fig. 1.3 become ineffective due to vanishing gradients after repeated multiplications. In this regard, the gradients tend to get smaller and smaller as they propagate backward through the hidden layers and units in the lower layers (close to the input layer) learn much more slowly than units in the higher layers (close to the output layer).

Regarding the vanishing gradient, Glorot and Bengio [8] pointed out that the use of logistic sigmoid activation function caused the derivatives of units with respect to connection weight parameters to saturate near 0 early in training, and hence substantially slowed down learning speed. As a potential remedy to this problem, they suggested alternative activation functions such as a hyperbolic tangent function and a soft sign function with ways to initialize weights [8]. In the meantime, Hinton and Salakhutdinov devised a greedy layer-wise pretraining technique [9] that made a breakthrough of sorts to the vanishing gradient problem. From a learning perspective, in conventional network training, we randomly initialize the connection weights and then learn the whole weight parameters jointly by means of gradient-descent methods along with backpropagation algorithm for gradients computation in a super-

**FIGURE 1.6**

A graphical illustration of an auto-encoder and a stacked auto-encoder.

vised manner. The idea of pretraining is that instead of directly adjusting the whole weight parameters from random initial values, it is beneficial to train the connection weights of the hidden layers in an unsupervised and layer-wise fashion first, and then fine-tune the connection weights jointly. This pretraining technique will be described with further details below in a stacked auto-encoder.

### 1.4.2 DEEP NEURAL NETWORKS

Here, we introduce a deep neural network that constructs a deep architecture by taking auto-encoders as building blocks for a hierarchical feature representation.

#### 1.4.2.1 Auto-Encoder

An auto-encoder, also called as auto-associator, is a special type of a two-layer neural network, composed of an input layer, a hidden layer, and an output layer. The input layer is fully connected to the hidden layer, which is further fully connected to the output layer as illustrated in Fig. 1.6A. The aim of an auto-encoder is to learn a latent or compressed representation of an input, by minimizing the reconstruction error between the input and the reconstructed values from the learned representation.

Let  $D_H$  and  $D_I$  denote, respectively, the number of hidden units and the number of input units in a neural network. For an input  $\mathbf{x} \in \mathbb{R}^{D_I}$ , an auto-encoder maps it to a latent representation  $\mathbf{a} \in \mathbb{R}^{D_H}$  through a linear mapping and then a nonlinear transformation with a nonlinear activation function  $f$  as follows:

$$\mathbf{a} = f(\mathbf{W}^{(1)}\mathbf{x} + \mathbf{b}^{(1)}) \quad (1.15)$$

where  $\mathbf{W}^{(1)} \in \mathbb{R}^{D_H \times D_I}$  is an encoding weight matrix and  $\mathbf{b}^{(1)} \in \mathbb{R}^{D_H}$  is a bias vector. The representation  $\mathbf{a}$  of the hidden layer is then mapped back to a vector  $\mathbf{y} \in \mathbb{R}^{D_I}$ , which approximately reconstructs the input vector  $\mathbf{x}$  by another mapping as follows:

$$\mathbf{y} = \mathbf{W}^{(2)}\mathbf{a} + \mathbf{b}^{(2)} \approx \mathbf{x} \quad (1.16)$$

where  $\mathbf{W}^{(2)} \in \mathbb{R}^{D_I \times D_H}$  and  $\mathbf{b}^{(2)} \in \mathbb{R}^{D_I}$  are a decoding weight matrix and a bias vector, respectively. Structurally, the number of input units and the number of output units are determined by the dimension of an input vector. Meanwhile, the number of hidden units can be determined based on the nature of the data. If the number of hidden units is less than the dimension of the input data, then the auto-encoder can be used for dimensionality reduction. However, it is worth noting that to obtain complicated nonlinear relations among input features, it is possible to allow the number of hidden units to be even larger than the input dimension, from which we can still find an interesting structure by imposing a sparsity constraint [10,11].

From a learning perspective, the goal of an auto-encoder is to minimize the reconstruction error between the input  $\mathbf{x}$  and the output  $\mathbf{y}$  with respect to the parameters. Given a training set  $\{\mathbf{X}, \mathbf{Y}\} = \{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^N$ , let  $E(\mathbf{X}, \mathbf{Y}) = \frac{1}{2} \sum_{i=1}^N \|\mathbf{x}_i - \mathbf{y}_i\|_2^2$  denote a reconstruction error over training samples. To encourage sparseness of the hidden units, it is common to use Kullback–Leibler (KL) divergence to measure the difference between the average activation  $\hat{\rho}_j$  of the  $j$ th hidden unit over the training samples and the target average activation  $\rho$  defined as [12]:

$$\text{KL}(\rho \parallel \hat{\rho}_j) = \rho \log \frac{\rho}{\hat{\rho}_j} + (1 - \rho) \log \frac{1 - \rho}{1 - \hat{\rho}_j}. \quad (1.17)$$

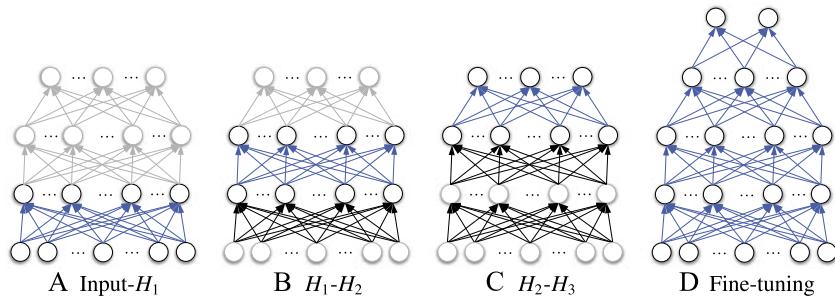
Then our objective function can be written as:

$$E(\mathbf{X}, \mathbf{Z}) + \gamma \sum_j^{D_H} \text{KL}(\rho \parallel \hat{\rho}_j) \quad (1.18)$$

where  $\gamma$  denotes a sparsity control parameter. With the introduction of the KL divergence, the error function is penalized by a large average activation of a hidden unit over the training samples by setting  $\rho$  to be small. This penalization drives the activation of many hidden units to be equal or close to zero by making sparse connections between layers.

#### 1.4.2.2 Stacked Auto-Encoder

Note that the outputs of units in the hidden layer become the latent representation of the input vector. However, due to its simple shallow structural characteristic, the representational power of a single-layer auto-encoder is known to be very limited. But when stacking with multiple auto-encoders by taking the activation values of hidden units of an auto-encoder as the input to the following upper auto-encoder, and thus building an SAE, it is possible to greatly improve the representational power [13]. Thanks to the hierarchical structure, one of the most important characteristics of the SAE is to learn or discover highly nonlinear and complicated patterns such as the relations among input features. When an input vector is presented to an SAE, the different layers of the network represent different levels of information. That is, the lower the layer in the network, the simpler the patterns that are learned; the higher the layer, the more complicated or abstract patterns inherent in the input feature vector.

**FIGURE 1.7**

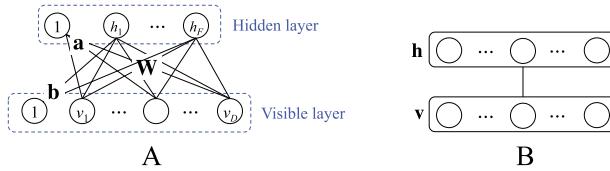
Greedy layer-wise pretraining and fine-tuning of the whole network. ( $H_i$  denotes the  $i$ th hidden layer in the network.)

With regard to training parameters of the weight matrices and the biases in an SAE, a straightforward way is to apply backpropagation with the gradient-based optimization technique starting from random initialization by regarding the SAE as a conventional multi-layer neural network. Unfortunately, it is generally known that deep networks trained in that manner perform worse than networks with a shallow architecture, suffering from falling into a poor local optimum [11]. To circumvent this problem, it is good to consider a greedy layer-wise learning [9]. The key idea in a greedy layer-wise learning is to train one layer at a time by maximizing the variational lower bound. That is, we first train the first hidden layer with the training data as input, and then train the second hidden layer with the outputs from the first hidden layer as input, and so on. That is, the representation of the  $l$ th hidden layer is used as input for the  $(l + 1)$ th hidden layer. This greedy layer-wise learning is performed as ‘*pretraining*’ (Figs. 1.7A–1.7C). The important feature of pretraining is that it is conducted in an unsupervised manner with a standard backpropagation algorithm [14].

When it comes to a classification problem, we stack another output layer on top of the SAE (Fig. 1.7D) with an appropriate activation function. This top output layer is used to represent the class-label of an input sample. Then by taking the pretrained connection weights as the initial parameters for the hidden units and randomly initializing the connection weights between the top hidden layer and the output layer, it is possible to train the whole parameters jointly in a supervised manner by gradient descent with a backpropagation algorithm. Note that the initialization of the parameters via pretraining helps the supervised optimization, called ‘*fine-tuning*’, reduce the risk of falling into local poor optima [9,11].

### 1.4.3 DEEP GENERATIVE MODELS

Unlike the deep neural networks, deep generative models draw samples from the trained model, which thus allows checking qualitatively what has been learned by these models, for instance, by visualizing images that the model has learned. Here,

**FIGURE 1.8**

An architecture of a restricted Boltzmann machine (A) and its simplified illustration (B).

we introduce two deep generative models, namely, Deep Belief Network (DBN) and Deep Boltzmann Machine (DBM), which use restricted Boltzmann machines (RBMs) as basic building blocks for their construction.

#### 1.4.3.1 Restricted Boltzmann Machine

An RBM is a two-layer undirected graphical model with visible and hidden units in each layer (Fig. 1.8). It assumes a symmetric connectivity  $\mathbf{W}$  between the visible layer and the hidden layer, but no connections within the layers, and each layer has a bias term,  $\mathbf{a}$  and  $\mathbf{b}$ , respectively. In Fig. 1.8, the units of the visible layer  $\mathbf{v} \in \mathbb{R}^D$  correspond to observations while the units of the hidden layer  $\mathbf{h} \in \mathbb{R}^F$  models the structures or dependencies over visible units, where  $D$  and  $F$  denote the numbers of visible and hidden units, respectively. Note that because of the symmetry of the weight matrix  $\mathbf{W}$ , it is possible to reconstruct the input observations from the hidden representations. Therefore, an RBM is naturally regarded as an auto-encoder [9] and this favorable characteristic is used in RBM parameters learning [9].

In RBM, a joint probability of  $(\mathbf{v}, \mathbf{h})$  is given by

$$P(\mathbf{v}, \mathbf{h}; \Theta) = \frac{1}{Z(\Theta)} \exp [-E(\mathbf{v}, \mathbf{h}; \Theta)] \quad (1.19)$$

where  $\Theta = \{\mathbf{W} \in \mathbb{R}^{D \times F}, \mathbf{a} \in \mathbb{R}^D, \mathbf{b} \in \mathbb{R}^F\}$ ,  $E(\mathbf{v}, \mathbf{h}; \Theta)$  is an energy function, and  $Z(\Theta)$  is a partition function that can be obtained by summing over all possible pairs of  $\mathbf{v}$  and  $\mathbf{h}$ . For the sake of simplicity, by assuming binary visible and hidden units, which are the commonly studied case, the energy function  $E(\mathbf{v}, \mathbf{h}; \Theta)$  is defined as

$$\begin{aligned} E(\mathbf{v}, \mathbf{h}; \Theta) &= -\mathbf{h}^\top \mathbf{W} \mathbf{v} - \mathbf{a}^\top \mathbf{v} - \mathbf{b}^\top \mathbf{h} \\ &= -\sum_{i=1}^D \sum_{j=1}^F v_i W_{ij} h_j - \sum_{i=1}^D a_i v_i - \sum_{j=1}^F b_j h_j. \end{aligned} \quad (1.20)$$

The conditional distribution of the hidden units given the visible units and also the conditional distribution of the visible units given the hidden units are respectively

computed as

$$P(h_j = 1|\mathbf{v}; \Theta) = \sigma \left( b_j + \sum_{i=1}^D W_{ij} v_i \right), \quad (1.21)$$

$$P(v_i = 1|\mathbf{h}; \Theta) = \sigma \left( a_i + \sum_{j=1}^F W_{ij} h_j \right) \quad (1.22)$$

where  $\sigma(\cdot)$  is a logistic sigmoid function. Due to the unobservable hidden units, the objective function is defined as the marginal distribution of the visible units as

$$P(\mathbf{v}; \Theta) = \frac{1}{Z(\Theta)} \sum_{\mathbf{h}} \exp(-E(\mathbf{v}, \mathbf{h}; \Theta)). \quad (1.23)$$

In medical image analysis, the observations can be real-valued voxel intensities  $\mathbf{v} \in \mathbb{R}^D$ . For this case, it is common to use a Gaussian RBM [9], in which the energy function is given by

$$E(\mathbf{v}, \mathbf{h}; \Theta) = \sum_{i=1}^D \frac{(v_i - a_i)^2}{2s_i^2} - \sum_{i=1}^D \sum_{j=1}^F \frac{v_i}{s_i} W_{ij} h_j - \sum_{j=1}^F b_j h_j \quad (1.24)$$

where  $s_i$  denotes a standard deviation of the  $i$ th visible unit and  $\Theta = \{\mathbf{W}, \mathbf{a}, \mathbf{b}, \mathbf{s} \in \mathbb{R}^D\}$ . This variation leads to the following conditional distribution of visible units given the binary hidden units

$$p(v_i | \mathbf{h}; \Theta) = \frac{1}{\sqrt{2\pi}s_i} \exp \left( -\frac{(v_i - a_i - \sum_{j=1}^F h_j W_{ij})^2}{2s_i^2} \right). \quad (1.25)$$

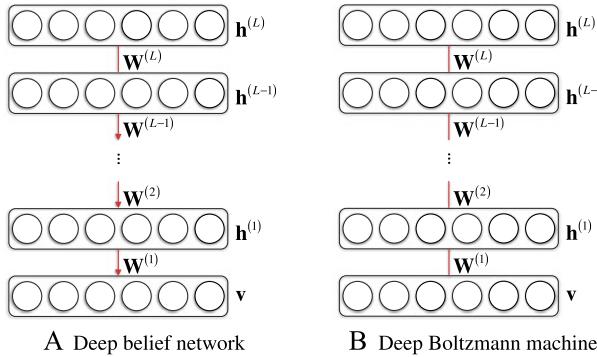
The RBM parameters are usually trained using a contrastive divergence algorithm [15] that maximizes the log-likelihood of observations. When taking the derivative of the log-likelihood with respect to parameters  $\Theta$ , we obtain

$$\frac{\partial}{\partial \Theta} \ln P(\mathbf{v}; \Theta) = \mathbb{E}_{P(\tilde{\mathbf{v}}, \tilde{\mathbf{h}})} \left[ \frac{\partial}{\partial \Theta} E(\tilde{\mathbf{h}}, \tilde{\mathbf{v}}) \right] - \mathbb{E}_{P(\mathbf{h}|\mathbf{v})} \left[ \frac{\partial}{\partial \Theta} E(\mathbf{h}, \mathbf{v}) \right]. \quad (1.26)$$

In the equation, the two expected values of  $\mathbb{E}_{P(\tilde{\mathbf{v}}, \tilde{\mathbf{h}})} [\mathbf{vh}^\top]$  and  $\mathbb{E}_{P(\mathbf{h}|\mathbf{v})} [\mathbf{vh}^\top]$  are approximated by means of a Gibbs sampling technique [16] and their difference provides the direction of updating parameters in (stochastic) gradient descent as follows:

$$\Delta \mathbf{W} = \alpha \left( \mathbb{E}_{P(\tilde{\mathbf{v}}, \tilde{\mathbf{h}})} [\mathbf{vh}^\top] - \mathbb{E}_{P(\mathbf{h}|\mathbf{v})} [\mathbf{vh}^\top] \right), \quad (1.27)$$

$$\Delta \mathbf{a} = \alpha \left( \mathbb{E}_{P(\tilde{\mathbf{v}}, \tilde{\mathbf{h}})} [\mathbf{h}] - \mathbb{E}_{P(\mathbf{h}|\mathbf{v})} [\mathbf{h}] \right), \quad (1.28)$$

**FIGURE 1.9**

Graphical illustrations of two different deep generative models with  $L$  hidden layers.

$$\Delta \mathbf{b} = \alpha \left( E_{P(\tilde{\mathbf{v}}, \tilde{\mathbf{h}})} [\mathbf{v}] - E_{P(\mathbf{h}|\mathbf{v})} [\mathbf{v}] \right) \quad (1.29)$$

where  $\alpha$  denotes the learning rate.

#### 1.4.3.2 Deep Belief Network

Since an RBM is a kind of an auto-encoder, it is straightforward to stack multiple RBMs for deep architecture construction, similar to SAE, which results in a single probabilistic model called a Deep Belief Network (DBN). That is, a DBN has one visible layer  $\mathbf{v}$  and a series of hidden layers  $\mathbf{h}^{(1)}, \dots, \mathbf{h}^{(L)}$  as shown in Fig. 1.9A. Between any two consecutive layers, let  $\{\Theta^{(l)}\}_{l=1}^L$  denote the corresponding RBM parameters. Note that while the top two layers still form an undirected generative model, i.e., RBM, the lower layers form directed generative models. Hence, the joint distribution of the observed units  $\mathbf{v}$  and the  $L$  hidden layers  $\mathbf{h}^{(l)}$  ( $l = 1, \dots, L$ ) in DBN is given as

$$P(\mathbf{v}, \mathbf{h}^{(1)}, \dots, \mathbf{h}^{(L)}) = \left( \prod_{l=0}^{L-2} P(\mathbf{h}^{(l)} | \mathbf{h}^{(l+1)}) \right) P(\mathbf{h}^{(L-1)}, \mathbf{h}^{(L)}) \quad (1.30)$$

where  $\mathbf{h}^{(0)} = \mathbf{v}$ ,  $P(\mathbf{h}^{(l)} | \mathbf{h}^{(l+1)})$  corresponds to a conditional distribution for the units of the layer  $l$  given the units of the layer  $l+1$ , and  $P(\mathbf{h}^{(L-1)}, \mathbf{h}^{(L)})$  denotes the joint distribution of the units in the layers  $L-1$  and  $L$ .

As for the parameter learning, the pretraining scheme described in Section 1.4.2.2 can also be applied as follows:

1. Train the first layer as an RBM with  $\mathbf{v} = \mathbf{h}^{(0)}$ .
2. Use the first layer to obtain a representation of the input that will be used as observation for the second layer, i.e., either the mean activations of  $P(\mathbf{h}^{(1)} = 1 | \mathbf{h}^{(0)})$  or samples drawn from  $P(\mathbf{h}^{(1)} | \mathbf{h}^{(0)})$ .

3. Train the second layer as an RBM, taking the transformed data (samples or mean activations) as training examples (for the visible layer of the RBM).
4. Iterate 2. and 3. for the desired number of layers, each time propagating upward either samples or mean activations.

This greedy layer-wise training of the DBN can be justified as increasing a variational lower bound on the log-likelihood of the data [9]. After the greedy layer-wise procedure is completed, it is possible to perform generative fine-tuning using the wake-sleep algorithm [17]. But in practice, no further procedure is made to train the whole DBN jointly. In order to use a DBN in classification, a trained DBN can also be directly used to initialize a deep neural network with the trained weights and biases. Then the deep neural network can be fine-tuned by means of backpropagation and (stochastic) gradient descent.

#### 1.4.3.3 Deep Boltzmann Machine

A DBM is also structured by stacking multiple RBMs in a hierarchical manner. However, unlike DBN, all the layers in DBM still form an undirected generative model after stacking RBMs as illustrated in Fig. 1.9B. Thus, for the hidden layer  $l$ , its probability distribution is conditioned by its two neighboring layers  $l + 1$  and  $l - 1$ .

Let us consider a three-layer DBM, i.e.,  $L = 2$  in Fig. 1.9B. The energy of the state  $(\mathbf{v}, \mathbf{h}^{(1)}, \mathbf{h}^{(2)})$  in the DBM is given by

$$E(\mathbf{v}, \mathbf{h}^{(1)}, \mathbf{h}^{(2)}; \Theta) = -\mathbf{v}^\top \mathbf{W}^{(1)} \mathbf{h}^{(1)} - (\mathbf{h}^{(1)})^\top \mathbf{W}^{(2)} \mathbf{h}^{(2)} \quad (1.31)$$

where  $\mathbf{W}^{(1)}$  and  $\mathbf{W}^{(2)}$  are symmetric connections of  $(\mathbf{v}, \mathbf{h}^{(1)})$  and  $(\mathbf{h}^{(1)}, \mathbf{h}^{(2)})$ , respectively, and  $\Theta = \{\mathbf{W}^{(1)}, \mathbf{W}^{(2)}\}$ . Given the values of the units in the neighboring layer(s), the probability of the binary visible or binary hidden units being set to 1 is computed as

$$P(h_j^{(1)} = 1 | \mathbf{v}, \mathbf{h}^{(2)}) = \sigma \left( \sum_i W_{ij}^{(1)} v_i + \sum_k W_{jk}^{(2)} h_k^{(2)} \right), \quad (1.32)$$

$$P(h_k^{(2)} = 1 | \mathbf{h}^{(1)}) = \sigma \left( \sum_j W_{jk}^{(2)} h_j^{(1)} \right), \quad (1.33)$$

$$P(v_i = 1 | \mathbf{h}^{(1)}) = \sigma \left( \sum_j W_{ij}^{(1)} h_j^{(1)} \right). \quad (1.34)$$

Note that in the computation of the conditional probability of the hidden units  $\mathbf{h}^{(1)}$ , we incorporate both the lower visible layer  $\mathbf{v}$  and the upper hidden layer  $\mathbf{h}^{(2)}$ , and this makes DBM differentiated from DBN and also more robust to noisy observations [18,19].

For a classification task, it is possible to use DBM by replacing an RBM at the top hidden layer with a discriminative RBM [20], which can also be applied for DBN.

That is, the top hidden layer is now connected to both the lower hidden layer and an additional label layer, which indicates the label of the input  $\mathbf{v}$ . In this way, a DBM can be trained to discover hierarchical and discriminative feature representations by integrating the process of discovering features of inputs with their use in classification [20]. With the inclusion of the additional label layer, the energy of the state  $(\mathbf{v}, \mathbf{h}^{(1)}, \mathbf{h}^{(2)}, \mathbf{o})$  in the discriminative DBM is given by

$$E(\mathbf{v}, \mathbf{h}^{(1)}, \mathbf{h}^{(2)}, \mathbf{o}; \Theta) = -\mathbf{v}^\top \mathbf{W}^{(1)} \mathbf{h}^{(1)} - (\mathbf{h}^{(1)})^\top \mathbf{W}^{(2)} \mathbf{h}^{(2)} - (\mathbf{h}^{(2)})^\top \mathbf{U} \mathbf{o} \quad (1.35)$$

where  $\mathbf{U}$  and  $\mathbf{o} \in \{0, 1\}^C$  denote a connectivity between the top hidden layer and the label layer and a class-label indicator vector, respectively,  $C$  is the number of classes, and  $\Theta = \{\mathbf{W}^{(1)}, \mathbf{W}^{(2)}, \mathbf{U}\}$ . The probability of an observation  $(\mathbf{v}, \mathbf{o})$  is computed by

$$P(\mathbf{v}, \mathbf{o}; \Theta) = \frac{1}{Z(\Theta)} \sum_{\mathbf{h}^{(1)}, \mathbf{h}^{(2)}} \exp(-E(\mathbf{v}, \mathbf{h}^{(1)}, \mathbf{h}^{(2)}, \mathbf{o}; \Theta)). \quad (1.36)$$

The conditional probability of the top hidden units being set to 1 is given by

$$P(h_k^{(2)} = 1 | \mathbf{h}^{(1)}, \mathbf{o}) = \sigma \left( \sum_j W_{jk}^{(2)} h_j^{(1)} + \sum_l U_{lk} o_l \right). \quad (1.37)$$

For the label layer, it uses a softmax function

$$P(o_l = 1 | \mathbf{h}^{(2)}) = \frac{\exp \left[ \sum_k U_{lk} h_k^{(2)} \right]}{\sum_{l'} \exp \left[ \sum_k U_{l'k} h_k^{(2)} \right]}. \quad (1.38)$$

In this way, the hidden units capture class-predictive information about the input vector.

In order to learn the parameters  $\Theta = \{\mathbf{W}^{(1)}, \mathbf{W}^{(2)}, \mathbf{U}\}$ , we maximize the log-likelihood of the observed data  $(\mathbf{v}, \mathbf{o})$ . The derivative of the log-likelihood of the observed data with respect to the model parameters takes the following simple form:

$$\frac{\partial}{\partial \mathbf{W}^{(l)}} \ln P(\mathbf{v}, \mathbf{o}; \Theta) = \mathbb{E}_{\text{data}} \left[ \mathbf{h}^{(l-1)} (\mathbf{h}^{(l)})^\top \right] - \mathbb{E}_{\text{model}} \left[ \mathbf{h}^{(l-1)} (\mathbf{h}^{(l)})^\top \right], \quad (1.39)$$

$$\frac{\partial}{\partial \mathbf{U}} \ln P(\mathbf{v}, \mathbf{o}; \Theta) = \mathbb{E}_{\text{data}} \left[ \mathbf{h}^{(2)} \mathbf{o}^\top \right] - \mathbb{E}_{\text{model}} \left[ \mathbf{h}^{(2)} \mathbf{o}^\top \right] \quad (1.40)$$

where  $\mathbb{E}_{\text{data}} [\cdot]$  denotes the data-dependent statistics obtained by sampling the model conditioned on the visible units  $\mathbf{v}$  ( $\equiv \mathbf{h}^{(0)}$ ) and the label units  $\mathbf{o}$  clamped to the observation and the corresponding label, respectively, and  $\mathbb{E}_{\text{model}} [\cdot]$  denotes the data-independent statistics obtained by sampling from the model. When the model approximates the data distribution well, it can be reached for the equilibrium of data-dependent and data-independent statistics.

In parameter learning, a gradient-based optimization strategy can be used. In Eq. (1.39) and Eq. (1.40), it is necessary to compute the data-dependent and the data-independent statistics. First, because of the two-way dependency in DBM, it is not tractable for the data-dependent statistics. Fortunately, variational mean-field approximation works well for estimating the data-dependent statistics. For the details of computing the data-dependent statistics, please refer to [21]. Similar to DBN, it can be applied for a greedy layer-wise pretraining strategy to provide a good initial configuration of the parameters, which helps the learning procedure converge much faster than random initialization. However, since the DBM integrates both bottom-up and top-down information, the first and last RBMs in the network need modification by using weights twice as big as in one direction. Then, it is performed for iterative alternation of variational mean-field approximation to estimate the posterior probabilities of hidden units and stochastic approximation to update model parameters.

## 1.5 TRICKS FOR BETTER LEARNING

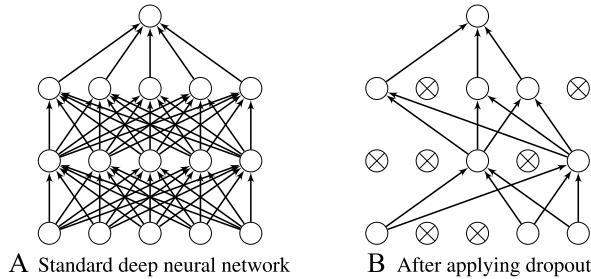
Earlier, LeCun et al. presented that by transforming data to have an identity covariance and a zero mean, i.e., data whitening, the network training could converge faster [3,22]. Besides such a simple trick, recent studies have devised other nice tricks to better train deep models.

### 1.5.1 RECTIFIED LINEAR UNIT (RELU)

The gradient of the logistic sigmoid function and the hyperbolic tangent function vanishes as the value of the respective inputs increases or decreases, which is known as one of the sources to cause the vanishing gradient problem. In this regard, Nair and Hinton suggested to use a Rectified Linear function,  $f(a) = \max(0, a)$ , for hidden Units (ReLU) and validated its usefulness to improve training time by resolving the vanishing gradient problem. However, mathematically, the ReLU has two problems: (i) it is non-differentiable at  $a = 0$ , and thus not valid to be used along with a gradient-based method; (ii) it is unbounded on the positive side, and thus can be a potential problem to cause overfitting. Nonetheless, as for the first problem, since it is highly unlikely that the input to any hidden unit will be at exactly  $a = 0$  at any time, in practice, the gradient of the ReLU at  $a = 0$  is set either 0 or 1. Regarding the unboundedness, the application of a regularization technique is helpful to limit the magnitude of weights, thus circumventing the overfitting issue. Glorot et al. showed that deep neural networks could be trained efficiently by using ReLU and  $\ell_2$ -regularization [23].

### 1.5.2 DROPOUT

A dropout is a simple but helpful technique to train a deep network with a relatively small dataset. The idea of dropout is to randomly deactivate a fraction of the units,

**FIGURE 1.10**

Comparison between a standard deep neural network and the same network with dropout application. The circles with a cross symbol inside denote deactivated units.

e.g., 50%, in a network on each training iteration (Fig. 1.10B). This helps prevent complex co-adaptations among units, i.e., undesirable dependence on the presence of particular other units [24]. By preventing complex co-adaptations with dropout, it naturally helps avoid overfitting, and thus makes the trained model better generalized. The other noteworthy effect of dropout is to provide a way of combining exponentially many different network architectures efficiently. The random and temporal removal of units in training results in different network architectures, and thus at each iteration, it can be thought to train different networks but their connection weights are shared. In testing, all units in the network should be on, i.e., no dropout, but the weights are halved to maintain the same output range.

### 1.5.3 BATCH NORMALIZATION

Ioffe and Szegedy [25] observed that the change in the distribution of network activations due to the change in network parameters during training, which they defined as *internal covariate shift*, causes longer training time. To tackle this issue, they introduced a batch normalization technique by performing normalization for each mini-batch and backpropagating the gradients through the normalization parameters (i.e., scale and shift). Specifically, for each unit in a layer  $l$ , their value is normalized as follows:

$$\hat{a}_k^{(l)} = \frac{a_k^{(l)} - \mathbb{E}[a_k^{(l)}]}{\sqrt{\text{Var}[a_k^{(l)}]}} \quad (1.41)$$

where  $k$  denotes an index of units in the layer  $l$ . A pair of learnable parameters  $\gamma_k^{(l)}$  and  $\beta_k^{(l)}$  are then introduced to scale and shift the normalized values to restore the representation power of the network as follows:

$$y_k^{(l)} = \gamma_k^{(l)} \hat{a}_k^{(l)} + \beta_k^{(l)}. \quad (1.42)$$

The scaled and shifted values are then fed into the following layer as input. In their experiments, it was shown that the batch normalization could greatly shorten the training time and reduce the need for dropout.

---

## 1.6 OPEN-SOURCE TOOLS FOR DEEP LEARNING

With the great successes of deep learning methods in different fields, the leading groups in deep learning have publicized their source codes, tools, or even their deep models trained for some applications. Thanks to their great efforts, it is easy for those who are not familiar with deep models to build their own system or methods. Here, we listed the most widely used tools for deep learning along with their features.

- Caffe<sup>3</sup> was originally developed by the Berkeley Vision and Learning Center (BVLC) at the University of California, Berkeley, and has been being updated by the group and community contributors. Of all the open-source tools for deep learning, it has been the most widely used. Its framework is a BSD-licensed C++ library with Python and MATLAB bindings for training and constructing various deep models [26].
- Theano<sup>4</sup> is a Python library, introduced to the machine learning community [27] and originated in 2008 at the Montreal Institute for Learning Algorithms at the University of Montreal. It has nice properties of tight integration with NumPy, transparent use of a GPU, efficient symbolic differentiation, high speed and stability optimizations, dynamic C code generation, and extensive unit-testing and self-verification.
- Torch<sup>5</sup> is a computing framework with wide support for machine learning algorithms. Similarly as other tools, it also allows to use GPUs and to build neural networks and train it with efficient optimization techniques. However, it is dependent on the programming language Lua.
- MatConvNet<sup>6</sup> is a MATLAB library, primarily designed for implementing CNNs, but also possibly to deploy other deep neural networks [28]. It is simple and efficient to use and also provide many pre-trained models proposed in the literature for image classification, segmentation, etc.
- Besides the tools above, Google and Microsoft recently opened their libraries of Tensorflow<sup>7</sup> and CNTK<sup>8</sup>, respectively.

---

## REFERENCES

1. F. Rosenblatt, The perceptron: a probabilistic model for information storage and organization in the brain, *Psychol. Rev.* (1958) 65–386.
2. D.E. Rumelhart, G.E. Hinton, R.J. Williams, Learning representations by back-propagating errors, *Nature* 323 (6088) (1986) 533–536.

3. Y. LeCun, L. Bottou, G.B. Orr, K.R. Müller, Efficient BackProp, in: Neural Networks: Tricks of the Trade, Springer, Berlin, Heidelberg, 1998, pp. 9–50.
4. M. Li, T. Zhang, Y. Chen, A.J. Smola, Efficient mini-batch training for stochastic optimization, in: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, 2014, pp. 661–670.
5. K. Hornik, Approximation capabilities of multilayer feedforward networks, *Neural Netw.* 4 (2) (1991) 251–257.
6. G. Schwarz, Estimating the dimension of a model, *Ann. Stat.* 6 (2) (1978) 461–464.
7. Y. Bengio, Learning deep architectures for AI, *Found. Trends Mach. Learn.* 2 (1) (2009) 1–127.
8. X. Glorot, Y. Bengio, Understanding the difficulty of training deep feedforward neural networks, in: Y.W. Teh, D.M. Titterington (Eds.), *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, vol. 9, 2010, pp. 249–256.
9. G.E. Hinton, R.R. Salakhutdinov, Reducing the dimensionality of data with neural networks, *Science* 313 (5786) (2006) 504–507.
10. H. Lee, C. Ekanadham, A. Ng, Sparse deep belief net model for visual area v2, in: J. Platt, D. Koller, Y. Singer, S. Roweis (Eds.), *Advances in Neural Information Processing Systems*, vol. 20, MIT Press, Cambridge, MA, 2008, pp. 873–880.
11. H. Larochelle, Y. Bengio, J. Louradour, P. Lamblin, Exploring strategies for training deep neural networks, *J. Mach. Learn. Res.* 10 (2009) 1–40.
12. H.-C. Shin, M.R. Orton, D.J. Collins, S.J. Doran, M.O. Leach, Stacked autoencoders for unsupervised feature learning and multiple organ detection in a pilot study using 4D patient data, *IEEE Trans. Pattern Anal. Mach. Intell.* 35 (8) (2013) 1930–1943.
13. Y. Bengio, P. Lamblin, D. Popovici, H. Larochelle, Greedy layer-wise training of deep networks, in: B. Schölkopf, J. Platt, T. Hoffman (Eds.), *Advances in Neural Information Processing Systems*, vol. 19, MIT Press, Cambridge, MA, 2007, pp. 153–160.
14. C.M. Bishop, *Neural Networks for Pattern Recognition*, Oxford University Press, Inc., New York, NY, USA, 1995.
15. G.E. Hinton, Training products of experts by minimizing contrastive divergence, *Neural Comput.* 14 (8) (2000) 1771–1800.
16. C.M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*, Springer-Verlag, New York, 2006.
17. G. Hinton, P. Dayan, B. Frey, R. Neal, The wake–sleep algorithm for unsupervised neural networks, *Science* 268 (5214) (1995) 1158–1161.
18. R. Salakhutdinov, G.E. Hinton, Deep Boltzmann machines, in: *Proceedings of the International Conference on Artificial Intelligence and Statistics*, 2009, pp. 448–455.
19. N. Srivastava, R. Salakhutdinov, Multimodal learning with deep Boltzmann machines, in: *Advances in Neural Information Processing Systems*, vol. 25, 2012, pp. 2231–2239.
20. H. Larochelle, Y. Bengio, Classification using discriminative restricted Boltzmann machines, in: *Proceedings of the 25th International Conference on Machine Learning*, 2008, pp. 536–543.
21. R. Salakhutdinov, G. Hinton, An efficient learning procedure for deep Boltzmann machines, *Neural Comput.* 24 (8) (2012) 1967–2006.
22. S. Wiesler, H. Ney, A convergence analysis of log-linear training, in: J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, K. Weinberger (Eds.), *Advances in Neural Information Processing Systems*, vol. 24, 2011, pp. 657–665.

23. X. Glorot, A. Bordes, Y. Bengio, Deep sparse rectifier neural networks, in: Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, 2011, pp. 315–323.
24. N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: a simple way to prevent neural networks from overfitting, *J. Mach. Learn. Res.* 15 (2014) 1929–1958.
25. S. Ioffe, C. Szegedy, Batch normalization: accelerating deep network training by reducing internal covariate shift, in: Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6–11 July 2015, 2015, pp. 448–456.
26. Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, T. Darrell, Caffe: convolutional architecture for fast feature embedding, in: Proceedings of the 22nd ACM International Conference on Multimedia, 2014, pp. 675–678.
27. J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, Y. Bengio, Theano: a CPU and GPU math expression compiler, in: Proceedings of the Python for Scientific Computing Conference, 2010.
28. A. Vedaldi, K. Lenc, MatConvNet – convolutional neural networks for Matlab.

---

## NOTES

1. Each unit in the visible layer takes a scalar value as input. Thus, the observation should be represented in a vector form with elements of raw voxel intensities or features, for instance.
2. It is mostly designed empirically.
3. <https://github.com/BVLC/caffe>.
4. <http://deeplearning.net/software/theano>.
5. <http://torch.ch>.
6. <http://www.vlfeat.org/matconvnet>.
7. <https://www.tensorflow.org>.
8. <https://github.com/Microsoft/CNTK>.

# An Introduction to Deep Convolutional Neural Nets for Computer Vision

# 2

Suraj Srinivas, Ravi K. Sarvadevabhatla, Konda R. Mopuri, Nikita Prabhu,  
Srinivas S.S. Kruthiventi, R. Venkatesh Babu

*Indian Institute of Science, Bangalore, India*

## CHAPTER OUTLINE

2.1	<b>Introduction</b>	26
2.2	<b>Convolutional Neural Networks</b>	27
2.2.1	<i>Building Blocks of CNNs</i>	27
2.2.1.1	<i>Why Convolutions?</i>	27
2.2.1.2	<i>Max-Pooling</i>	29
2.2.1.3	<i>Nonlinearity</i>	29
2.2.1.4	<i>Fully-Connected Layers</i>	29
2.2.2	<i>Depth</i>	29
2.2.3	<i>Learning Algorithm</i>	30
2.2.3.1	<i>Gradient-Based Optimization</i>	30
2.2.3.2	<i>Dropout</i>	31
2.2.3.3	<i>Batch Normalization</i>	31
2.2.4	<i>Tricks to Increase Performance</i>	31
2.2.5	<i>Putting It All Together: AlexNet</i>	32
2.2.6	<i>Using Pre-Trained CNNs</i>	32
2.2.6.1	<i>Fine-Tuning</i>	32
2.2.6.2	<i>CNN Activations as Features</i>	33
2.2.7	<i>Improving AlexNet</i>	33
2.3	<b>CNN Flavors</b>	34
2.3.1	<i>Region-Based CNNs</i>	34
2.3.2	<i>Fully Convolutional Networks</i>	35
2.3.3	<i>Multi-Modal Networks</i>	38
2.3.4	<i>CNNs with RNNs</i>	40
2.3.4.1	<i>Action Recognition</i>	41
2.3.4.2	<i>Image and Video Captioning</i>	42
2.3.4.3	<i>Visual Question Answering</i>	42
2.3.5	<i>Hybrid Learning Methods</i>	43

2.3.5.1	<i>Multi-Task Learning</i> .....	43
2.3.5.2	<i>Similarity Learning</i> .....	44
2.4	<b>Software for Deep Learning</b> .....	45
	<b>References</b> .....	46

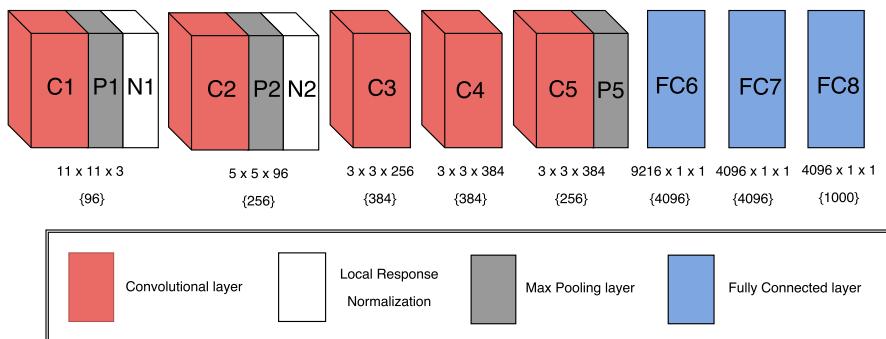
---

## 2.1 INTRODUCTION

Computer vision problems like image classification and object detection have traditionally been approached using hand-engineered features like SIFT [63] and HoG [19]. Representations based on the bag-of-visual-words descriptor [110], in particular, enjoyed success in image classification. These were usually followed by learning algorithms like Support Vector Machines (SVMs). As a result, the performance of these algorithms crucially relied on the features used. This meant that progress in computer vision was based on hand-engineering better sets of features. With time, these features started becoming more and more complex, resulting in a difficulty of coming up with better, more complex features. From the perspective of the computer vision practitioner, there were two steps to be followed: feature design and learning algorithm design, both of which were largely independent.

Meanwhile, some researchers in the machine learning community had been working on learning models which incorporated learning of features from raw images. These models typically consisted of multiple layers of nonlinearity. This property was considered to be very important, and this lead to the development of the first deep learning models. Early examples like Restricted Boltzmann Machines [40], Deep Belief Networks [41], and Stacked Autoencoders [97] showed promise on small datasets. The primary idea behind these works was to leverage the vast amount of unlabeled data to train models. This was called the ‘unsupervised pre-training’ stage. It was believed that these ‘pre-trained’ models would serve as a good initialization for further supervised tasks such as image classification. Efforts to scale these algorithms on larger datasets culminated in 2012 during the ILSVRC competition [79], which involved, among other things, the task of classifying an image into one of thousand categories. For the first time, a Convolutional Neural Network (CNN) based deep learned model [56] brought down the error rate on that task by half, beating traditional hand-engineered approaches. Surprisingly, this could be achieved by performing end-to-end supervised training, without the need for unsupervised pre-training. Over the next couple of years, ‘ImageNet classification using deep neural networks’ [56] became one of the most influential papers in computer vision. Convolutional Neural Networks, a particular form of deep learning models, have since been widely adopted by the vision community. In particular, the network trained by Alex Krizhevsky, popularly called “AlexNet” has been used and modified for various vision problems. Hence, in this chapter, we primarily discuss CNNs, as they are more relevant to the vision community. This can also serve as a guide for beginning practitioners in deep learning/computer vision.

The chapter is organized as follows. We first develop the general principles behind CNNs (Section 2.2), and then discuss various modifications to suit different problems

**FIGURE 2.1**

An illustration of the weights in the AlexNet model. Note that after every layer, there is an implicit ReLU nonlinearity. The number inside curly braces represents the number of filters with dimensions mentioned above it.

(Section 2.3). Finally, we discuss briefly about some of the existing software tools available for implementing these algorithms.

## 2.2 CONVOLUTIONAL NEURAL NETWORKS

The idea of a Convolutional Neural Network (CNN) is not new. This model had been shown to work well for hand-written digit recognition [59] as early as 1998. However, due to the inability of these networks to scale to much larger images, they slowly fell out of favor. This was largely due to memory and hardware constraints, and the unavailability of large amounts of training data. With increase in computational power thanks to wide availability of GPUs, and the introduction of large scale datasets like the ImageNet [79], it was possible to train larger, more complex models. This was first shown by the popular *AlexNet* model which was discussed earlier. This largely kick-started the usage of deep networks in computer vision.

### 2.2.1 BUILDING BLOCKS OF CNNS

In this section, we shall look at the basic building blocks of CNNs in general. This assumes that the reader is familiar with traditional neural networks, which we shall call “fully connected layers”. Fig. 2.1 shows a representation of the weights in the *AlexNet* model. While the first five layers are convolutional, the last three are fully connected layers.

#### 2.2.1.1 Why Convolutions?

Using traditional neural networks for real-world image classification is impractical for the following reason: Consider a 2D image of size  $200 \times 200$  for which we would

have 40,000 input nodes. If the hidden layer has 20,000 nodes, the size of the matrix of input weights would be  $40,000 \times 20,000 = 800$  million. This is just for the first layer – as we increase the number of layers, this number increases even more rapidly. Besides, vectorizing an image completely ignores the complex 2D spatial structure of the image. How do we build a system that overcomes both these disadvantages?

One way is to use 2D convolutions instead of matrix multiplications. Learning a set of convolutional filters (each of  $11 \times 11$ , say) is much more tractable than learning a large matrix ( $40,000 \times 20,000$ ). 2D convolutions also naturally take the 2D structure of images into account. Alternately, convolutions can also be thought of as regular neural networks with two constraints [11]:

- *Local connectivity.* This comes from the fact that we use a convolutional filter with dimensions much smaller than the image it operates on. This contrasts with the *global* connectivity paradigm typically relevant to vectorized images.
- *Weight sharing.* This comes from the fact that we perform convolutions, i.e., we apply the same filter across the image. This means that we use the same *local* filters on many locations in the image. In other words, the weights between all these filters are shared.

There is also evidence from visual neuroscience for similar computations within the human brain. Hubel and Wiesel [47] found two types of cells in the primary visual cortex, the simple cells and the complex cells. The simple cell responded primarily to oriented edges and gratings, which are reminiscent of Gabor filters, a special class of convolutional filters. The complex cells were also sensitive to these edges and grating. However, they exhibited spatial invariance as well. This motivated the Neocognitron model [25], which proposed the learning of convolutional filters in an artificial neural network. This model is said to have inspired convolutional networks, which are analogous to the simple cells mentioned above.

In practical CNNs, however, the convolution operations are not applied in the traditional sense wherein the filter shifts one position to the right after each multiplication. Instead, it is common to use larger shifts (commonly referred to as stride). This is equivalent to performing image down-sampling after regular convolution.

If we wish to train these networks on RGB images, one would need to learn multiple *multi-channel* filters. In the representation in Fig. 2.1, the numbers  $11 \times 11 \times 3$ , along with {96} below C1, indicate that there are 96 filters in the first layers, each of spatial dimension of  $11 \times 11$ , with one for each of the 3 RGB channels. As a result, the resulting feature activation after convolution has 96 channels, and we can now apply another set of multi-channel features for these as well.

We note that this paradigm of convolution-like operations (location independent feature-detectors) is not entirely suitable for registered images. As an example, images of faces require different feature-detectors at different spatial locations. To account for this, some models [90] consider only locally-connected networks with no weight-sharing. Thus, the choice of layer connectivity depends on the underlying type of problem.

### 2.2.1.2 Max-Pooling

The Neocognitron model inspired the modeling of simple cells as convolutions. Continuing in the same vein, the complex cells can be modeled as a max-pooling operation. This operation can be thought of as a *max filter*, where each  $n \times n$  region is replaced with its max value. This operation serves two purposes:

1. It picks out the highest activation in a local region, thereby providing a small degree of spatial invariance. This is analogous to the operation of complex cells.
2. It reduces the size of the activation for the next layer by a factor of  $n^2$ . With a smaller activation size, a smaller number of parameters need to be learned in the later layers.

Other types of pooling include average-pooling, winner-takes-all pooling [87] and stochastic pooling [117]. However, these are not as commonly used max-pooling.

### 2.2.1.3 Nonlinearity

Deep networks usually consist of convolutions followed by a nonlinear operation after each layer. This is necessary because cascading linear systems (like convolutions) is another linear system. Nonlinearities between layers ensure that the model is more expressive than a linear model.

In theory, no nonlinearity has more expressive power than any other, as long as they are continuous, bounded and monotonically increasing [44]. Traditional feed-forward neural networks used the sigmoid ( $\sigma(x) = \frac{1}{1+e^{-x}}$ ) or the tanh ( $\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$ ) nonlinearities. However, modern convolutional networks use the Rectified Linear Unit (ReLU), which is defined as  $\text{ReLU}(x) = \max(0, x)$ . CNNs with this nonlinearity have been found to train faster [70].

Recently, Maas et al. [64] introduced a new kind of nonlinearity, called the leaky-ReLU. It was defined as  $\text{Leaky-ReLU}(x) = \max(0, x) + \alpha \min(0, x)$ , where  $\alpha$  is a pre-determined parameter. He et al. [39] improved on this by suggesting that the  $\alpha$  parameter also be learned, leading to a much richer model.

### 2.2.1.4 Fully-Connected Layers

Traditionally, neural networks were composed of matrix multiplications alternated with nonlinearities like sigmoid. These matrix multiplication layers are now called as *fully-connected layer*, owing to the fact that each unit in the previous layer is connected to every unit in the next layer. This is in contrast with convolutional layers, where there is only a local spatial connection. In modern networks, using many fully connected layers are generally avoided as it uses an extremely large number of parameters [38].

## 2.2.2 DEPTH

The Universal Approximation theorem [44] states that a neural network with a single hidden layer is sufficient to model any continuous function. However, it has been

shown [10] that such networks need an exponentially large number of neurons when compared to a neural network with many hidden layers.

Although the motivation for creating deeper networks was clear, for a long time researchers did not have an algorithm that could efficiently train neural networks with more than 3 layers. With the introduction of greedy layer-wise pre-training [41], researchers were able to train much deeper networks. This played a major role in bringing the so-called *Deep Learning* systems into mainstream machine learning. Modern deep networks such as AlexNet have 7 layers. Recent models like VGGnet [83] and GoogleNet [89] have 19 and 22 layers, respectively, and were shown to perform much better than AlexNet. One disadvantage of these very deep networks is that they become very difficult to train. Srivastava et al. [86] introduced Highway networks, using which it was possible to train very deep neural networks, up to a 100 layers deep. He et al. [38] used a modification of this technique with 152 layers to obtain state-of-the-art results on the ILSVRC 2015 dataset.

### 2.2.3 LEARNING ALGORITHM

A powerful, expressive model is of little use without an algorithm to learn the model’s parameters efficiently. Greedy layer-wise pre-training approaches in the pre-AlexNet era attempted to create such an efficient algorithm. For computer vision tasks, however, it turns out that a simpler supervised training procedure is enough to learn a powerful model.

Learning is generally performed by minimizing a loss function which is dependent on the underlying task. Tasks based on classification use the softmax loss function or the sigmoid cross-entropy function, while those involving regression use the Euclidean error function. In the example of Fig. 2.1, the output of the FC8 layer is trained to represent one of thousand classes of the dataset.

#### 2.2.3.1 Gradient-Based Optimization

Neural networks are generally trained using the backpropagation algorithm [78], which uses the chain rule to speed up computation of the gradient for the gradient descent (GD) algorithm. However, for datasets with thousands (or more) of data points, using GD is impractical. In such cases, an approximation called the Stochastic Gradient Descent (SGD) is often used, where one computes gradients with respect to individual data points rather than the entire dataset. It has been found that training with SGD generalizes better than with GD. However, one disadvantage of SGD is that it is relatively slow to converge [12]. To counteract this, SGD is typically used with a mini-batch, where the mini-batch typically contains a small number of data-points ( $\sim 100$ ) rather than a single data-point.

Momentum [74] belongs to a family of methods that aim to speed the convergence of SGD. This is largely used in practice to train deep networks, and is often considered as an essential component. Extensions like Adagrad [23], Nesterov’s accelerated GD [72], Adadelta [116] and Adam [55] are known to work equally well,

if not better than vanilla momentum in certain cases. For detailed discussion on how these methods work, the reader is encouraged to read the following survey paper [88].

### 2.2.3.2 Dropout

When training a network with a large number of parameters, an effective regularization mechanism is essential to combat overfitting. Classical regularizers such as  $\ell_1$  or  $\ell_2$  regularization on the weights of the neural net have been found to be insufficient in this aspect. Dropout is a powerful regularization method [42] which has been shown to improve generalization for large neural nets. In dropout, we *randomly* drop neurons with a probability  $p$  during training. As a result, only a random subset of neurons are trained in a single iteration of SGD. At test time, we use all neurons, however, we simply multiply the activation of each neuron with  $p$  to account for the scaling. Hinton et al. [42] showed that this procedure was equivalent to training a large ensemble of neural nets with shared parameters, and then using their geometric mean to obtain a single prediction.

Many extensions to dropout like DropConnect [99] and Fast Dropout [103] have been shown to work better in certain cases. Maxout [32] is a nonlinearity that improves performance of a network which uses dropout.

### 2.2.3.3 Batch Normalization

Batch Normalization (BN) [48] is another useful regularizer that improves generalization as well as drastically speeds up convergence. This technique tackles the problem of *internal covariate shift*, where the distribution of each layer's inputs change continuously during the training process. This is a result of each layer's predecessor's changing weights, which result in a changing distribution of output activations. This phenomenon usually slows down training and requires careful initialization.

To counteract this problem, BN normalizes the output activations of a layer to ensure that it's range is within a small interval. Specifically, BN performs normalization with the running average of the mean-variance statistics of each mini-batch. Additional shift-and-scale parameters are also learned to counteract the normalization effect if needed. In recent times, BN has been found to be an essential component in training very deep networks.

## 2.2.4 TRICKS TO INCREASE PERFORMANCE

While the techniques and components described above are well-grounded, some additional *tricks* are crucial to obtaining *state-of-the-art* performance.

It is well known that machine learning models perform better in the presence of more data. Data augmentation is a process by which some geometric transforms are applied to training data to increase their number. Some examples of commonly used geometric transforms include random cropping, RGB jittering, image flipping and small rotations. It has been found that using augmented data typically boosts performance by about 3% [14].

Also well-known is the fact that an ensemble of models perform better than one. Hence, it is the commonplace to train several CNNs and average their predictions at test time. Using ensembles has been found to typically boost accuracy by 1–2% [83,89].

### 2.2.5 PUTTING IT ALL TOGETHER: ALEXNET

The building blocks discussed above largely describe AlexNet as a whole. As shown in Fig. 2.1, only layers 1, 2, and 5 contain max-pooling, while dropout is only applied to the last two fully connected layers as they contain the most number of parameters. Layers 1 and 2 also contain Local Response Normalization, which has not been discussed as it has been shown [14] that its absence does not impact performance.

This network was trained on the ILSVRC 2012 training data, which contained 1.2 million training images belonging to 1000 classes. This was trained on 2 GPUs over the course of one month. The same network can be trained today in little under a week using more powerful GPUs [14]. The hyper-parameters of the learning algorithms like learning rate, momentum, dropout and weight decay were hand tuned. It is also interesting to note the trends in the nature of features learned at different layers. The earlier layers tend to learn Gabor-like oriented edges and blob-like features, followed by layers that seem to learn more higher order features like shapes. The very last layers seem to learn semantic attributes such as eyes or wheels, which are crucial parts in several categories. A method to visualize these was provided by Zeiler and Fergus [115].

### 2.2.6 USING PRE-TRAINED CNNS

One of the main reasons for the success of the AlexNet model was that it was possible to directly use the pre-trained model to do various other tasks which it was not originally intended for. It became remarkably easy to download a learned model, and then tweak it slightly to suit the application at hand. We describe two such ways to use models in this manner.

#### 2.2.6.1 *Fine-Tuning*

Given a model trained for image classification, how does one modify it to perform a different (but related) task? The answer is to just use the trained weights as an initialization and run SGD again for this new task. Typically, one uses a learning rate much lower than what was used for learning the original net. If the new task is very similar to the task of image classification (with similar categories), then one need not re-learn a lot of layers. The earlier layers can be fixed and only the later, more semantic layers need to be re-learned. However, if the new task is very different, one ought to either re-learn all layers, or learn everything from scratch. The number of layers to re-learn also depends on the number of data points available for training the new task. The more the data, the higher is the number of layers that can be re-learned. The reader is urged to refer to Yosinski et al. [112] for more thorough guidelines.

### 2.2.6.2 CNN Activations as Features

As remarked earlier, the later layers in AlexNet seem to learn visually semantic attributes. These intermediate representations are crucial in performing 1000-way classification. Since these represent a wide variety of classes, one can use the FC7 activation of an image as a generic feature descriptor. These features have been found to be better than hand-crafted features like SIFT or HoG for various computer vision tasks.

Donahue et al. [22] first introduced the idea of using CNN activations as features and performed tests to determine their suitability for various tasks. Activations of fully-connected layers are also used for image retrieval [7]. In fact, these serve as generic features and can be used in several tasks [75] as a strong baseline. Hypercolumns [36] are activations *across* layers as a feature. Specifically, they look at the activations produced by a single image pixels across the network and pool them together. They were found to be useful for fine-grained tasks such as keypoint localization.

### 2.2.7 IMPROVING ALEXNET

The performance of AlexNet motivated a number of CNN-based approaches, all aimed at a performance improvement over and above that of AlexNet's. Just as AlexNet was the winner for ILSVRC challenge in 2012, a CNN-based net Overfeat [80] was the top-performer at ILSVRC-2013. Their key insight was that training a convolutional network to simultaneously classify, locate, and detect objects in images can boost the classification accuracy and the detection and localization accuracy of all tasks. Given its multi-task learning paradigm, we discuss Overfeat when we discuss hybrid CNNs and multi-task learning in Section 2.3.5.

GoogleNet [89], the top-performer at ILSVRC-2014, established that very deep networks can translate to significant gains in classification performance. Since naively increasing the number of layers results in a large number of parameters, the authors employ a number of “design tricks”. One such trick is to have a trivial  $1 \times 1$  convolutional layer after a regular convolutional layer. This has the net effect of not only reducing the number of parameters, but also results in CNNs with more expressive power. In fact, it is argued that having one or more  $1 \times 1$  convolutional layers is akin to having a multi-layer Perceptron network processing the outputs of the convolutional layer that precedes it. Another trick that the authors utilize is to involve inner layers of the network in the computation of the objective function instead of the typical final softmax layer (as in AlexNet). The authors attribute scale invariance as the reason behind this design decision.

VGG-19 [83] is another example of a high-performing CNN where the deeper-is-better philosophy is applied in the net design. An interesting feature of VGG design is that it forgoes larger sized convolutional filters for stacks of smaller sized filters. These smaller sized filters tend to be chosen so that they contain approximately the same number of parameters as the larger filters they supposedly replace. The net ef-

fect of this design decision is efficiency and regularization-like effect on parameters due to the smaller size of the filters involved.

Deep Residual Networks [38] takes depth to the extreme by training layers that are as deep as 152–200 layers. This is achieved by adding so-called *residual* connections to layers of the neural network. The motivation behind this is to train layers to fit the residual map, rather than directly fit the underlying map. In other words, instead of learning a mapping  $h(x)$ , we learn  $f(x) = h(x) - x$ , and then later add  $f(x) + x$ . This simple technique was found to be useful to learn models with hundreds of layers, yielding state-of-the-art accuracy in many tasks.

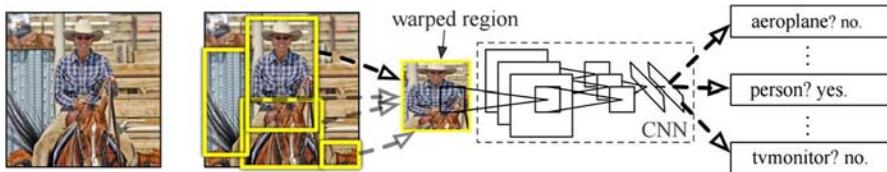
## 2.3 CNN FLAVORS

### 2.3.1 REGION-BASED CNNS

Most CNNs that are trained for image classification are trained using a dataset of images containing a single object. At test time, even in case of multiple objects, the CNN may still predict a single class. This inherent problem with the design of the CNNs is not restricted to image classification alone. For example, the problem of object detection and localization requires not only classifying the image but also estimating the class and precise location of the object(s) present in the image. Object detection is challenging since we potentially want to detect multiple objects with varying sizes within a single image. It generally requires processing the image patch-wise, looking for the presence of objects. Neural nets have been employed in this way for detecting specific objects like faces [93,77] and for pedestrians [81].

Meanwhile, detecting a set of object-like regions in a given image – also called region proposals or object proposals – has gained a lot of attention [45]. These region proposals are class agnostic and reduce the overhead incurred by the traditional exhaustive sliding window approach. These region proposal algorithms operate at low level and output hundreds of object like image patches at multiple scales. In order to employ a classification net toward the task of object localization, image regions of different scales have to be searched one at a time.

Girshick et al. [28] attempt to solve the object localization problem using a set of region proposals. During test time, the method generates around 2000 category independent region proposals using selective search approach [92] from the test image. They employ a simple affine image warping to feed each of these proposals to a CNN trained for classification. The CNN then describes each of these regions with a fixed size high level semantic feature. Finally, a set of category specific linear SVMs classify each region, as shown in Fig. 2.2. This method achieved the best detection results on the PASCAL VOC 2012 dataset. As this method uses image regions followed by a CNN, it is dubbed R-CNN (Region-based CNN). Generating region proposals is a computational bottleneck. Improvements [27,37] are proposed to make this approach computationally efficient and also to increase the detection accuracy. Fully convolutional nets called Region Proposal Nets (RPNs) [76,109] are presented to compute

**FIGURE 2.2**

Object detection system of Girshick et al. [28] using deep features extracted from image regions.

*Source: Girshick et al. [28]*

region proposals at no extra cost while detecting the objects. These networks detect the objects with a very less number of proposals (as low as 20) compared to the earlier region proposal works.

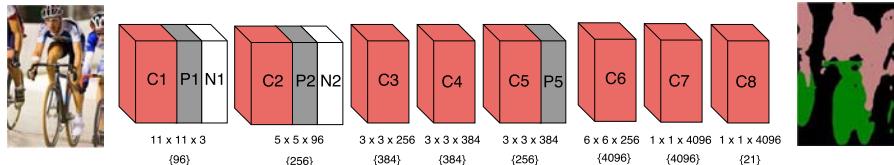
A series of works adapted the R-CNN approach to extract richer set of features at patch or region level to solve a wide range of target applications in vision. However, CNN representations lack robustness to geometric transformations restricting their usage. Gkioxari and Malik [31] show empirical evidence that the global CNN features are sensitive to general transformations such as translation, rotation and scaling. In their experiments, they report that this inability of global CNN features translates directly into a loss in the classification accuracy. They proposed a simple technique to pool the CNN activations extracted from the local image patches. The method extracts image patches in an exhaustive sliding-window manner at different scales and describes each of them using a CNN. The resulting dense CNN features are pooled using VLAD [50] in order to result in a representation which incorporates spatial as well as semantic information.

Instead of considering the image patches at exhaustive scales and image locations, it is possible to utilize the objectness prior to automatically extract the image patches at different scales [69]. This is shown to be a more robust image representation by aggregating the individual CNN features from the patches for image retrieval.

A related technique can be used to perform multi-label classification as well [105]. This can be done by considering an arbitrary number of region proposals in an image and sharing a common CNN across all of them in order to obtain individual predictions. Finally, we employ a simple pooling technique to produce the final multi-label prediction.

### 2.3.2 FULLY CONVOLUTIONAL NETWORKS

The success of Convolutional Neural Networks in the tasks of image classification [56,89] and object detection [28] has inspired researchers to use deep networks for more challenging recognition problems like semantic object segmentation and scene parsing. Unlike image classification, semantic segmentation and scene parsing are problems of structured prediction where every pixel in the image grid needs to be

**FIGURE 2.3**

Fully Convolutional Net – AlexNet modified to be fully convolutional for performing semantic object segmentation on PASCAL VOC 2012 dataset with 21 classes.

assigned a label of the class to which it belongs (e.g., road, sofa, table, etc.). This problem of per-pixel classification has been traditionally approached by generating region-level (e.g., superpixel) hand crafted features and classifying them using a Support Vector Machine (SVM) into one of the possible classes.

Doing away with these engineered features, Farabet et al. [24] used hierarchical learned features from a convolutional neural net for scene parsing. Their approach comprised of densely computing multi-scale CNN features for each pixel and aggregating them over image regions upon which they are classified. However, their method still required the post-processing step of generating over-segmented regions, like superpixels, for obtaining the final segmentation result. Additionally, the CNNs used for multi-scale feature learning were not very deep with only three convolution layers.

Later, Long et al. [61] proposed a fully convolutional network architecture for learning per-pixel tasks, like semantic segmentation, in an end-to-end manner. This is shown in Fig. 2.3. Each layer in the fully convolutional net (FullConvNet) performs a location invariant operation, i.e., a spatial shift of values in the input to the layer will only result in an equivalent scaled spatial shift in its output while keeping the values nearly intact. This property of translational invariance holds true for the convolutional and maxpool layers which form the major building blocks of a FullConvNet. Further, these layers have an output-centered, fixed-size receptive field on its input blob. These properties of the layers of FullConvNet allow it to retain the spatial structure present in the input image in all of its intermediate and final outputs.

Unlike CNNs used for image classification, a FullConvNet does not contain any densely connected/inner product layers as they are not translation invariant. The restriction on the size of input image to a classification CNN (e.g.,  $227 \times 227$  for AlexNet [56],  $224 \times 224$  for VGG [83]) is imposed due to the constraint on the input size to its inner product layers. Since a FullConvNet does not have any of these inner product layers, it can essentially operate on input images of any arbitrary size.

During the design of CNN architectures, one has to make a trade-off between the number of channels and the spatial dimensions for the data as it passes through each layer. Generally, the number of channels in the data are made to increase progressively while bringing down its spatial resolution, by introducing stride in the convolution and max-pool layers of the net. This is found to be an effective strategy

for generating richer semantic representations in a hierarchical manner. While this method enables the net to recognize complex patterns in the data, it also diminishes the spatial resolution of the data blob progressively after each layer. While this is not a major concern for classification nets which require only a single label for the entire image, this results in per-pixel prediction only at a sub-sampled resolution in case of FullConvNets. For tackling this problem, Long et al. [61] have proposed a deconvolution layer which brings back the spatial resolution from the sub-sampled output through a learned upsampling operation. This upsampling operation is performed at intermediate layers of various spatial dimensions and are concatenated to obtain pixel-level features at the original resolution.

On the other hand, a more simplistic approach [15] for maintaining resolution can be done by removing the stride in the layers of FullConvNet, wherever possible. Following this, the FullConvNet predicted output is modeled as a unary term for Conditional Random Field (CRF) constructed over the image grid at its original resolution. With labeling smoothness constraint enforced through pair-wise terms, the per-pixel classification task is modeled as a CRF inference problem. While this post-processing of FullConvNet's coarse labeling using CRF has been shown to be effective for pixel-accurate segmentation, a better approach [119] would be where the CRF constructed on image is modeled as a Recurrent Neural Network (RNN). By modeling the CRF as an RNN, it can be integrated as a part of any Deep Convolutional Net making the system efficient at both semantic feature extraction and fine-grained structure prediction. This enables the end-to-end training of the entire FullConvNet + RNN system using the stochastic gradient descent (SGD) algorithm to obtain fine pixel-level segmentation.

Noh et al. [73] proposed a semantic object segmentation architecture where a full-fledged deconvolution network follows the convolution network in the segmentation pipeline. Here, the architecture of the deconvolution network mirrors that of the convolution network with the convolution and pooling layers replaced by deconvolution and unpooling layers, respectively. In this architecture, the authors observe that the convolution network plays the role of a multi-dimensional semantic feature extractor whereas the deconvolution network plays the role of an object shape generator using the extracted features. This pipeline of convolution–deconvolution network when used together with a FullConvNet architecture as an ensemble is observed to give superior segmentation performance.

Visual saliency prediction is another important per-pixel prediction problem in computer vision. This task involves predicting the per-pixel saliency map for an image as given by human eye fixations. Huang et al. [46] modified deep networks trained for image classification into fully convolutional networks and fine-tuned them for eye fixation prediction with saliency based objectives. Their model handles multi-scale aspects of saliency by considering a two-stream architecture with each stream specializing for a scale. Kruthiventi and Babu [57] proposed another fully convolutional architecture which characterizes the multi-scale aspects using inception blocks and captures the global context using convolutional layers with large receptive fields. Another work [108], proposed a multi-task FullConvNet architecture – DeepSaliency

– for joint saliency detection and semantic object segmentation. Their work showed that learning features collaboratively for two related prediction tasks can boost overall performance.

### 2.3.3 MULTI-MODAL NETWORKS

The success of CNNs on standard RGB vision tasks is naturally extended to works on other perception modalities like RGB-D and motion information in the videos. There has been an increasing evidence for the successful adaptation of the CNNs to learn efficient representations from the depth images. Socher et al. [84] exploited the information from color and depth modalities for addressing the problem of classification. In their approach, a single layer of CNN extracts low level features from both RGB and depth images separately. These low level features from each modality are given to a set of RNNs for embedding into a lower dimension. Concatenation of the resulting features forms the input to the final soft-max layer. The work by Couprise et al. [18] extended the CNN method of [24] to label the indoor scenes by treating depth information as an additional channel to the existing RGB data. Similarly, Wang et al. [100] adapt an unsupervised feature learning approach to scene labeling using RGB-D input with four channels. Gupta et al. [35] proposed an encoding for the depth images that allows CNNs to learn stronger features than from the depth image alone. They encode depth image into three channels at each pixel: horizontal disparity, height above ground, and the angle the pixel’s local surface normal makes with the inferred gravity direction. Their approach for object detection and segmentation processes RGB and the encoded depth channels separately. The learned features are fused by concatenating and further classified using an SVM classifier.

Similarly, one can think of extending these works for video representation and understanding. When compared to still images, videos provide important additional information in the form of motion. However, a majority of the early works that attempted to extend CNNs for video, fed the networks with raw frames. This makes for a much difficult learning problem. Jhuang et al. [51], proposed a biologically inspired model for action recognition in videos with a predefined set of spatio-temporal filters in the initial layer. Combined with a similar but spatial HMAX (Hierarchical model and X) model, Kuehne et al. [58] proposed spatial and temporal recognition streams. Ji et al. [52] addressed an end-to-end learning of the CNNs for videos for the first time using 3-D convolutions over a bunch of consecutive video frames. A more recent work by Karpathy et al. [54] proposes a set of techniques to fuse the appearance information present from a stack of consecutive frames in a video. However, the authors report that the net that processes individual frames performs on par with the net that operates on a stack of frames. This might suggest that the learned spatio-temporal filters are not suitable to capture the motion patterns efficiently.

A more suitable CNN model to represent videos is proposed in a contemporaneous work by Simonyan and Zisserman [82], which is called a two-stream network approach. Though the model in Kuehne et al. [58] is also a two stream model, the

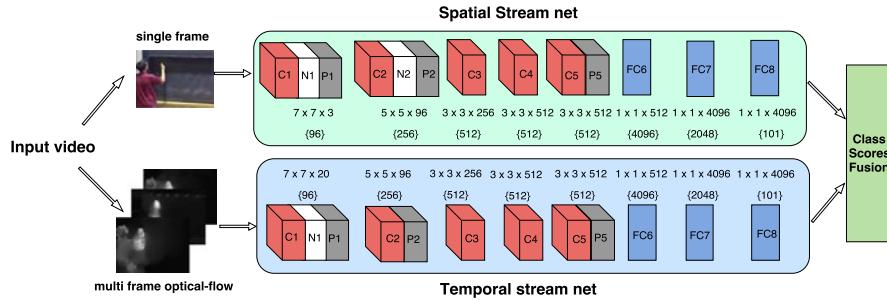


FIGURE 2.4

Two stream architecture for video classification from Simonyan and Zisserman [82].

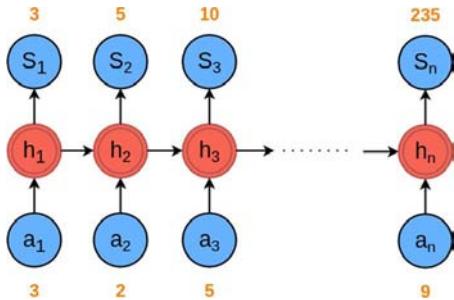
main difference is that the streams are shallow and implemented with hand-crafted models. The reason for the success of this approach is the natural ability of the videos to be separated into spatial and temporal components. The spatial component in the form of frames captures the appearance information like the objects present in the video. The temporal component in the form of motion (optical flow) across the frames captures the movement of the objects. These optical flow estimates can be obtained either from classical approaches [9] or deep-learned approaches [106].

This approach models the recognition system dividing into two parallel streams as depicted in Fig. 2.4. Each is implemented by a dedicated deep CNN, whose predictions are later fused. The net for the spatial stream is similar to the image recognition CNN and processes one frame at a time. However, the temporal stream takes the stacked optical flow of a bunch of consecutive frames as input and predicts the action. Both the nets are trained separately with the corresponding input. An alternative motion representation using the trajectory information similar to Wang et al. [101] is also observed to perform similar to optical flow.

The most recent methods that followed [82] have similar two-stream architecture. However, their contribution is to find the most active spatio-temporal volume for the efficient video representation. Inspired from the recent progress in the object detection in images, Georgia et al. [30] built action models from shape and motion cues. They start from the image proposals and select the motion salient subset of them and extract spatio-temporal features to represent the video using the CNNs.

Wang et al. [102] employ deep CNNs to learn discriminative feature maps and conduct trajectory constrained pooling to summarize into an effective video descriptor. The two streams operate in parallel extracting local deep features for the volumes centered around the trajectories.

In general, these multi-modal CNNs can be modified and extended to suit any other kind of modality like audio, text to complement the image data leading to a better representation of image content.

**FIGURE 2.5**

Toy RNN example – problem of sequence addition. The inputs and outputs are shown in blue. The red cells correspond to the hidden units. An unrolled version of the RNN is shown.

### 2.3.4 CNNS WITH RNNS

While CNNs have made remarkable progress in various tasks, they are not very suitable for learning sequences. Learning such patterns requires memory of previous states and feedback mechanisms which are not present in CNNs. RNNs are neural nets with at least one feedback connection. This looping structure enables the RNN to have an internal memory and to learn temporal patterns in data.

[Fig. 2.5](#) shows the unrolled version of a simple RNN applied to a toy example of sequence addition. The problem is defined as follows: Let  $a_t$  be a positive number, corresponding to the input at time  $t$ . The output at time  $t$  is given by

$$S_t = \sum_{i=1}^t a_i.$$

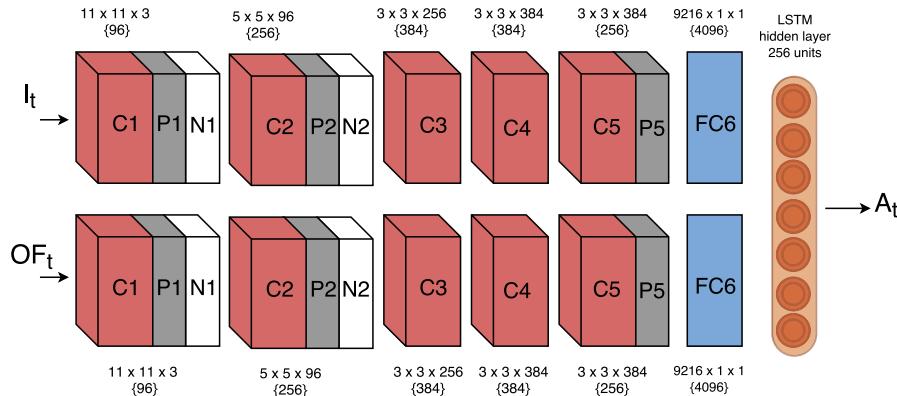
We consider a very simple RNN with just one hidden layer. The RNN can be described by equations below:

$$\begin{aligned} h_{t+1} &= f_h(W_{ih} \times a_t + W_{hh} \times h_t), \\ S_{t+1} &= f_o(W_{ho} \times h_{t+1}) \end{aligned}$$

where  $W_{ih}$ ,  $W_{hh}$ , and  $W_{ho}$  are learned weights, and  $f_h$  and  $f_o$  are nonlinearities. For the toy problem considered above, the weights learned would result in  $W_{ih} = W_{hh} = W_{ho} = 1$ . Let us consider the nonlinearity to be ReLu. The equations would then become

$$\begin{aligned} h_{t+1} &= \text{ReLu}(a_t + h_t), \\ S_{t+1} &= \text{ReLu}(h_{t+1}). \end{aligned}$$

Thus, as shown in [Fig. 2.5](#), the RNN stores previous inputs in memory and learns to predict the sum of the sequence up to the current time step  $t$ .

**FIGURE 2.6**

LRCN. A snapshot of the model at time  $t$ .  $I_t$  refers to the frame and  $OF_t$  the optical flow at  $t$ . The video is classified by averaging the output  $A_t$  over all  $t$ .

As with CNNs, recurrent neural networks have been trained with various back propagation techniques. These conventional methods, however, resulted in the *vanishing gradient problem*, i.e., the errors sent backward over the network, either grew very large or vanished leading to problems in convergence. In 1997, Hochreiter and Schmidhuber [43] introduced LSTM (Long Short Term Memory), which succeeded in overcoming the vanishing gradient problem, by introducing a novel architecture consisting of units called Constant Error Carousels. LSTMs were thus able to learn very deep RNNs and successfully remembered important events over long (thousands of steps) durations of time.

Over the next decade, LSTMs became the network of choice for several sequence learning problems, especially in the fields of speech and handwriting recognition [33,34]. In the sections that follow, we shall discuss applications of RNNs in various computer vision problems.

### 2.3.4.1 Action Recognition

Recognizing human actions from videos has long been a pivotal problem in the tasks of video understanding and surveillance. Actions, being events which take place over a finite length of time, are excellent candidates for a joint CNN–RNN model.

In particular, we discuss the model proposed by Donahue et al. [21]. They use raw *RGB* values as well as *optical flow* features to jointly train a variant of AlexNet combined with a single layer of LSTM (256 hidden units). Frames of the video are sampled, passed through the trained network, and classified individually. The final prediction is obtained by averaging across all the frames. A snapshot of this model at time  $t$  is shown in Fig. 2.6.

Yue-Hei Ng et al. [113] instead utilize a deep LSTM having five stacked layers of 512 units each following the CNN. Each video frame is fed to the CNN and then

processed by the multi-layer LSTM. A softmax classifier forms the final layer and performs action classification at each time step.

### 2.3.4.2 Image and Video Captioning

Another important aspect of scene understanding is the textual description of images and videos. Relevant textual description also helps complement image information, as well as form useful queries for retrieval.

RNNs have long been used for machine translation [8,16], where the task is to convert a sentence from one language to another. In these works, the common approach has been to use encoder RNNs to generate a latent vector representation of the sentence in the source language. A decoder RNN then transforms this latent vector into the sentence representation for the target language. This general framework can be used for image description as well, where the idea is to perform image-to-sentence ‘translation’ instead of between sentences of two languages. Vinyals et al. [98] developed such an end-to-end system called *Show and Tell*, by first encoding an image using a CNN and then using the encoded image as an input to an RNN, which generated sentences. An alternative way to approach the problem is using the multi-modal RNN (m-RNN) [67], which generates sentences by predicting the next word from the present word. To make this decision, image embeddings are used by RNN, making it multi-modal. This was further extended [66] by incorporating a special form of weight-sharing, enabling the network to learn novel visual concepts from images. One problem with the methods discussed above is that it is difficult to understand *why* the network produced the caption that it did. To this end, Xu et al. developed *Show, Attend and Tell*, which incorporates attention into the image description framework. Attention mechanisms act as gates to the input image. Given an image, an attention mechanism learns to select only a small spatial region in the image. Using this, the authors are able to localize nouns in a text description to objects in the image.

Similar to images, natural language description can also be obtained for videos. Considering a video as a collection of image frames, a simple technique would involve simply collecting image representations and use the set of representations for video description. Venugopalan et al. [96] use a mean-pooled CNN representation of image frames for a video. They train an LSTM to use this input to generate a description for the video. They further improve upon this task by developing S2VT [95], a stacked LSTM model which accounts for both the RGB as well as flow information available in videos, rather than just the RGB information used previously.

### 2.3.4.3 Visual Question Answering

A more comprehensive understanding of images should enable a system not only to make a statement about it, but also to answer questions related to it. Therefore answering questions based on visual concepts in an image is the next natural step for machine understanding algorithms. This requires the system to model both the textual question and the image representation, before generating an answer.

A critical question when designing such a system is that of combining visual and text (question) information in an efficient manner. Malinowski et al. [65] train

an LSTM which models the concatenation of a question and an answer. After the question is specified, the model is expected to predict the answer word-by-word. At each time step, visual information in the form of CNN features is also available to the LSTM. This system can, in principle, use information from both modalities at each time step. The disadvantage of this model is again that it is not possible to know *why* the network gave a particular answer for a given question. To overcome this, attention mechanism is used, as mentioned earlier for the image captioning example.

Yang et al. [111] develop a stack attention framework, where an image is queried multiple times and the answer is determined progressively. Different parts of the image are described by CNN features, and the question is described by an LSTM. The stacked attention layers use these descriptions to select regions of the image which may be relevant to the answer. These regions are progressively refined until the final attention layer, where the answer is predicted.

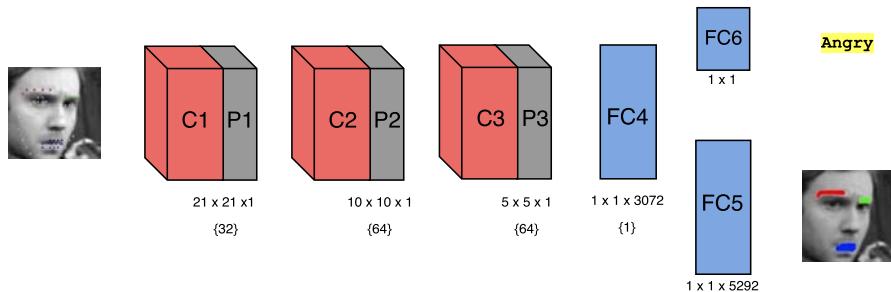
Wu et al. [107] use knowledge generated from attributes and image captions along with that mined from knowledge bases to perform visual question answering. They first train a CNN to generate attributes for a given image. These attributes are used to query the knowledge base. The retrieved information is collated and converted to a vector representation. The attributes are also used to generate image captions. The top five captions are also summarized to produce a combined representation. The attribute vector in combination with the caption vector and the knowledge base vector are used to initialize the LSTM. The question is then fed to the LSTM, which proceeds to generate the answer.

### 2.3.5 HYBRID LEARNING METHODS

#### 2.3.5.1 Multi-Task Learning

Multi-task learning is essentially a machine learning paradigm wherein the objective is to train the learning system to perform well on multiple tasks. Multi-task learning frameworks tend to exploit shared representations that exist among the tasks to obtain a better generalization performance than counterparts developed for a single task alone.

In CNNs, multi-task learning is realized using different approaches. One class of approaches utilize a multi-task loss function with hyper-parameters typically regulating the task losses. For example, Girshik et al. [29] employ a multi-task loss to train their network jointly for classification and bounding-box regression tasks thereby improving performance for object detection. Zhang et al. [118] propose a facial landmark detection network which adaptively weights auxiliary tasks (e.g., head pose estimation, gender classification, age estimation) to ensure that a task that is deemed not beneficial to accurate landmark detection is prevented from contributing to the network learning. Devries et al. [20] demonstrate improved performance for facial expression recognition task by designing a CNN for simultaneous landmark localization and facial expression recognition. A hallmark of these approaches is the division of tasks into primary task and auxiliary task(s) wherein the purpose of the latter is typically to improve the performance of the former (see Fig. 2.7). Note that in some

**FIGURE 2.7**

The facial expression recognition system of Devries et al. [20] which utilizes facial landmark (shown overlaid on the face toward the right of the image) recognition as an auxiliary task which helps improve performance on the main task of expression recognition.

cases, the auxiliary task may play an adversarial role in improving the performance. For instance, Ganin et al. [26] design a network for visual domain-adaptation wherein the performance of the auxiliary task (domain classification) is sought to be degraded to assist the main task of domain-adaptation.

Some approaches tend to have significant portions of the original network modified for multiple tasks. For instance, Sermanet et al. [80] replace pre-trained layers of a net originally designed to provide spatial (per-pixel) classification maps with a regression network and fine-tune the resulting net to achieve simultaneous classification, localization, and detection of scene objects.

Another class of multi-task approaches tend to have task-specific sub-networks as a characteristic feature of CNN design. Li et al. [60] utilize separate sub-networks for the joint point regression and body part detection tasks. Kruthiventi et al. [85] proposed a deep fully convolutional network with a branched architecture for simultaneously predicting eye fixations and segmenting salient objects in an image. Wang et al. [104] adopt a serially stacked design wherein a localization sub-CNN and the original object image are fed into a segmentation sub-CNN to generate its object bounding box and extract its segmentation mask. To solve an unconstrained word image recognition task, Jaderberg et al. [49] propose an architecture consisting of a character sequence CNN and an N-gram encoding CNN which act on an input image in parallel and whose outputs are utilized along with a CRF model to recognize the text content present within the image. Typically, the task-specific sub-networks do not communicate directly. However, there are approaches where cross-network connections exist. These connections are designed to achieve an optimal trade off between shared and task-specific aspects of the overall learning framework [68,62].

### 2.3.5.2 Similarity Learning

Apart from classification, CNNs can also be used for tasks like metric learning. Rather than asking the CNN to identify objects, we can instead ask it to verify

**Table 2.1** A curated list of software for Convolutional Neural Networks

Software	Interfaces provided	AutoDiff
Caffe [53]	Python, MATLAB, C++, command-line	No
Theano [91]	Python	Yes
Tensorflow [5]	C++, Python	Yes
Torch [4]	Torch, C	Yes (non-native)
CNTK [6]	C++, command-line	No
MatConvNet [94]	MATLAB	No
Chainer [1]	Python	No

whether two images contain the same object or not. In other words, we ask the CNN to learn which images are *similar*, and which are not. Image retrieval is one application where such questions are routinely asked.

Structurally, Siamese networks resemble two-stream networks discussed previously. However, the difference here is that both ‘streams’ have identical weights. Siamese networks consist of two separate (but identical) networks, where two images are fed in as input. Their activations are combined at a later layer, and the output of the network consists of a single number, or a *metric*, which is a notion of distance between the images. Training is done so that images which are considered to be similar have a lower output score than images which are considered different. Bromley et al. [13] introduced the idea of Siamese networks and used it for signature verification. Later on, Chopra et al. [17] extended it for face verification. Zagoruyko et al. [114] further extended and generalized this to learning similarity between image patches.

## 2.4 SOFTWARE FOR DEEP LEARNING

We would like to end this chapter with a short discussion on the various software packages available for deep learning. **Table 2.1** provides a non-exhaustive list.

For computer vision applications, Caffe seems to be the most widely used software framework. It is written in C++ and is often used via the command line/Python interfaces. The design of this framework makes it easy to use pre-trained neural networks out of the box.

In the general deep learning community, Theano, Torch, and Tensorflow seem to be more popular. These frameworks are designed to make it easy to modify low level details of neural network models. While Theano and Tensorflow are used in Python, Torch is used in Lua. There also exist a few wrapper libraries for Theano, such as Lasagne [3] and Keras [2], which provide commonly used functionalities as well as some support for pre-trained models.

In this context, Automatic Differentiation (AutoDiff) [71] is an important feature to have in such framework. When specifying new layers/regularizers for neural networks, one also needs to specify how the gradient through that layer would look like.

AutoDiff automatically computes the derivative of complicated functions in closed form using the definitions of derivatives of its primitives.

---

## REFERENCES

1. Chainer, <http://chainer.org/>, 2016, accessed: 17 May 2016.
2. Keras, <http://keras.io/>, 2016, accessed: 17 May 2016.
3. Lasagne, <http://lasagne.readthedocs.io/>, 2016, accessed: 17 May 2016.
4. Torch, <http://torch.ch/>, 2016, accessed: 17 May 2016.
5. M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G.S. Corrado, A. Davis, J. Dean, M. Devin, et al., Tensorflow: large-scale machine learning on heterogeneous distributed systems, arXiv:1603.04467, 2016.
6. A. Agarwal, E. Akchurin, C. Basoglu, G. Chen, S. Cyphers, J. Droppo, A. Eversole, B. Guenter, M. Hillebrand, R. Hoens, et al., An Introduction to Computational Networks and the Computational Network Toolkit, Technical report, 2014.
7. A. Babenko, A. Slesarev, A. Chigorin, V. Lempitsky, Neural codes for image retrieval, in: D. Fleet, T. Pajdla, B. Schiele, T. Tuytelaars (Eds.), Computer Vision – ECCV 2014, in: Lect. Notes Comput. Sci., vol. 8689, Springer International Publishing, 2014, pp. 584–599.
8. D. Bahdanau, K. Cho, Y. Bengio, Neural machine translation by jointly learning to align and translate, arXiv:1409.0473, 2014.
9. S. Baker, I. Matthews, Lucas–Kanade 20 years on: a unifying framework, Int. J. Comput. Vis. 56 (3) (2004) 221–255.
10. Y. Bengio, Learning deep architectures for AI, Found. Trends Mach. Learn. 2 (1) (2009) 1–127.
11. C.M. Bishop, Pattern Recognition and Machine Learning (Information Science and Statistics), Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.
12. L. Bottou, Large-scale machine learning with stochastic gradient descent, in: Proceedings of COMPSTAT’2010, Springer, 2010, pp. 177–186.
13. J. Bromley, J.W. Bentz, L. Bottou, I. Guyon, Y. LeCun, C. Moore, E. Säckinger, R. Shah, Signature verification using a “siamese” time delay neural network, Int. J. Pattern Recognit. Artif. Intell. 7 (04) (1993) 669–688.
14. K. Chatfield, K. Simonyan, A. Vedaldi, A. Zisserman, Return of the devil in the details: delving deep into convolutional nets, in: Proceedings of the British Machine Vision Conference, Nottingham, UK, BMVA Press, 2014.
15. L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, A.L. Yuille, Semantic image segmentation with deep convolutional nets and fully connected CRFS, arXiv:1412.7062, 2014.
16. K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, Y. Bengio, Learning phrase representations using RNN encoder–decoder for statistical machine translation, arXiv:1406.1078, 2014.
17. S. Chopra, R. Hadsell, Y. LeCun, Learning a similarity metric discriminatively, with application to face verification, in: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 1, CVPR 2005, San Diego, CA, USA, IEEE, 2005, pp. 539–546.
18. C. Couprie, C. Farabet, L. Najman, Y. LeCun, Indoor semantic segmentation using depth information, arXiv:1301.3572, 2013.

19. N. Dalal, B. Triggs, Histograms of oriented gradients for human detection, in: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 1, CVPR 2005, San Diego, CA, USA, IEEE, 2005, pp. 886–893.
20. T. Devries, K. Biswaranjan, G.W. Taylor, Multi-task learning of facial landmarks and expression, in: 2014 Canadian Conference on Computer and Robot Vision (CRV), IEEE, 2014, pp. 98–103.
21. J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, T. Darrell, Long-term recurrent convolutional networks for visual recognition and description, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 2625–2634.
22. J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, T. Darrell, Decaf: a deep convolutional activation feature for generic visual recognition, arXiv:1310.1531, 2013.
23. J. Duchi, E. Hazan, Y. Singer, Adaptive subgradient methods for online learning and stochastic optimization, *J. Mach. Learn. Res.* 12 (2011) 2121–2159.
24. C. Farabet, C. Couprie, L. Najman, Y. LeCun, Learning hierarchical features for scene labeling, *IEEE Trans. Pattern Anal. Mach. Intell.* 35 (8) (2013) 1915–1929.
25. K. Fukushima, Neocognitron: a self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position, *Biol. Cybern.* 36 (4) (1980) 193–202.
26. Y. Ganin, V. Lempitsky, Unsupervised domain adaptation by backpropagation, in: Proceedings of the 32nd International Conference on Machine Learning, 2015, pp. 1180–1189.
27. R. Girshick, Fast R-CNN, in: Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 1440–1448.
28. R. Girshick, J. Donahue, T. Darrell, J. Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, in: 2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Columbus, OH, USA, IEEE, 2014, pp. 580–587.
29. R.B. Girshick, Fast R-CNN, arXiv:1504.08083, 2015.
30. G. Gkioxari, J. Malik, Finding action tubes, in: CVPR, 2015.
31. Y. Gong, L. Wang, R. Guo, S. Lazebnik, Multi-scale orderless pooling of deep convolutional activation features, in: D. Fleet, T. Pajdla, B. Schiele, T. Tuytelaars (Eds.), Computer Vision – ECCV 2014, in: Lect. Notes Comput. Sci., vol. 8695, Springer International Publishing, 2014, pp. 392–407.
32. I.J. Goodfellow, D. Warde-Farley, M. Mirza, A. Courville, Y. Bengio, Maxout networks, arXiv:1302.4389, 2013.
33. A. Graves, M. Liwicki, S. Fernández, R. Bertolami, H. Bunke, J. Schmidhuber, A novel connectionist system for unconstrained handwriting recognition, *IEEE Trans. Pattern Anal. Mach. Intell.* 31 (5) (2009) 855–868.
34. A. Graves, A.-r. Mohamed, G. Hinton, Speech recognition with deep recurrent neural networks, in: 2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Vancouver, Canada, IEEE, 2013, pp. 6645–6649.
35. S. Gupta, R. Girshick, P. Arbelaez, J. Malik, Learning rich features from RGB-D images for object detection and segmentation, in: ECCV, 2014.
36. B. Hariharan, P. Arbeláez, R. Girshick, J. Malik, Hypercolumns for object segmentation and fine-grained localization, arXiv:1411.5752, 2014.
37. K. He, X. Zhang, S. Ren, J. Sun, Spatial pyramid pooling in deep convolutional networks for visual recognition, arXiv:1406.4729, 2014.
38. K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, arXiv:1512.03385, 2015.

39. K. He, X. Zhang, S. Ren, J. Sun, Delving deep into rectifiers: surpassing human-level performance on ImageNet classification, arXiv:1502.01852, 2015.
40. G.E. Hinton, Training products of experts by minimizing contrastive divergence, *Neural Comput.* 14 (8) (2002) 1771–1800.
41. G.E. Hinton, S. Osindero, Y.-W. Teh, A fast learning algorithm for deep belief nets, *Neural Comput.* 18 (7) (2006) 1527–1554.
42. G.E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, R.R. Salakhutdinov, Improving neural networks by preventing co-adaptation of feature detectors, arXiv:1207.0580, 2012.
43. S. Hochreiter, J. Schmidhuber, Long short-term memory, *Neural Comput.* 9 (8) (1997) 1735–1780.
44. K. Hornik, Approximation capabilities of multilayer feedforward networks, *Neural Netw.* 4 (2) (1991) 251–257.
45. J. Hosang, R. Benenson, P. Dollár, B. Schiele, What makes for effective detection proposals?, *IEEE Trans. Pattern Anal. Mach. Intell.* 38 (4) (2016) 814–830.
46. X. Huang, C. Shen, X. Boix, Q. Zhao, Salicon: reducing the semantic gap in saliency prediction by adapting deep neural networks, in: *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 262–270.
47. D.H. Hubel, T.N. Wiesel, Receptive fields, binocular interaction and functional architecture in the cat's visual cortex, *J. Physiol.* 160 (1) (1962) 106.
48. S. Ioffe, C. Szegedy, Batch normalization: accelerating deep network training by reducing internal covariate shift, arXiv:1502.03167, 2015.
49. M. Jaderberg, K. Simonyan, A. Vedaldi, A. Zisserman, Deep structured output learning for unconstrained text recognition, arXiv:1412.5903, 2014.
50. H. Jegou, F. Perronnin, M. Douze, J. Sanchez, P. Perez, C. Schmid, Aggregating local image descriptors into compact codes, *IEEE Trans. Pattern Anal. Mach. Intell.* 34 (9) (2012) 1704–1716.
51. H. Jhuang, T. Serre, L. Wolf, T. Poggio, A biologically inspired system for action recognition, in: *IEEE 11th International Conference on Computer Vision*, ICCV 2007, 2007, pp. 1–8.
52. S. Ji, W. Xu, M. Yang, K. Yu, 3d convolutional neural networks for human action recognition, *IEEE Trans. Pattern Anal. Mach. Intell.* 35 (1) (2013) 221–231.
53. Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, T. Darrell, Caffe: convolutional architecture for fast feature embedding, arXiv:1408.5093, 2014.
54. A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, L. Fei-Fei, Large-scale video classification with convolutional neural networks, in: *2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Columbus, OH, USA, IEEE, 2014, pp. 1725–1732.
55. D. Kingma, J. Ba, Adam: a method for stochastic optimization, arXiv:1412.6980, 2014.
56. A. Krizhevsky, I. Sutskever, G.E. Hinton, ImageNet classification with deep convolutional neural networks, in: F. Pereira, C. Burges, L. Bottou, K. Weinberger (Eds.), *Advances in Neural Information Processing Systems*, vol. 25, Curran Associates, Inc., 2012, pp. 1097–1105.
57. S.S. Kruthiventi, K. Ayush, R.V. Babu, DeepFix: a fully convolutional neural network for predicting human eye fixations, arXiv:1510.02927, 2015.
58. H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, T. Serre, HMDB: a large video database for human motion recognition, in: *2011 IEEE International Conference on Computer Vision (ICCV)*, Barcelona, Spain, IEEE, 2011, pp. 2556–2563.

59. Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, *Proc. IEEE* 86 (11) (1998) 2278–2324.
60. S. Li, Z.-Q. Liu, A. Chan, Heterogeneous multi-task learning for human pose estimation with deep convolutional neural network, *Int. J. Comput. Vis.* 113 (1) (2015) 19–36.
61. J. Long, E. Shelhamer, T. Darrell, Fully convolutional networks for semantic segmentation, in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3431–3440.
62. M. Long, Y. Cao, J. Wang, M.I. Jordan, Learning transferable features with deep adaptation networks, in: *Proceedings of the 32nd International Conference on Machine Learning*, ICML 2015, Lille, France, 6–11 July 2015, 2015, pp. 97–105.
63. D.G. Lowe, Distinctive image features from scale-invariant keypoints, *Int. J. Comput. Vis.* 60 (2) (2004) 91–110.
64. A.L. Maas, A.Y. Hannun, A.Y. Ng, Rectifier nonlinearities improve neural network acoustic models, in: *Proc. ICML*, vol. 30, 2013.
65. M. Malinowski, M. Rohrbach, M. Fritz, Ask your neurons: a neural-based approach to answering questions about images, in: *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1–9.
66. J. Mao, X. Wei, Y. Yang, J. Wang, Z. Huang, A.L. Yuille, Learning like a child: fast novel visual concept learning from sentence descriptions of images, in: *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 2533–2541.
67. J. Mao, W. Xu, Y. Yang, J. Wang, A. Yuille, Deep captioning with multimodal recurrent neural networks (M-RNN), arXiv:1412.6632, 2014.
68. I. Misra, A. Shrivastava, A. Gupta, M. Hebert, Cross-stitch networks for multi-task learning, in: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.
69. K. Mopuri, R. Babu, Object level deep feature pooling for compact image representation, in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2015, pp. 62–70.
70. V. Nair, G.E. Hinton, Rectified linear units improve restricted Boltzmann machines, in: *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, 2010, pp. 807–814.
71. R.D. Neidinger, Introduction to automatic differentiation and Matlab object-oriented programming, *SIAM Rev.* 52 (3) (2010) 545–563.
72. Y. Nesterov, A method of solving a convex programming problem with convergence rate  $O(1/k^2)$ , *Sov. Math. Dokl.* 27 (1983) 372–376.
73. H. Noh, S. Hong, B. Han, Learning deconvolution network for semantic segmentation, in: *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1520–1528.
74. B.T. Polyak, Some methods of speeding up the convergence of iteration methods, *USSR Comput. Math. Math. Phys.* 4 (5) (1964) 1–17.
75. A.S. Razavian, H. Azizpour, J. Sullivan, S. Carlsson, CNN features off-the-shelf: an astounding baseline for recognition, in: *2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, Columbus, OH, USA, IEEE, 2014, pp. 512–519.
76. S. Ren, K. He, R. Girshick, J. Sun, Faster R-CNN: towards real-time object detection with region proposal networks, in: *Advances in Neural Information Processing Systems*, 2015, pp. 91–99.

77. H.A. Rowley, S. Baluja, T. Kanade, Neural network-based face detection, *IEEE Trans. Pattern Anal. Mach. Intell.* 20 (1) (1998) 23–38.
78. D.E. Rumelhart, G.E. Hinton, R.J. Williams, Learning representations by back-propagating errors, *Cogn. Model.* 5 (1988) 3.
79. O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. Berg, L. Fei-Fei, ImageNet large scale visual recognition challenge, *Int. J. Comput. Vis.* 115 (3) (2015) 211–252.
80. P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, Y. LeCun, Overfeat: integrated recognition, localization and detection using convolutional networks, arXiv:1312.6229, 2013.
81. P. Sermanet, K. Kavukcuoglu, S. Chintala, Y. LeCun, Pedestrian detection with unsupervised multi-stage feature learning, in: 2013 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Portland, USA, IEEE, 2013, pp. 3626–3633.
82. K. Simonyan, A. Zisserman, Two-stream convolutional networks for action recognition in videos, in: Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, K. Weinberger (Eds.), *Adv. Neural Inf. Process. Syst.*, vol. 27, 2014, pp. 568–576.
83. K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, arXiv:1409.1556, 2014.
84. R. Socher, B. Huval, B. Bath, C.D. Manning, A.Y. Ng, Convolutional-recursive deep learning for 3D object classification, in: F. Pereira, C. Burges, L. Bottou, K. Weinberger (Eds.), *Adv. Neural Inf. Process. Syst.*, vol. 25, 2012, pp. 656–664.
85. Srinivas S.S. Kruthiventi, J.H.D. Vennela Gudisa, R.V. Babu, Saliency unified: a deep architecture for simultaneous eye fixation prediction and salient object segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016.
86. R.K. Srivastava, K. Greff, J. Schmidhuber, Training very deep networks, in: Advances in Neural Information Processing Systems, 2015, pp. 2368–2376.
87. R.K. Srivastava, J. Masci, S. Kazeroonian, F. Gomez, J. Schmidhuber, Compete to compute, in: Advances in Neural Information Processing Systems, 2013, pp. 2310–2318.
88. I. Sutskever, J. Martens, G. Dahl, G. Hinton, On the importance of initialization and momentum in deep learning, in: Proceedings of the 30th International Conference on Machine Learning (ICML-13), 2013, pp. 1139–1147.
89. C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, arXiv:1409.4842, 2014.
90. Y. Taigman, M. Yang, M. Ranzato, L. Wolf, DeepFace: closing the gap to human-level performance in face verification, in: 2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Columbus, OH, USA, IEEE, 2014, pp. 1701–1708.
91. Theano Development Team, Theano: a Python framework for fast computation of mathematical expressions, arXiv:1605.02688, 2016.
92. J.R.R. Uijlings, K.E.A. van de Sande, T. Gevers, A.W.M. Smeulders, Selective search for object recognition, *Int. J. Comput. Vis.* 104 (2) (2013) 154–171.
93. R. Vaillant, C. Monrocq, Y. Le Cun, Original approach for the localisation of objects in images, *IEE Proc., Vis. Image Signal Process.* 141 (4) (1994) 245–250.
94. A. Vedaldi, K. Lenc, MatConvNet – convolutional neural networks for MATLAB, 2015.
95. S. Venugopalan, M. Rohrbach, J. Donahue, R. Mooney, T. Darrell, K. Saenko, Sequence to sequence – video to text, in: The IEEE International Conference on Computer Vision (ICCV), 2015.
96. S. Venugopalan, H. Xu, J. Donahue, M. Rohrbach, R. Mooney, K. Saenko, Translating videos to natural language using deep recurrent neural networks, arXiv:1412.4729, 2014.

97. P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, P.-A. Manzagol, Stacked denoising autoencoders: learning useful representations in a deep network with a local denoising criterion, *J. Mach. Learn. Res.* 11 (2010) 3371–3408.
98. O. Vinyals, A. Toshev, S. Bengio, D. Erhan, Show and tell: a neural image caption generator, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 3156–3164.
99. L. Wan, M. Zeiler, S. Zhang, Y.L. Cun, R. Fergus, Regularization of neural networks using DropConnect, in: Proceedings of the 30th International Conference on Machine Learning (ICML-13), 2013, pp. 1058–1066.
100. A. Wang, J. Lu, G. Wang, J. Cai, T.-J. Cham, Multi-modal unsupervised feature learning for RGB-D scene labeling, in: D. Fleet, T. Pajdla, B. Schiele, T. Tuytelaars (Eds.), Computer Vision – ECCV 2014, Zurich, Switzerland, in: *Lect. Notes Comput. Sci.*, vol. 8693, Springer, 2014, pp. 453–467.
101. H. Wang, C. Schmid, Action recognition with improved trajectories, in: IEEE International Conference on Computer Vision, Sydney, Australia, 2013.
102. L. Wang, Y. Qiao, X. Tang, Action recognition with trajectory-pooled deep-convolutional descriptors, in: CVPR, 2015, pp. 4305–4314.
103. S. Wang, C. Manning, Fast dropout training, in: Proceedings of the 30th International Conference on Machine Learning (ICML-13), 2013, pp. 118–126.
104. X. Wang, L. Zhang, L. Lin, Z. Liang, W. Zuo, Deep joint task learning for generic object extraction, arXiv:1502.00743, 2015.
105. Y. Wei, W. Xia, J. Huang, B. Ni, J. Dong, Y. Zhao, S. Yan, CNN: single-label to multi-label, arXiv:1406.5726, 2014.
106. P. Weinzaepfel, J. Revaud, Z. Harchaoui, C. Schmid, DeepFlow: large displacement optical flow with deep matching, in: 2013 IEEE International Conference on Computer Vision (ICCV), Portland, USA, IEEE, 2013, pp. 1385–1392.
107. Q. Wu, C. Shen, A.v.d. Hengel, P. Wang, A. Dick, Image captioning and visual question answering based on attributes and their related external knowledge, arXiv:1603.02814, 2016.
108. L. Xi, L. Zhao, L. Wei, M. Yang, F. Wu, Y. Zhuang, H. Ling, J. Wang, DeepSaliency: multi-task deep neural network model for salient object detection, arXiv:1510.05484, 2015.
109. B. Yang, J. Yan, Z. Lei, S.Z. Li, Craft objects from images, arXiv:1604.03239, 2016.
110. J. Yang, Y.-G. Jiang, A.G. Hauptmann, C.-W. Ngo, Evaluating bag-of-visual-words representations in scene classification, in: Proceedings of the International Workshop on Workshop on Multimedia Information Retrieval, ACM, New York, NY, USA, 2007, pp. 197–206.
111. Z. Yang, X. He, J. Gao, L. Deng, A. Smola, Stacked attention networks for image question answering, arXiv:1511.02274, 2015.
112. J. Yosinski, J. Clune, Y. Bengio, H. Lipson, How transferable are features in deep neural networks?, in: Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, K. Weinberger (Eds.), *Advances in Neural Information Processing Systems*, vol. 27, Curran Associates, Inc., 2014, pp. 3320–3328.
113. J. Yue-Hei Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, G. Toderici, Beyond short snippets: deep networks for video classification, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 4694–4702.
114. S. Zagoruyko, N. Komodakis, Learning to compare image patches via convolutional neural networks, arXiv:1504.03641, 2015.

115. M. Zeiler, R. Fergus, Visualizing and understanding convolutional networks, in: D. Fleet, T. Pajdla, B. Schiele, T. Tuytelaars (Eds.), Computer Vision – ECCV 2014, in: Lect. Notes Comput. Sci., vol. 8689, Springer International Publishing, 2014, pp. 818–833.
116. M.D. Zeiler, ADADELTA: an adaptive learning rate method, arXiv:1212.5701, 2012.
117. M.D. Zeiler, R. Fergus, Stochastic pooling for regularization of deep convolutional neural networks, arXiv:1301.3557, 2013.
118. Z. Zhang, P. Luo, C.C. Loy, X. Tang, Learning and transferring multi-task deep representation for face alignment, arXiv:1408.3967, 2014.
119. S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, P. Torr, Conditional random fields as recurrent neural networks, arXiv:1502.03240, 2015.

PART

# Medical Image Detection and Recognition

2

This page intentionally left blank

# Efficient Medical Image Parsing

# 3

Florin C. Ghesu<sup>\*,†</sup>, Bogdan Georgescu<sup>\*</sup>, Joachim Hornegger<sup>†</sup>

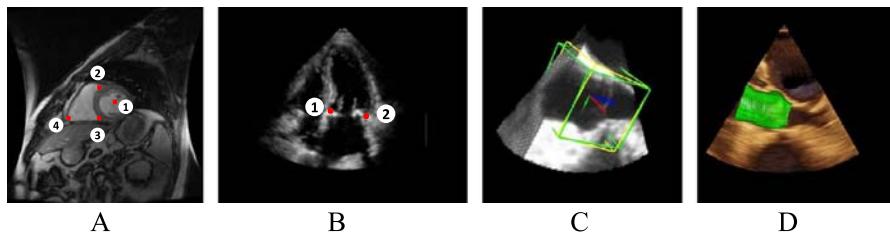
Siemens Medical Solutions USA, Inc., Princeton, NJ, United States<sup>\*</sup> Friedrich-Alexander University Erlangen-Nürnberg, Erlangen, Germany<sup>†</sup>

## CHAPTER OUTLINE

<b>3.1</b>	<b>Introduction</b>	55
<b>3.2</b>	<b>Background and Motivation</b>	57
3.2.1	Object Localization and Segmentation: Challenges	58
<b>3.3</b>	<b>Methodology</b>	58
3.3.1	Problem Formulation	58
3.3.2	Sparse Adaptive Deep Neural Networks	59
3.3.3	Marginal Space Deep Learning	61
3.3.3.1	Cascaded Negative Filtering	63
3.3.3.2	Nonrigid Deformation Estimation	63
3.3.4	An Artificial Agent for Image Parsing	64
3.3.4.1	Cognitive Modeling Using Deep Reinforcement Learning	65
3.3.4.2	Learning to Scan for Anatomical Objects	67
<b>3.4</b>	<b>Experiments</b>	71
3.4.1	Anatomy Detection and Segmentation in 3D	71
3.4.1.1	Dataset	71
3.4.1.2	Model Selection and Training Parameters	71
3.4.1.3	Evaluation	71
3.4.1.4	Additional Experiments and Discussion	73
3.4.2	Landmark Detection in 2D and 3D	74
3.4.2.1	Datasets	74
3.4.2.2	Evaluation: Learning to Parse	75
<b>3.5</b>	<b>Conclusion</b>	77
<b>Disclaimer</b>		78
<b>References</b>		78

## 3.1 INTRODUCTION

Powerful data representations drive the performance of many machine learning algorithms [1]. Designing features that generate such representations to effectively

**FIGURE 3.1**

Example images displaying parsed anatomical objects. Figures A and B show anatomical landmarks in cardiac magnetic resonance and ultrasound images, while C and D show the bounding box and boundary mesh for the aortic root in 3D ultrasound.

capture the information encoded in the given data is a particularly difficult task [2–5]. In practice, this requires complex data preprocessing pipelines that do not generalize well between different image modalities or learning problems. The reason for that is that most of these systems are manually engineered for specific applications and rely exclusively on human ingenuity to disentangle and understand prior information hidden in the data in order to design the required features and discover intrinsic information about the given task [1].

In the context of volumetric image parsing, machine learning is used for anatomy detection and organ segmentation [5,6] (see Fig. 3.1). Here, the task of feature engineering becomes increasingly complex since the feature extraction is typically performed under challenging transformations such as arbitrary orientations or scales. Moreover, for robust parameter estimation, scanning the parameter space exhaustively is not feasible given the exponential increase in the size of the space with respect to the space dimensionality, i.e. the number of considered transformation parameters.

We overcome all these challenges by proposing *Marginal Space Deep Learning* (MSDL), a feature-learning-based framework to support the efficient parsing of arbitrary anatomical structures. For this we formulate a two-step approach using deep learning (DL) as a powerful solution for joint feature learning and task learning in each step: object localization and boundary estimation. The framework relies on the computational advantages of scanning hierarchical parametric spaces through the mechanism of marginal space learning [5]. In the case of a restricted affine transformation, we estimate first the location (3D space), then the location and orientation (6D space), and finally the complete transformation including the anisotropic scaling (9D space). Given the rigid transformation of the object we propose a deep-learning-based active shape model (ASM) [7] to guide the shape deformation starting from the mean shape. For a defined object parametrization, the system learns classifiers in marginal parameter spaces thereby capturing the appearance of the object under specific transformations. We present two principled solutions for this, one based on exhaustive hypotheses scanning using advanced cascade filtering and sparse feature

sampling, and the second following a more intelligent search scheme, learning optimal trajectories in parameter space that lead to the target location describing the actual object transformation.

The first part of this chapter is dedicated to the scanning-based solution which we published in [8,9]. As exhaustive scanning comes at a high computational cost, the current applications of state-of-the-art DL architectures are focused on pixel(voxel)-wise classification in 2D or 2.5D data, with no generic extension supporting the parsing of large volumetric images. Capturing the complex appearance of 3D structures and ensuring the efficient scanning of high-dimensional parameter spaces are not straightforward given that the size of the parameter space increases exponentially with respect to the number of transformation parameters. To account for this we propose cascaded *sparse adaptive deep neural networks* (SADNN) to learn parametrized representations from 3D medical image modalities and enable the efficient scanning of large parameter spaces. We propose to use sparsity as a means to simplify the network and adaptively focus its sampling pattern on context-rich parts of the input by explicitly discarding neural connections. Note that this is different from typical weight-dropping regularization approaches like dropout [10]. We validate this solution on the problem of detecting and segmenting the aortic heart valve using a large 3D ultrasound dataset.

In the second part of the chapter, we present an alternative solution first introduced in [11], which unifies the learning of the appearance model and the object search to a behavioral problem for an artificial agent. Inspired by the fundamental work of Mnih et al. [12], we leverage state-of-the-art representation learning techniques through deep learning [13,14] and powerful solutions for generic behavior learning through reinforcement learning [15,16,12] to create a model encapsulating a cognitive-like learning process covering both the appearance of the object and search strategy. Initial results on the task of landmark detection prove that this type of approach is very promising in terms of accuracy, robustness, and particularly speed by avoiding typical inefficient scanning routines.

---

## 3.2 BACKGROUND AND MOTIVATION

Medical image parsing subsumes the robust recognition, detection, and segmentation of objects which proves to be a particularly difficult task for arbitrary 3D anatomical structures considering the variance in location, the nonrigid nature of the shape, as well as the differences in anatomy among different cases [5]. Many solutions focus mainly on the segmentation task, for example, Active Shape Models [17,7] and deformable models [18,19]. Nonetheless, in order to achieve an accurate automatic segmentation of arbitrary anatomical structures, a robust and efficient solution is required for localizing the object of interest.

### 3.2.1 OBJECT LOCALIZATION AND SEGMENTATION: CHALLENGES

Initially introduced in the 2D context [4,3], machine learning can be used for the efficient and robust localization of objects. In this context object localization is formulated as a classification problem over patches of image intensities extracted from transformed boxes defined in a parametric space  $\mathcal{U}$ . Typically a classifier is learned in this space using discrete positive and negative hypotheses  $h \in \mathcal{U}$ . During testing the classifier is applied to exhaustively scan the parameter space, an effort which grows exponentially with the dimensionality of the space. Extending the logic to 3D is however not straightforward. Let us start from the example of a restricted affine transformation. This type of rigid transformation is defined by nine parameters spanning a 9D space  $\mathcal{U} = (\mathcal{U}_T, \mathcal{U}_R, \mathcal{U}_S)$ . Three parameters are required to define the location  $\vec{T} = (t_x, t_y, t_z)$ ,  $\vec{T} \in \mathcal{U}_T$ , three to capture the orientation  $\vec{R} = (\phi_x, \phi_y, \phi_z)$ ,  $\vec{R} \in \mathcal{U}_R$ , and three to define the anisotropic scale  $\vec{S} = (s_x, s_y, s_z)$ ,  $\vec{S} \in \mathcal{U}_S$ . Even with a very coarse discretization of  $d = 10$  possible outcomes for each parameter, the number of hypotheses will be as high as  $d^9 = 1,000,000,000$ , virtually impossible to evaluate on any current consumer machine. Using statistical shape modeling, also the nonrigid shape of the object can be represented parametrically by considering the coefficients  $c_1, c_2, \dots, c_K$  of the major deformation modes, with  $K$  sufficiently large to closely approximate the true shape boundary. In this case the parameter space expands to  $\mathcal{U} = (\mathcal{U}_T, \mathcal{U}_R, \mathcal{U}_S, c_1, c_2, \dots, c_K)$ , leading to an explosion in the number of sample hypotheses. In addition, focusing only on the feature extraction in each of these spaces, we face the challenge of ensuring an efficient feature computation also under challenging transformations such as arbitrary orientations or scales, without explicitly transforming the data. All these challenges motivate a drive toward alternative non-learning based boundary delineation methods (see, for example, [7,17,20,21,19,18,22,23]) since they do not require large image databases for the model estimation. However, such methods typically suffer from a series of limitations in terms of robustness to noise and generalization, for which machine learning lends itself as an elegant solution.

---

## 3.3 METHODOLOGY

In this section we present a solution to these challenges, a novel feature-learning-based framework for parsing volumetric images split in a two-stage approach: anatomical object localization and nonrigid shape estimation.

### 3.3.1 PROBLEM FORMULATION

In our model we reformulate the detection and segmentation of an object to a patch-wise classification task described by a set of  $m$  parametrized input patches  $\vec{X}$  (i.e. observations as regions of interest, image intensities extracted within translated, rotated and scaled boxes) with a corresponding set of class assignments  $\vec{y}$ , with  $y_i = 1$  if the sought anatomical structure is contained in the patch and  $y_i = 0$  otherwise,

$1 \leq i \leq m$ . In this work, the classifiers are fully connected neural networks, meaning that the size of the filters is equal to the size of the underlying representations. We can define a deep fully connected DNN with the parameters  $(\vec{w}, \vec{b})$ , where  $\vec{w} = (\vec{w}_1, \vec{w}_2, \dots, \vec{w}_n)^\top$  represents the parameters of all  $n$  concatenated kernels over the layers of the network, i.e. the weighted connections between neurons, and  $\vec{b}$  encodes the biases of the neurons. The response of a neuron indexed  $k$  in the network is defined as:

$$o_k = \delta(x_k^\top w_k + b_k), \quad (3.1)$$

where  $\delta$  represents a nonlinear activation function,  $w_k$  the weights of incoming connections,  $x_k$  the activations of the connected neurons from the previous layer and  $b_k$  the bias of the neuron.

The activation function  $\delta$  is used as a nonlinear operator to synthesize the input information. Possible alternatives are the rectified linear units (ReLU) [24], the hyperbolic tangent or the sigmoid function. For our experiments we use the sigmoid function defined as  $\delta(y) = 1/(1 + e^{-y})$ , building through our network a multivariate logistic regression model for classification. We define the network response function  $\mathcal{R}(\cdot; \vec{w}, \vec{b})$  as a probabilistic approximation of the supervised class label for any given example,  $\hat{y}^{(i)} = \mathcal{R}(x^{(i)}; \vec{w}, \vec{b})$ , where  $1 \leq i \leq m$ . This is equivalent to estimating the conditional probability density function  $p(y^{(i)}|x^{(i)}; \vec{w}, \vec{b})$ . Since the learning setup is fully supervised and the input observations are independent, we can use the Maximum Likelihood Estimation (MLE) method:

$$\left(\hat{\vec{w}}, \hat{\vec{b}}\right) = \arg \max_{\vec{w}, \vec{b}} \mathcal{L}(\vec{w}, \vec{b}; \vec{X}) = \arg \max_{\vec{w}, \vec{b}} \prod_{i=1}^m p(y^{(i)}|\vec{x}^{(i)}; \vec{w}, \vec{b}), \quad (3.2)$$

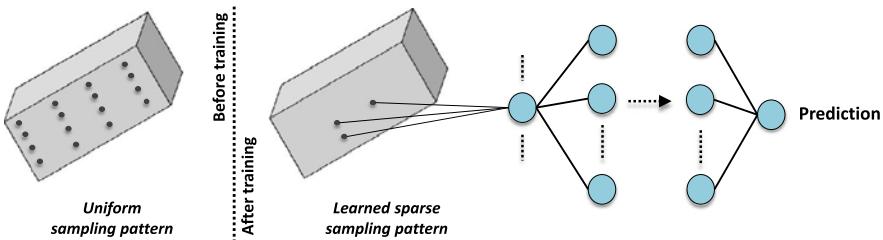
where  $m$  represents the number of training samples. This is equivalent to minimizing a cost function  $\mathcal{C}(\cdot)$  measuring the discrepancy between the network prediction and the expected output, i.e. the true label. We choose the  $L_2$ -norm as opposed to the cross-entropy classification loss given that the smoothness of this norm around the ground-truth ensures a more robust aggregation of the most probable hypotheses to a final detection result. As such, we obtain the following minimization problem:

$$\left(\hat{\vec{w}}, \hat{\vec{b}}\right) = \arg \min_{\vec{w}, \vec{b}} \left[ \mathcal{C}(\vec{X}; \vec{w}, \vec{b}) = \|\mathcal{R}(\vec{X}; \vec{w}, \vec{b}) - \vec{y}\|_2^2 \right]. \quad (3.3)$$

This can be solved with a standard Stochastic Gradient Descent (SGD) approach [13].

### 3.3.2 SPARSE ADAPTIVE DEEP NEURAL NETWORKS

The straightforward application of typical neural networks is however not feasible in the volumetric setting. To enable this we propose sparse adaptive deep neural networks, selecting *sparsity* as a means to simplify the neural network, aiming for computational efficiency and to avoid overfitting (see Fig. 3.2). We achieve this by

**FIGURE 3.2**

Visualization of the difference between uniform/handcrafted feature patterns and self-learned, sparse, adaptive patterns.

---

**Algorithm 3.1** Learning algorithm with iterative threshold-enforced sparsity

---

```

1: Pre-training:  $\vec{w}^{(0)} \leftarrow \vec{w}$  (small # epochs)
2: Set sparsity map  $\vec{s}^{(0)} \leftarrow \vec{1}$ 
3:  $t \leftarrow 1$ 
4: for training round  $t \leq T$  do
5:   for all filters  $i$  with sparsity do
6:      $\vec{s}_i^{(t)} \leftarrow \vec{s}_i^{(t-1)}$ 
7:     Update  $\vec{s}_i^{(t)}$  – remove smallest active weights
8:      $\vec{w}_i^{(t)} = \vec{w}_i^{(t-1)} \odot \vec{s}_i^{(t)}$ 
9:     Normalize active coefficients s.t.  $\|\vec{w}_i^{(t)}\|_1 = \|\vec{w}_i^{(t-1)}\|_1$ 
10:    end for
11:     $\vec{b}^{(t)} \leftarrow \vec{b}^{(t-1)}$ 
12:    Train network on active weights (small # epochs)
13:     $t \leftarrow t + 1$ 
14:  end for
15: Sparse kernels:  $\vec{w}_s \leftarrow \vec{w}^{(T)}$ 
16: Bias values:  $\vec{b}_s \leftarrow \vec{b}^{(T)}$ 

```

---

permanently eliminating filter weights, while approximating as well as possible the original filter response. In general terms, we aim to find a sparsity map  $\vec{s}$  for the network weights  $\vec{w}$ , such that over  $T$  training rounds, the response residual  $\epsilon$  given by:

$$\epsilon = \|\mathcal{R}(X; \vec{w}_s, \vec{b}_s) - \vec{y}\|_2^2 \quad (3.4)$$

is minimal, where  $\vec{b}_s$  denotes the biases of neurons in the sparse network and  $\vec{w}_s$  denotes the learned sparse weights, determined by the sparsity map  $\vec{s}$  with  $s_i \in \{0, 1\}, \forall i$ . The learning is based on a greedy iterative process in which we gradually eliminate neural connections with minimal impact on the network response, while continuing the training on the remaining active connections (see [Algorithm 3.1](#)).

In each round  $t \leq T$  a subset of active connections with minimal absolute value is selected and permanently removed from the network. In order to restore the network response to a certain degree, we re-normalize each filter to preserve its  $L_1$ -norm (see [Algorithm 3.1](#)). The training is then continued on the remaining active connections, driving the recovery of the neurons from the missing information (see step 12 of [Algorithm 3.1](#)):

$$\left( \hat{\vec{w}}^{(t)}, \hat{\vec{b}}^{(t)} \right) = \arg \min_{\substack{\vec{w}: \vec{w}^{(t)} \\ \vec{b}: \vec{b}^{(t)}}} \mathcal{C}(\vec{X}; \vec{w}, \vec{b}), \quad (3.5)$$

where  $\vec{w}^{(t)}$  and  $\vec{b}^{(t)}$  (computed from the values in round  $t - 1$ ) are used as initial values in the optimization step.

Our system learns on the lowest level adaptive, sparse data patterns which capture essential structures in the data, explicitly discarding input with minimal impact on the network response function  $\mathcal{R}$  (see [Fig. 3.2](#)). The computed patterns can reach sparsity levels of **90–95%**, meaning that only 5–10% of the weights of the low-level, fully-connected kernels can approximate the original network response  $\mathcal{R}$ . This not only increases the evaluation speed by around 2 orders of magnitude, but also acts as regularization for the model and reduces the likelihood of overfitting during training. We call this type of networks Sparse Adaptive Deep Neural Networks (SADNN).

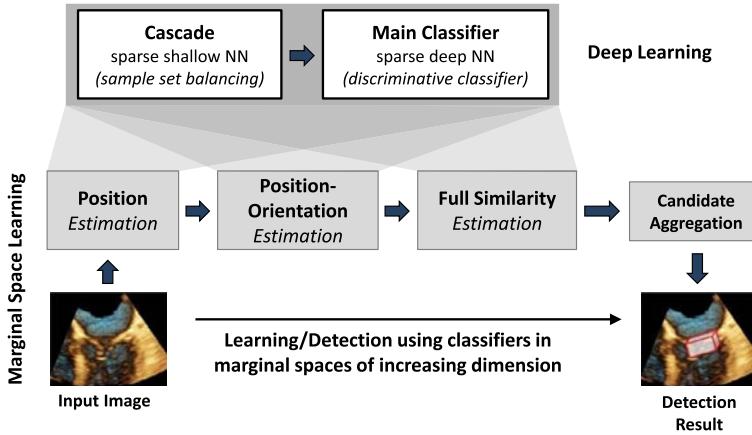
### 3.3.3 MARGINAL SPACE DEEP LEARNING

In the context of volumetric image parsing we propose to use SADNN to estimate the transformation parameters. Assuming a restricted affine transformation, let  $\vec{T} = (t_x, t_y, t_z)$ ,  $\vec{R} = (\phi_x, \phi_y, \phi_z)$  and  $\vec{S} = (s_x, s_y, s_z)$  define the translation, orientation, and anisotropic scaling of an anatomical object. Given an observed input image  $I$ , the estimation of the transformation parameters is equivalent to maximizing the posterior probability:

$$\left( \hat{\vec{T}}, \hat{\vec{R}}, \hat{\vec{S}} \right) = \arg \max_{\vec{T}, \vec{R}, \vec{S}} p(\vec{T}, \vec{R}, \vec{S}|I). \quad (3.6)$$

Using a classifier to directly approximate this distribution is not feasible given the large dimensionality of the parameter space. The mechanism of Marginal Space Learning (MSL) [\[5\]](#) proposes to split this space in a hierarchy of clustered, high-probability regions of increasing dimensionality starting in the position space, extending to the position–orientation space, and finally to the full 9D space, including also the anisotropic scaling information of the object. This can be easily achieved based on the following factorization:

$$\begin{aligned} \left( \hat{\vec{T}}, \hat{\vec{R}}, \hat{\vec{S}} \right) &= \arg \max_{\vec{T}, \vec{R}, \vec{S}} p(\vec{T}|I) p(\vec{R}|\vec{T}, I) p(\vec{S}|\vec{T}, \vec{R}, I) \\ &= \arg \max_{\vec{T}, \vec{R}, \vec{S}} p(\vec{T}|I) \frac{p(\vec{T}, \vec{R}|I)}{p(\vec{T}|I)} \frac{p(\vec{T}, \vec{R}, \vec{S}|I)}{p(\vec{T}, \vec{R}|I)}, \end{aligned} \quad (3.7)$$

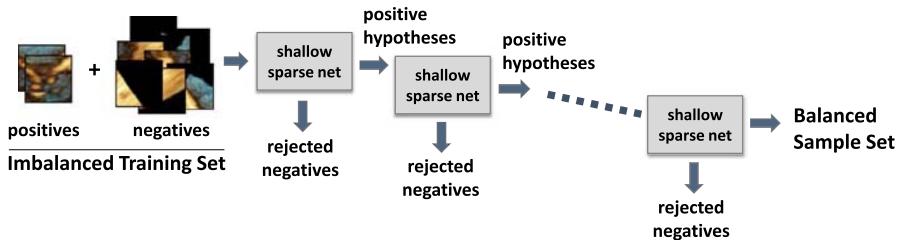
**FIGURE 3.3**

Schematic visualization of the marginal space deep learning framework.

where the probabilities  $p(\vec{T}|I)$ ,  $p(\vec{T}, \vec{R}|I)$ , and  $p(\vec{T}, \vec{R}, \vec{S}|I)$  are defined in the previously enumerated spaces, also called *marginal spaces* (see Fig. 3.3). In our method we directly approximate each of these distributions with an SADNN,  $\mathcal{R}(X; \vec{w}_s, \vec{b}_s)$ . This reduces the problem to learning classifiers in these marginal spaces and then scanning each space exhaustively to estimate the object transformation. The workflow starts with learning the translation parameters in the translation space  $\mathcal{U}_T(I)$ . After this only the positive hypotheses with highest probability, clustered in a dense region are augmented with discretized orientation information, to build the joint translation–orientation space  $\mathcal{U}_{TR}(I)$ . The same principle applies when extending to the full 9D space  $\mathcal{U}_{TRS}(I)$ . The end-to-end optimization is defined by:

$$\begin{aligned} (\hat{\vec{T}}, \mathcal{U}_{TR}(I)) &\leftarrow \arg \max_{\vec{T}} \mathcal{R}(\mathcal{U}_T(I); \vec{w}_s, \vec{b}_s) \\ (\hat{\vec{T}}, \hat{\vec{R}}, \mathcal{U}_{TRS}(I)) &\leftarrow \arg \max_{\vec{T}, \vec{R}} \mathcal{R}(\mathcal{U}_{TR}(I); \vec{w}_s, \vec{b}_s) \\ (\hat{\vec{T}}, \hat{\vec{R}}, \hat{\vec{S}}) &\leftarrow \arg \max_{\vec{T}, \vec{R}, \vec{S}} \mathcal{R}(\mathcal{U}_{TRS}(I); \vec{w}_s, \vec{b}_s), \end{aligned} \quad (3.8)$$

where  $\mathcal{R}(\cdot; \vec{w}_s, \vec{b}_s)$  denotes the response of each of the sparse adaptive deep neural networks, learned from the supervised training data  $(\vec{X}, \vec{y})$  in each marginal space. Using this type of hierarchical parametrization brings a speed-up of 6 orders of magnitude compared to the exhaustive search in the 9D space (see proof in [5]). The pipeline is visualized in Fig. 3.3.

**FIGURE 3.4**

Schematic visualization of the negative-sample filtering cascade.

---

**Algorithm 3.2** Negative sample filtering algorithm
 

---

- 1:  $P$  – set of positive samples
  - 2:  $N$  – set of negative samples ( $|P| \ll |N|$ )
  - 3: **while**  $|N| \geq 1.5 \times |P|$  **do**
  - 4:     Learn shallow SADNN using [Algorithm 3.1](#)
  - 5:      $d \leftarrow$  largest decision boundary with FNR = 0
  - 6:      $N' \leftarrow$  filter  $N$  based on  $d$  – eliminate true negatives
  - 7:      $N \leftarrow N'$
  - 8: **end while**
- 

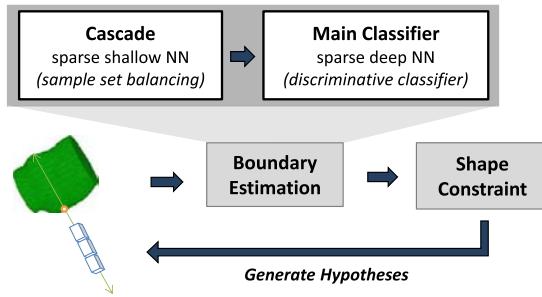
### **3.3.3.1 Cascaded Negative Filtering**

One important particularity of the learning task in each marginal space is the high class-imbalance which can reach ratios of 1 : 1000 positive to negative samples. This impacts not only the training efficiency but also the stochastic sampling of the gradient. The main observation is that typically a large percentage of the negative hypotheses are very easy to classify. As such, we propose to use a cascade of shallow neural networks to efficiently and effectively filter the negative hypotheses prior to using a deep powerful model for classification. In each stage of the cascade we train a shallow, sparse neural network and adaptively tune its decision boundary to eliminate as many true negative hypotheses from the training set as possible. The remaining hypotheses, classified as positives, are propagated to the next stage of the cascade where the same filtering step is applied until the training set is balanced (see [Fig. 3.4](#) and [Algorithm 3.2](#)).

Using this mechanism the size of the sample set processed by the main classifier in each stage is reduced from  $|N| + |P|$  to approximately  $3 \times |P|$ , improving the scanning performance by an additional 2 orders of magnitude (depending on the imbalance ratio).

### **3.3.3.2 Nonrigid Deformation Estimation**

Based on the underlying image information and the initial mean shape computed based on the ground-truth data, we propose an active shape model to guide the shape

**FIGURE 3.5**

Schematic visualization of the boundary deformation with SADNN. Starting from the current shape, the SADNN is aligned and applied along the normal for each point of the mesh, the boundary is deformed and projected under the current shape space. The process is iteratively repeated.

deformation. Since the original approach based on energy deformation [7] is not feasible in a 3D setup where the image information and the boundary context are very complex, machine learning has been used as an alternative mechanism to guide the object boundary in both 2D [25,26] and 3D [5]. Typically, a classifier labeled as boundary detector is trained at specific anatomical locations to detect the shape boundary based on local evidence. Following this approach we propose to use the SADNN (with negative filtering cascade) as a boundary classifier to automatically learn adaptive, sparse feature sampling patterns directly from low-level image data. Essentially, for a given control point of the warping mesh, we are dealing with the same problem as faced in the joint translation–orientation space, in the localization stage. Training is performed by using positive samples on the current ground-truth boundary for each mesh point (aligned with the corresponding normal) and using negative samples at various distances from the boundary. Our experiments show that the sparse adaptive patterns are essential in efficiently and effectively capturing the relevant anatomical structures around the boundary. The iterative process is illustrated in Fig. 3.5. The boundary estimation is followed by constraining the deformed shape to the space of shapes corresponding to the current object. We use statistical shape modeling for the constraint, where we estimate from the training set the linear shapes subspace through principal components analysis and online we project the current shape into this subspace using the learned linear projector. The process of boundary estimation and shape constraint enforcement are iteratively applied for a number of pre-determined iterations or until there are no large deformations.

### 3.3.4 AN ARTIFICIAL AGENT FOR IMAGE PARSING

While the proposed framework systematically addresses many limitations of the state-of-the-art, the algorithmic steps remain unconstrained, exhaustive, and thus sub-

optimal in terms of runtime. For example, the scanning is always performed on the entire parameter space at testing time, even if the positive hypotheses are typically clustered in dense regions. To address this limitation, a principled more intelligent mechanism is required. In general, there is no precise definition or quantification of intelligence or intelligent behavior [27–29]. We can argue that intelligence represents the ability of an algorithm or artificial entity to learn and understand tasks, as opposed to mechanically following predefined solution steps [30]. Relating this to the introduced context of image parsing using machine learning the most important limitation of our approach and other state-of-the-art solutions is the fact that the modeling of object appearance and the search for the optimal transformation parameters are completely decoupled steps.

In the following, we make a first step toward self-taught virtual agents for image understanding in the context of medical image parsing, by formulating the landmark detection problem as a generic learning task for an artificial agent. Our solution leverages state-of-the-art representation learning techniques through deep learning [1] and powerful solutions for generic behavior learning through reinforcement learning [12] to create a model encapsulating a cognitive-like learning process to discover strategies for localizing arbitrary landmarks, using only the raw input image information and the landmark annotations. In other words, we do not use the machine to execute predefined solution steps, but give the machine the ability to find itself the solution.

### **3.3.4.1 Cognitive Modeling Using Deep Reinforcement Learning**

Building powerful artificial agents that can emulate or even surpass human performance at given tasks requires the use of an automatic, generic learning model observed not only in exploratory, unsupervised human cognition [30], but also in basic reward-based animal learning methods [31]. The artificial agent needs to be equipped with at least two fundamental capabilities found at the core of the human and animal intelligence. At perceptual level is the automatic capturing and disentangling of high-dimensional signal data which describes the complete situation in which the agent can find itself, while on cognitive level is the ability to reach decisions and act upon this entire observed information flow [30]. While deep learning can be used to process and interpret the perceived stream of data, reinforcement learning lends itself as a powerful tool for cognitive modeling.

Reinforcement learning (RL) is a technique aimed at effectively describing learning as an end-to-end cognitive process, instead of as a predefined methodology [32]. A typical RL setting is composed by an artificial agent that can interact with an uncertain environment with the target of reaching pre-determined goals. The agent can observe the state of the environment and choose to act on it, similar to a trial-and-error search [32], maximizing the future reward signal received as a response from the environment (see Fig. 3.6 for a schematic overview). This reward-based decision process is modeled in RL theory as a *Markov Decision Process* (MDP) [32] defined by a tuple  $\mathcal{M} := (\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma)$ , where:

- $\mathcal{S}$  represents a finite set of states,  $s_t \in \mathcal{S}$  being the state of the agent at time  $t$

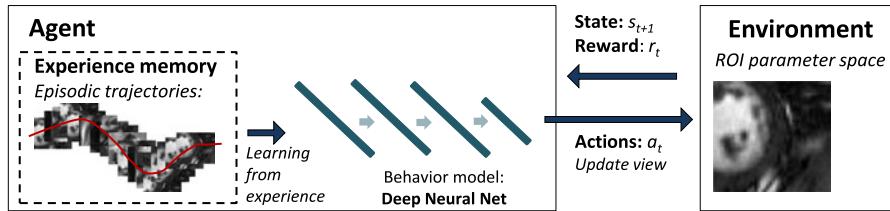


FIGURE 3.6

System diagram showing the interaction between the artificial agent and the environment during training, in the context of landmark detection. The state  $s_t$  of the environment at time  $t$  is defined by the current view of the agent, given as a fixed patch. Depending on the current state, the actions of the agent directly impact the environment, resulting in a new state and a quantitative reward,  $(s_{t+1}, r_t)$ . The experience memory stores the last states visited by the agent, which are randomly sampled to periodically update the parameters of the neural network.

- $\mathcal{A}$  represents a finite set of actions allowing the agent to interact with the environment,  $a_t \in \mathcal{A}$  being the action the agent performs at time  $t$
- $\mathcal{T} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0; 1]$  is a stochastic transition function, where  $\mathcal{T}_{s,a}^{s'}$  describes the probability of arriving in state  $s'$  after the agent performed action  $a$  in state  $s$
- $\mathcal{R} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow \mathbb{R}$  is a scalar reward function, where  $\mathcal{R}_{s,a}^{s'}$  denotes the expected reward after a state transition
- $\gamma$  is the discount factor controlling the importance of future versus immediate rewards.

Formally, the future discounted reward of an agent at time  $\hat{t}$  can be written as  $R_{\hat{t}} = \sum_{t=\hat{t}}^T \gamma^{t-\hat{t}} r_t$ , with  $T$  marking the end of a learning episode and  $r_t$  defining the immediate reward the agent receives at time  $t$ . Especially in model-free reinforcement learning, the target is to find the optimal so-called action-value function  $Q^*(s, a)$ , denoting the maximum expected future discounted reward when starting in state  $s$  and performing action  $a$ :

$$Q^*(s, a) = \max_{\pi} \mathbb{E} [R_t | s_t = s, a_t = a, \pi], \quad (3.9)$$

where  $\pi$  is an action policy, or in other words, a probability distribution over actions in each given state. Once the optimal action-value function is estimated, the optimal action policy, determining the behavior of the agent, can be directly computed in each state as:

$$\forall s \in \mathcal{S} : \pi^*(s) = \arg \max_{a \in \mathcal{A}} Q^*(s, a). \quad (3.10)$$

One important relation satisfied by the optimal action-value function  $Q^*$  is the Bellman optimality equation [33] which represents a recursive formulation of Eq. (3.9).

This is defined as:

$$\begin{aligned} Q^*(s, a) &= \sum_{s'} \mathcal{T}_{s,a}^{s'} \left( \mathcal{R}_{s,a}^{s'} + \gamma \max_{a'} Q^*(s', a') \right) \\ &= \mathbb{E}_{s'} \left( r + \gamma \max_{a'} Q^*(s', a') \right), \end{aligned} \quad (3.11)$$

where  $s'$  defines a possible state visited after  $s$ ,  $a'$  the corresponding action and  $r = R_{s,a}^{s'}$  represents a compact notation for the current, immediate reward. Viewed as an operator  $\tau$ , the Bellman equation defines a contraction mapping. Theoretical results [33] show that by iteratively applying  $Q_{i+1} = \tau(Q_i), \forall(s, a)$ , the function  $Q_i$  converges to  $Q^*$  at infinity. This standard policy iteration approach is however not feasible in practice. An alternative is to use model-free temporal difference methods, typically Q-Learning [15], which exploit correlations of consecutive states. A step further toward a higher computational efficiency is the use of parametric functions to approximate the  $Q$ -function [34]. Considering the expected nonlinear structure of the action-value function [15], neural networks represent a potentially powerful approximation solution [35,16,36,37]. The first end-to-end reinforcement learning approach making use of state-of-the-art deep neural networks for direct policy approximation is presented by Mnih et al. [12].

In the following we will show how these reinforcement learning concepts can be applied in the context of intelligent image parsing, grounded on recent advances in deep learning. Fig. 3.6 gives an overview of our approach.

### 3.3.4.2 Learning to Scan for Anatomical Objects

Let us focus on the problem of landmark detection. The principles can easily be extended to different parametric spaces as modeled in marginal space learning. Given a set of  $N$  training images  $I_1, I_2, \dots, I_N$ , each containing  $M$  annotated landmarks. Focusing on one particular landmark indexed in each training example, we would like to train an intelligent agent that can automatically discover strategies for finding the selected landmark as opposed to following tedious scanning routines.

#### 3.3.4.2.1 Problem Formulation

We reformulate our task as a *Markov Decision Process*  $\mathcal{M} := (\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma)$ . In the following we will specify the state and action spaces and define the reward system (note that the transition probabilities  $\mathcal{T}$  are unknown in our model-free approach).

**States.** The agent needs to be able to capture its surrounding environment and be able to react upon what is observed. In consequence, states need to be discriminative and self-describing. Following this requirement, the selection of the state space is inspired by principles found in the animal and human visual perception system [38]. When analyzing an image or a static scene, with the focus set on one particular point, the surrounding context which is actively perceived is very limited. In other words, what is captured is dense local information around the focus point and limited

global context from the surrounding neighborhood [38]. Starting from this observation, we make a similar locality assumption and define a state observed at time  $t$  as  $s_t = (I_t, x_t, y_t, l_t)$ , i.e. a local patch of size  $l_t \times l_t$  centered at position  $(x_t, y_t)$  in the observed image  $I_t$ . States which are close to the target landmark location will directly capture the position of the landmark in their context. For distant states, the relation to the landmark location is intrinsic, captured indirectly by information from the context of the current patch. This is consistent with the general properties of the perception model introduced in [38].

**Actions.** In each state the agent interacts with the enclosing environment by performing certain actions from a predefined action set [15]. The set of actions is chosen in such a way that the agent is given the possibility to explore the entire environment. Located in state  $s_t$  at time  $t$ , with the aim of reaching the location of a target landmark, we give the agent the possibility to perform one of the following discrete actions: move *upward*, *downward*, *left* or *right*. For each action, we simplify to a single-pixel move:  $x_{t+1} \leftarrow x_t \pm 1$  and  $y_{t+1} \leftarrow y_t \pm 1$ . Once the target has been reached, no further action is performed and the search is finished. At testing time this results in an oscillatory transition between neighboring states at the target, eliminating the need for an additional *stop* action.

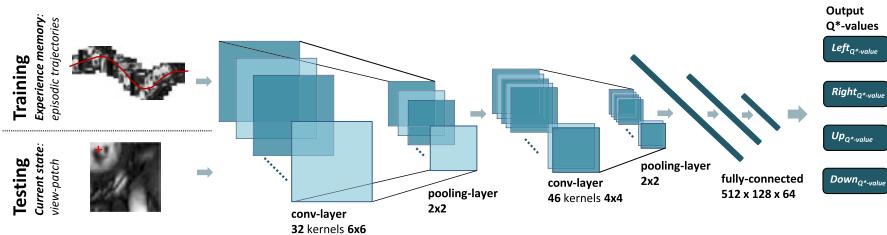
**Rewards.** The reward system is based on the change in relative position at state  $s_t : (x_t, y_t)$  with respect to the target position of the landmark  $s_{target} : (x_{target}, y_{target})$ . Intuitively, for a move in the correct direction, a positive reward proportional to the target-distance reduction is given, whereas a move in the wrong direction is punished by a negative reward of equal magnitude. Formally, the reward at time  $t$  is given by:

$$r_t = dist(s_t, s_{target}) - dist(s_{t+1}, s_{target}). \quad (3.12)$$

The only exception to this rule is an attempt to leave the image field by crossing the image border. Such an action is always given the highest punishment of  $-1$ . As it can be observed, the reward is always correlated with the goodness of a performed action. The motivation for this more complex reward system compared to the simple  $\pm 1$  reward used by Mnih et al. [12] is inspired by principles applied in complex trial-error systems [30], simulating more closely human experience. In other words, good actions, contributing significantly toward reaching the goal, are given a high reward, whereas actions that only marginally improve the state of the agent receive little reward.

### 3.3.4.2.2 Proposed Method

Given the environment definition and reward system, the goal of the agent is to select actions by repeatedly interacting with the enclosing environment in order to maximize cumulative future reward (see Eq. (3.9)). This optimal behavior is defined by the optimal policy  $\pi^*$  selected from the space of all possible policies  $\pi \leftarrow p(action|state)$ . As shown in Eq. (3.10), finding the optimal policy is straightforward given the optimal action-value function  $Q^*$ .

**FIGURE 3.7**

Schematic illustration of the 7-layer deep convolutional neural network used to approximate the optimal action-value function  $Q^*(s_t, a_t)$ . The network output is the  $Q^*$  value, the maximum expected future reward for all possible actions in the current state (moving *left*, *right*, *up*, or *down*). During the training stage we apply experience replay by randomly sampling patches from all episodic trajectories stored in memory to define the mini-batch used to update the network parameters. During the testing stage the input to the network is one single patch, the current state, or in other words, the current view of the agent on the image. Evaluating the network on this patch helps the agent decide in which direction to navigate in order to reach the target location (visualized as a red +).

In this work we propose a model-free, temporal difference approach introduced in the context of game learning by Mnih et al. [12], using a deep convolutional neural network (CNN) to approximate the optimal action-value function  $Q^*$ . Defining the parameters of a deep CNN as  $\theta = [\vec{w}, \vec{b}]$ , we use this architecture as a generic, non-linear function approximator  $Q(s, a; \theta) \approx Q^*(s, a)$ , called deep Q network (DQN), which we visualize in Fig. 3.7. To account for the possible divergence issues during training we apply the concepts of *reference update-delay* and *experience replay*.

Similar to the temporal difference Q-Learning algorithm [15], a deep Q network can be trained in a reinforcement learning setup using an iterative approach to minimize the mean squared error based on the Bellman optimality criterion (see Eq. (3.11)). At any iteration  $i$ , we can approximate the optimal expected target values using a set of reference parameters  $\theta_i^{ref} := \theta_j$  from a previous iteration  $j < i$  as:

$$y = r + \gamma \max_{a'} Q(s', a'; \theta_i^{ref}). \quad (3.13)$$

As such we obtain a sequence of well-defined optimization problems driving the evolution of the network parameters. The error function at each step  $i$  is defined as:

$$\theta_i = \min_{\theta_i} \mathbb{E}_{s,a,r,s'} \left[ (y - Q(s, a; \theta_i))^2 \right] + \mathbb{E}_{s,a,r} \left[ \mathbb{V}_{s'}[y] \right]. \quad (3.14)$$

This is a standard, supervised setup for deep learning for which mini-batch gradient-based approaches [13] combined with backpropagation [39] represent a powerful solution. In our framework we periodically apply stochastic gradient descent steps,

approximating the gradient by randomly sampling the gradient function, given as:

$$\nabla_{\theta_i} = \mathbb{E}_{s,a,r,s'} [(y - Q(s, a; \theta_i)) \nabla_{\theta_i} Q(s, a; \theta_i)]. \quad (3.15)$$

**Reference Update-Delay.** As motivated in [12], using a different network to compute the reference values for training brings robustness to the algorithm. In such a setup, changes to the current parameters  $\theta_i$  and implicitly to the current approximator  $Q(\cdot; \theta_i)$  cannot directly impact the reference output  $y$ , introducing an update-delay and thereby reducing the probability to diverge and oscillate in suboptimal regions of the optimization space [12]. Fig. 3.7 depicts the used architecture providing a different, more in depth visualization of the main system diagram introduced in Fig. 3.6. In a given state, specified by the current view-patch, we evaluate the neural network on that particular patch to simultaneously obtain  $Q^*$ -value estimates for all possible actions:  $Q^*(s, a_1), Q^*(s, a_2), \dots$ . Given the estimates, the agent applies an  $\epsilon$ -greedy policy, choosing a random action with probability  $\epsilon$  and following the current policy estimation (choosing the action with the maximum future, discounted reward) with probability  $1 - \epsilon$ . During learning a value decay is applied on the parameter  $\epsilon$  reaching a trade-off between an effective space exploration and a greedy, consistent policy exploitation strategy.

**Experience Replay.** To ensure the robustness of the parameter updates and train more efficiently, we propose to use the concept of experience replay [40]. In experience replay, the agent stores a limited amount of previously visited states (typically the last states) in a so-called experience memory and then samples that memory to update the parameters of the underlying neural network [16,12]. In our case, we learn in a sequence of episodes which quantify the local performance of the agent on given training images. Before the start of one episode, the agent is given a random image from the complete training set and a random start-state, i.e. start position in that image. During the course of the episode, the agent performs actions applying the  $\epsilon$ -greedy behavior policy and navigating through this local environment (the given training image). The episode finishes when the target state is reached, or in other words, the landmark location is found, or a pre-defined maximum number of actions are executed. This defines a so-called *trajectory*  $t_i$  (we also call it episodic trajectory) in the image space which encodes the applied search strategy as a sequence of visited states. All these trajectories are stored in our replay memory since they represent the entire experience the agent has accumulated on different images. In our experiments we store the last  $P$  trajectories as  $\mathcal{E} = [t_1, t_2, \dots, t_P]$ . At fixed intervals during training (typically every 4–6 state transitions) a parameter update is performed using a random mini-batch of states extracted from  $\mathcal{E}$ . Similarly to [40,16,12], we argue that this kind of approach is essential for ensuring the training convergence. Updating the deep neural network on locally correlated states (similar to Q-Learning) does not generalize; on the contrary, this negatively affects the performance of the network in other parts of the state-space. Using a uniformly sampled set of previous experiences, averages the distribution of the network input, reducing oscillations, and ensuring a

much faster and robust learning experience. Figs. 3.6 and 3.7 visualize this type of experience-based learning approach.

---

## 3.4 EXPERIMENTS

In this section we present the experiments we performed for validation. These include object detection and segmentation in 3D and landmark detection in 2D and 3D.

### 3.4.1 ANATOMY DETECTION AND SEGMENTATION IN 3D

In order to evaluate the performance of the MSDL framework, we compare against the state-of-the-art MSL solution [5] on detecting and segmenting the aortic valve (root) in 3D transesophageal echocardiogram (TEE) images. The aortic root connects the ascending aorta to the left ventricular outflow tract and is represented through a tubular grid (see Fig. 3.9). This is a particularly challenging task, considering the high variation in anatomical appearance of the valve, the heart-motion, and implicitly motion of the valve-leaflets, as well as the image quality limitations, i.e. noise in ultrasound images.

#### 3.4.1.1 *Dataset*

The dataset used for evaluation stems from 869 patients and contains 2891 3D TEE volumes. The size of the images vary from  $100 \times 100 \times 50$  to  $250 \times 250 \times 150$  voxels while the spatial resolution lies between 0.75 to 1 mm. The preprocessing step includes the intensity normalization of the image as well as the resampling of the volumes to an isotropic 3 mm resolution.

The validation setup is the same for both approaches and is based on a random split of the volumes at patient level in 2481 training and 410 testing volumes (about 84% and 16%). At patient level this results in 719 patients in the training set and 150 patients in the test set. The ground-truth data was obtained through manual annotation performed by experts [41]. Note that the pose of the 3D bounding box is fully determined by the underlying anatomy.

#### 3.4.1.2 *Model Selection and Training Parameters*

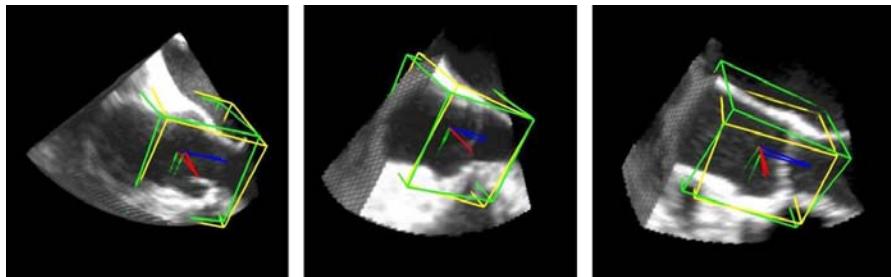
All meta-parameters related to the sub-space sampling and learning model are optimized using a systematic grid-search. Across all 3 marginal spaces we use the same SADNN architecture for the cascade and main classifier, i.e. cascade with 2 layers = 5832 (sparse)  $\times$  60  $\times$  1, and main classifier with 4 layers = 5832 (sparse)  $\times$  150  $\times$  80  $\times$  50  $\times$  1 hidden units. In all networks we used sigmoid activations.

#### 3.4.1.3 *Evaluation*

The measures used to quantify the performance of the object localization are the difference in center position and the corner distance error, and the average distance between the 8 corners of the detected box and the ground-truth box. The second value

**Table 3.1** Comparison of the performance of the state-of-the-art MSL [5] and the proposed MSDL framework for aortic valve detection. The measures used to quantify the quality of the results w.r.t. to the ground-truth data are the error of the position of the box and mean corner distance (both measured in millimeters). The superior results are displayed in bold

	Position error [mm]				Corner error [mm]			
	Training data		Test data		Training data		Test data	
	MSL	MSDL	MSL	MSDL	MSL	MSDL	MSL	MSDL
Mean	3.12	<b>1.47</b>	3.34	<b>1.83</b>	5.42	<b>2.80</b>	6.16	<b>3.72</b>
Median	2.80	<b>1.27</b>	3.05	<b>1.58</b>	4.98	<b>2.58</b>	5.85	<b>3.34</b>
STD	1.91	<b>0.99</b>	1.85	<b>1.31</b>	2.47	<b>1.23</b>	2.31	<b>1.74</b>



**FIGURE 3.8**

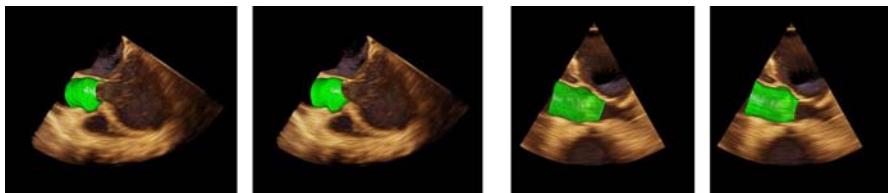
Example of detected bounding boxes for images from different patients from the test set. The detected box is shown in green and the ground-truth in yellow.

gives also an indication about the accuracy of the orientation and scale estimation. **Table 3.1** shows the obtained results. The MSDL framework significantly improves the performance of the state-of-the-art MSL solution, improving the mean position error by 45.2% and the corner distance error by 39.6%. **Fig. 3.8** shows qualitative results for different patients from the test set. The MSDL framework matches the excellent run-time performance of the MSL framework running in less than 0.5 seconds using only the CPU. This is greatly due to the use of sparse data sampling patterns which bring a speed-up of 300× to the MSDL-non-sparse, hence also the significant computational benefit of the network simplification. We also emphasize here that using a cascade for the early rejection of hypotheses is essential to reaching this competitive performance. Depending on the imbalance ratio in each marginal space, using a cascade brings around 2 orders of magnitude speed-wise improvement.

For the segmentation experiments the performance is measured by computing the distance between the segmentation and the ground-truth annotation mesh. **Table 3.2** shows that the MSDL approach outperforms the MSL method by reducing the average mesh error from 1.04 to 0.9 mm. This represents an improvement of 13.5% at a runtime of less than 1 seconds/volume. Qualitative results can be seen in **Fig. 3.9**.

**Table 3.2** Comparison of the performance of the state-of-the-art MSL [5] and the proposed MSDL framework for aortic valve segmentation. The measure illustrates the distance of the detected mesh to the ground-truth for both the initialization from the detected box as well as the distance after boundary refinement. The superior results are displayed in bold

	Training data, dist. to gt. mesh [mm]				Test data, dist. to gt. mesh [mm]			
	Initialization		Final		Initialization		Final	
	MSL	MSDL	MSL	MSDL	MSL	MSDL	MSL	MSDL
Mean	2.08	<b>1.16</b>	1.17	<b>0.89</b>	2.06	<b>1.21</b>	1.04	<b>0.90</b>
Median	1.94	<b>1.09</b>	1.05	<b>0.82</b>	1.95	<b>1.10</b>	0.98	<b>0.80</b>
STD	0.83	<b>0.40</b>	0.66	<b>0.35</b>	0.79	<b>0.55</b>	0.50	<b>0.48</b>



**FIGURE 3.9**

Example images showing the aortic valve segmentation results for different patients from the test set. The images are organized as two pairs, with the detection on the left and ground-truth on the right.

#### 3.4.1.4 Additional Experiments and Discussion

Given the fact that the accuracy of the nonrigid shape segmentation directly depends on the accuracy of the object detection, Table 3.2 is not an optimal indicator for the superiority of the DL-based segmentation alone. In this context we have investigated the performance of two mixed framework-versions: one combining our MSDL object detection with the original segmentation approach from Zheng et al. [5] and the other combining the MSL object detection solution [5] with our the DL-based segmentation. This gives a direct comparison between typical handcrafted features and adaptive sparse patterns in capturing the image context around the boundary. The experiment highlights the overall superiority of our end-to-end DL-based approach on the test set. Table 3.3 shows the results.

We also highlight the performance of our cascaded sparse architecture compared to a single standard end-to-end deep network in terms of speed. We have experimented with using a CNN both in the cascade and as main deep classifier. Our implementation is based on direct calls to the latest cuDNN 4.0 library on top of a high-end GTX980 GPU architecture. We found that reasonable architectures with 1 convolution layer for the cascade (respectively, 3 convolution layers (with pooling and fully-connected) for the main classifier) are about 400–500 times slower than our original solution. For example, in the orientation stage a deep 3D-CNN could pro-

**Table 3.3** Comparison of the performance of our DL-based framework with the state-of-the-art solution [5] and the two mixed variants: combining the MSDL box detection with the original segmentation method [5], and the MSL box detection [5] with our DL-based segmentation method. The superior results are displayed in bold

	Segmentation error [mm]		
	Mean	Median	STD
Ours	<b>0.90</b>	<b>0.80</b>	<b>0.48</b>
Reference [5]	1.04	0.98	0.50
Detection (ours) + Seg. [5]	0.95	0.88	0.48
Detection [5] + Seg. (ours)	1.00	0.90	0.51

cess around 600 hypotheses/second compared to 420,000 processed with an SADNN. When comparing shallow versions of these type of networks for cascade filtering, the SADNN is around  $200 \times$  faster, processing over 1,200,000 hypotheses/second using only the CPU. In this context, removing the cascade and using a single end-to-end deep CNN can only further increase this performance deficit, both during training and testing. As such, training our framework with standard deep networks and investigating their accuracy becomes infeasible, requiring in the order of thousands of hours of training time.

The main reason for this speed difference is not only the cascade filtering but also the data-efficiency of the sparse sampling. Recall that most sparse patterns in our SADNN architecture reach sparsity levels of 90–95%, thus indexing only a small fraction of the data voxels during scanning and minimizing the memory-footprint in the large incoming stream of volumetric data. This is different from the CNN architecture which samples every single voxel multiple times, proportionally to the size of the convolution kernel. However, integrating the CNN in our pipeline in a type of hybrid learning system is part of our ongoing work. For example, we are optimistic in managing to integrate such precise deep architectures into our pipeline to process only small subsets of difficult hypotheses, to further increase the accuracy of the system.

### 3.4.2 LANDMARK DETECTION IN 2D AND 3D

Accurate landmark detection is a fundamental prerequisite for medical image analysis. In the following we demonstrate the performance of our agent-driven intelligent parsing method on this type of application.

#### 3.4.2.1 Datasets

We perform the evaluation on three datasets. They are composed of 891 short-axis view MR images from 338 patients, 1186 cardiac ultrasound apical four-chamber view images from 361 patients, and 455 head-neck CT scans from 455 patients. The landmarks selected for testing are presented in Fig. 3.10. Each dataset is split

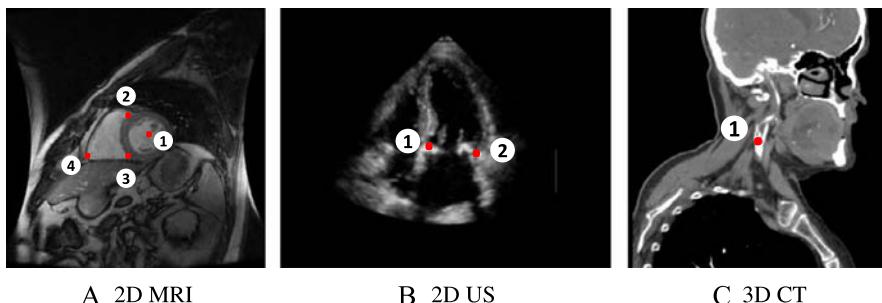
**FIGURE 3.10**

Figure A shows the LV-center (1), the anterior/posterior RV-insertion points (2)/(3) and the RV-extreme point (4) in a short-axis cardiac MR image. Figure B highlights the mitral septal annulus (1) and the mitral lateral annulus points (2) in a cardiac ultrasound image and figure C the right carotid artery bifurcation (1) in a head-neck CT scan.

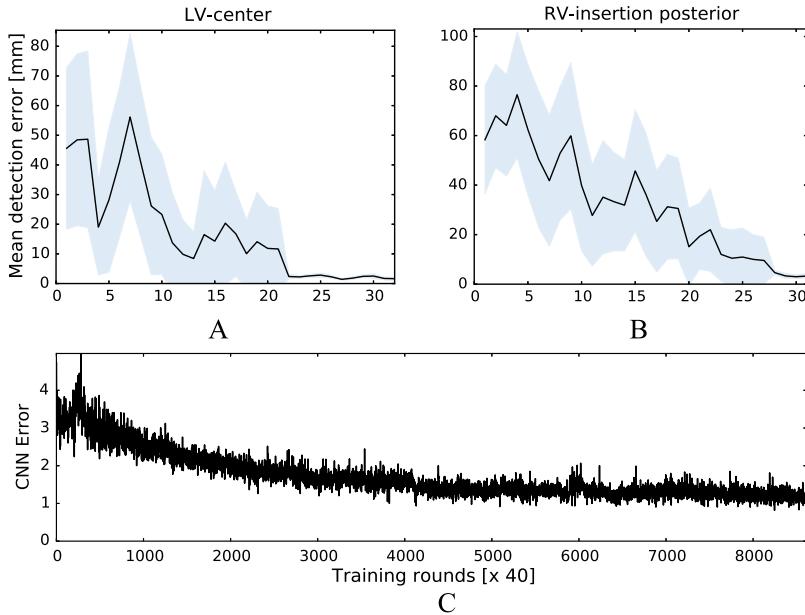
**Table 3.4** The detection error on the test sets with superior results highlighted in bold. The error is quantified as the distance to the ground-truth, measured in mm. With \* we signify that the results are reported on the same dataset, but on a different training/test data-split than ours

	Detection error [mm]									
	2D-MRI			2D-US				3D-CT		
	LV-center		RV-ext	RV-post		RV-ant	M-sep	M-lat	Bifurc.	
	Our	[43]	Our	[42]	Our	[43]	[42]	Our	Our	[6]
Mean	<b>1.8</b>	6.2*	<b>4.9</b>	8.4*	<b>2.2</b>	7.9*	5.9*	<b>3.7</b>	<b>1.3</b>	1.6
Median	<b>1.7</b>	5.4*	<b>4.2</b>	5.9*	<b>1.8</b>	4.7*	3.9*	<b>3.0</b>	<b>1.2</b>	1.3
STD	<b>2.2</b>	4.0*	<b>3.6</b>	16.5*	<b>1.5</b>	11.5*	16.0*	<b>2.3</b>	<b>0.8</b>	1.4
										2.6
										0.8
										1.2
										5.0

randomly at patient level in approximately 80% training, 10% validation, and 10% testing. The results on the MR dataset are compared to the state-of-the-art results achieved in [42,43] with methods combining context modeling with machine-learning for robust landmark detection. On the CT dataset we compare to [6], a state-of-the-art deep learning solution combined with exhaustive hypotheses scanning. Table 3.4 shows the obtained results.

#### 3.4.2.2 Evaluation: Learning to Parse

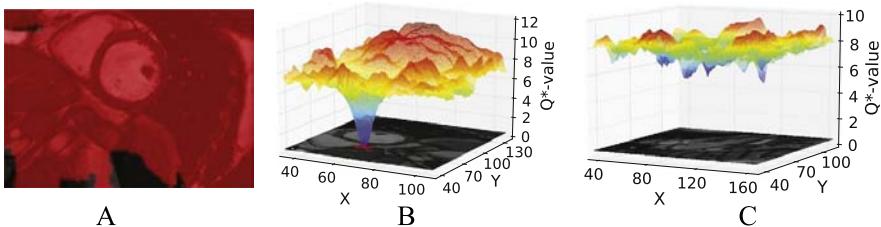
The training starts from a random initialization of the policy. In a sequence of learning episodes, the agent is given random training images with corresponding random start-states. The agent then follows the  $\epsilon$ -greedy search strategy in the selected image, generating at the end of the episode a trajectory which is added to its experience memory. During the exploration, periodic updates are applied to the parameters of the

**FIGURE 3.11**

Plots A and B show the progression of the detection accuracy on the validation set during training. Plot C represents the error of the Bellman optimality equation.

neural network, leading to a more accurate approximation of the optimal  $Q^*$  function, given the current experience. This process is repeated in an iterative manner until the detection accuracy on the validation set is minimal. Note that for all experiments the network architecture and training parameters are the same. In terms of training, we propose to use the *rms-prop* [12] mini-batch approach, yielding a better performance than standard stochastic gradient descent. The learning rate is set to  $\eta = 0.00025$ , justified by the sparse sampling applied in experience replay, while the discount factor is fixed to  $\gamma = 0.9$ . Other important training parameters are the replay memory size ( $P = 100,000$  view-patches) and  $\epsilon = 0.8$  decaying linearly to 0.05.

Fig. 3.11 shows the performance evolution during training. The high standard deviation of the detection accuracy is correlated with divergent trajectories, given the random initialization of the policy. However, as the policy improves, the detection accuracy increases significantly, reaching the minimum point when all trajectories converge to the correct landmark position. On the second line we show the average expected reward, computed for random states that are kept fixed across the training stages. The plot on the last line in Fig. 3.11 shows the progression of the mean squared error in the Bellman equation. In our experiments we quantify the quality of the learned policy and decide the number of training rounds based on the mean detection error on the cross-validation set.



**FIGURE 3.12**

Figure A highlights in transparent red all the starting positions converging to the correct landmark position. Figures B and C visualize the optimal action-value function  $Q^*$ , with each state (image position) encoding the highest expected reward, considering all actions allowed in that state. We also call it the optimal  $Q^*$ -field. In the first example, as can be observed, the global minimum point reaches near zero-value at the location of the target landmark (highlighted with a red **x** on the X-Y projected MR image). This is different from the other example where the  $Q^*$ -field is visualized for an image that does not contain the target landmark. This clearly indicates the absence of the object.

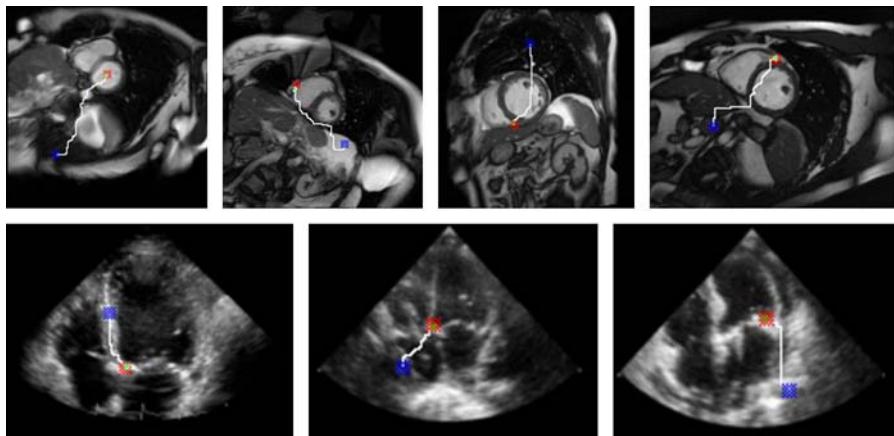
#### 3.4.2.2.1 Agent Evaluation

During the evaluation the agent starts in a random or predefined state (e.g. expected landmark location based on the ground-truth), and follows the computed policy, iterating through the state space until an oscillation occurs (an infinite loop between two neighboring states). The end state is considered a high-confidence solution for the position of the target landmark, if the expected reward  $\max_a Q^*(s_{target}, a) < 1$  (closer than one pixel). If this is not the case, then we consider that the search failed. This gives us a powerful confidence measure for the performance of the agent (see Fig. 3.12B). Using this property we not only detect diverging trajectories, but can also recognize if the landmark is not contained in the image. For this we evaluated trained agents on 100 long-axis cardiac MR images from different patients, showing that in such cases the oscillation occurs at points where the expected future reward is significantly high – at least 4 (see Fig. 3.12C). This suggests the ability of our algorithm to detect when the anatomical landmark is not in the image. Regarding the relation between trajectory convergence and start-state, we observed in randomly selected test images that more than 90% of the possible starting points converge to the correct solution (see Fig. 3.12A). In other words, with only 3 random attempts, the probability of diverging is less than 0.1%. Fig. 3.13 shows examples of trajectories.

---

## 3.5 CONCLUSION

In this chapter we presented Marginal Space Deep Learning as an efficient solution for parsing volumetric medical image data. By exploiting the computational efficiency of learning in marginal parameter spaces and the discriminative power

**FIGURE 3.13**

Example strategies, i.e. trajectories, the considered anatomical landmarks. The blue point denotes the starting position (chosen randomly), the red point the ground-truth position, while green denotes the detection result. The trajectory is visualized as a white curve.

of deep neural network representations, we managed to build a system that significantly outperforms the state-of-the-art. However, the performance is achieved at the cost of tedious scanning efforts that are necessary for the parameter estimation, efforts which limit the scalability of the algorithm to more complex transformations. In this context we proposed an alternative solution that couples the learning of the appearance model and the parameter search as a unified behavioral task for an artificial agent. Instead of following exhaustive search schemes, the algorithm finds an optimal, strategic path in parameter space converging to the optimal location. We showed on the example of landmark detection that this alternative solution achieves accurate results and addresses the computational limitations of exhaustive scanning. Our future work is focused on extending this framework for generic image parsing.

---

## DISCLAIMER

This feature is based on research, and is not commercially available. Due to regulatory reasons, its future availability cannot be guaranteed.

---

## REFERENCES

1. Yoshua Bengio, Aaron C. Courville, Pascal Vincent, Unsupervised feature learning and deep learning: a review and new perspectives, arXiv:1206.5538, 2012.

2. David G. Lowe, Object recognition from local scale-invariant features, in: ICCV, vol. 2, 1999, pp. 1150–1157.
3. Navneet Dalal, Bill Triggs, Histograms of oriented gradients for human detection, in: Conference on Computer Vision and Pattern Recognition, 2005, pp. 886–893.
4. P. Viola, M. Jones, Rapid object detection using a boosted cascade of simple features, in: Conference on Computer Vision and Pattern Recognition, vol. 1, 2001, pp. 511–518.
5. Yefeng Zheng, Adrian Barbu, Bogdan Georgescu, Michael Scheuering, Dorin Comaniciu, Four-chamber heart modeling and automatic segmentation for 3-D cardiac CT volumes using marginal space learning and steerable features, IEEE TMI 27 (11) (2008) 1668–1681.
6. Yefeng Zheng, David Liu, Bogdan Georgescu, Hien Nguyen, Dorin Comaniciu, 3D deep learning for efficient and robust landmark detection in volumetric data, in: Nassir Navab, Joachim Hornegger, M. William Wells, F. Alejandro Frangi (Eds.), MICCAI 2015, Part I, in: Lect. Notes Comput. Sci., vol. 9349, Springer International Publishing, 2015, pp. 565–572.
7. T.F. Cootes, C.J. Taylor, D.H. Cooper, J. Graham, Active shape models – their training and application, Comput. Vis. Image Underst. 61 (1) (1995) 38–59.
8. Florin C. Ghesu, Bogdan Georgescu, Yefeng Zheng, Joachim Hornegger, Dorin Comaniciu, Marginal space deep learning: efficient architecture for detection in volumetric image data, in: Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015 – 18th International Conference, Part I, Munich, Germany, October 5–9, 2015, 2015, pp. 710–718.
9. F.C. Ghesu, E. Krubasik, B. Georgescu, V. Singh, Y. Zheng, J. Hornegger, D. Comaniciu, Marginal space deep learning: efficient architecture for volumetric image parsing, IEEE TMI 35 (5) (2016) 1217–1228.
10. Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, Ruslan Salakhutdinov, Dropout: a simple way to prevent neural networks from overfitting, J. Mach. Learn. Res. 15 (2014) 1929–1958.
11. F.C. Ghesu, B. Georgescu, T. Mansi, D. Neuman, J. Hornegger, D. Comaniciu, An artificial agent for anatomical landmark detection in medical images, in: S. Ourselin, L. Joskowicz, M.R. Sabuncu, G. Unal, W. Wells (Eds.), Medical Image Computing and Computer-Assisted Intervention – MICCAI 2016: 19th International Conference, Part III, Athens, Greece, October 17–21, 2016, 2016, pp. 229–237.
12. Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, Demis Hassabis, Human-level control through deep reinforcement learning, Nature 518 (7540) (2015) 529–533.
13. Y. Lecun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proc. IEEE 86 (11) (1998) 2278–2324.
14. Geoffrey E. Hinton, Simon Osindero, Yee-Whye Teh, A fast learning algorithm for deep belief nets, Neural Comput. 18 (7) (2006) 1527–1554.
15. C.J.C.H. Watkins, P. Dayan, Q-learning, Mach. Learn. 8 (3) (1992) 279–292.
16. Martin Riedmiller, Neural fitted Q iteration – first experiences with a data efficient neural reinforcement learning method, in: João Gama, Rui Camacho, Pavel B. Brazdil, Alípio Mário Jorge, Luís Torgo (Eds.), Machine Learning: ECML 2005, in: Lect. Notes Comput. Sci., vol. 3720, Springer, Berlin, Heidelberg, 2005, pp. 317–328.
17. Hans C. van Assen, Mikhail G. Danilouchkine, Alejandro F. Frangi, Sebastián Ordas, Jos J.M. Westenberg, Johan H.C. Reiber, Boudeijn P.F. Lelieveldt, SPASM: a 3D-ASM for

- segmentation of sparse and arbitrarily oriented cardiac MRI data, *Med. Image Anal.* 10 (2) (2006) 286–303.
- 18. Alison M. Pouch, Hongzhi Wang, Manabu Takabe, Benjamin M. Jackson, Joseph H. Gorman III, Robert C. Gorman, Paul A. Yushkevich, Chandra M. Sehgal, Fully automatic segmentation of the mitral leaflets in 3D transesophageal echocardiographic images using multi-atlas joint label fusion and deformable medial modeling, *Med. Image Anal.* 18 (1) (2014) 118–129.
  - 19. Robert J. Schneider, Neil A. Tenenholtz, Douglas P. Perrin, Gerald R. Marx, Pedro J. del Nido, Robert D. Howe, Patient-specific mitral leaflet segmentation from 4D ultrasound, in: *MICCAI 2011*, 2011, pp. 520–527.
  - 20. Steven C. Mitchell, Johan G. Bosch, Boudewijn P.F. Lelieveldt, Rob J. van der Geest, Johan H.C. Reiber, Milan Sonka, 3-D active appearance models: segmentation of cardiac MR and ultrasound images, *IEEE Trans. Med. Imaging* 21 (9) (2002) 1167–1178.
  - 21. F. Tombari, L. Di Stefano, 3D data segmentation by local classification and Markov random fields, in: *Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission, 2011*, 2011, pp. 212–219.
  - 22. Mustafa Elattar, Esther Wiegerinck, Floortje Kesteren, Lucile Dubois, Nils Planken, Ed Vanbavel, Jan Baan, Henk Marquering, Automatic aortic root landmark detection in CTA images for preprocedural planning of transcatheter aortic valve implantation, *Int. J. Cardiovasc. Imaging* (2015) 1–11.
  - 23. Alison M. Pouch, Sijie Tian, Manabu Takabe, Jiefu Yuan, Robert Gorman Jr., Albert T. Cheung, Hongzhi Wang, Benjamin M. Jackson, Joseph H. Gorman III, Robert C. Gorman, Paul A. Yushkevich, Medially constrained deformable modeling for segmentation of branching medial structures: application to aortic valve segmentation and morphometry, *Med. Image Anal.* 26 (1) (2015) 217–231.
  - 24. Xavier Glorot, Antoine Bordes, Yoshua Bengio, Deep sparse rectifier neural networks, in: Geoffrey J. Gordon, David B. Dunson (Eds.), *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, AISTATS-11, J. Mach. Learn. Res. Workshop Conf. Proc.* 15 (2011) 315–323, <http://www.jmlr.org/proceedings/papers/v15/glorot11a/glorot11a.pdf>.
  - 25. B. van Ginneken, A.F. Frangi, J.J. Staal, B.M. ter Haar Romeny, M.A. Viergever, Active shape model segmentation with optimal features, *IEEE Trans. Med. Imaging* 21 (8) (2002) 924–933.
  - 26. D.R. Martin, C.C. Fowlkes, J. Malik, Learning to detect natural image boundaries using local brightness, color, and texture cues, *IEEE Trans. Pattern Anal. Mach. Intell.* 26 (5) (2004) 530–549.
  - 27. A.M. Turing, *Computing Machinery and Intelligence*, 1950.
  - 28. K. Appel, W. Haken, J. Koch, Every planar map is four colorable. Part II: Reducibility, *Illinois J. Math.* 21 (3) (1977) 491–567.
  - 29. Hee-Jun Park, Byung Kook Kim, Kye Young Lim, Measuring the machine intelligence quotient (MIQ) of human–machine cooperative systems, *IEEE Trans. Syst. Man Cybern., Part A, Syst. Hum.* 31 (2) (2001) 89–96.
  - 30. Stuart J. Russell, Peter Norvig, *Artificial Intelligence: A Modern Approach*, second ed., Pearson Education, ISBN 0137903952, 2003.
  - 31. E.L. Thorndike, *Animal Intelligence: Experimental Studies*, Animal Behavior Series, Macmillan, 1911.
  - 32. Richard S. Sutton, Andrew G. Barto, *Introduction to Reinforcement Learning*, first ed., MIT Press, 1998.

33. Richard Bellman, Dynamic Programming, first ed., Princeton University Press, Princeton, NJ, USA, 1957.
34. J.N. Tsitsiklis, B. Van Roy, An analysis of temporal-difference learning with function approximation, *IEEE Trans. Autom. Control* 42 (5) (1997) 674–690, <http://dx.doi.org/10.1109/9.580874>.
35. Brian Sallans, Geoffrey E. Hinton, Reinforcement learning with factored states and actions, *J. Mach. Learn. Res.* 5 (2004) 1063–1088.
36. S. Lange, M. Riedmiller, Deep auto-encoder neural networks in reinforcement learning, in: The 2010 International Joint Conference on Neural Networks (IJCNN), 2010, pp. 1–8.
37. Nicolas Heess, David Silver, Yee Whye Teh, Actor-critic reinforcement learning with energy-based policies, in: Marc Peter Deisenroth, Csaba Szepesvári, Jan Peters (Eds.), *EWRL*, *J. Mach. Learn. Workshop Conf. Proc.* 24 (2012) 43–58.
38. D.H. Hubel, T.N. Wiesel, Shape and arrangement of columns in cat's striate cortex, *J. Physiol.* 165 (3) (1963) 559–568, <http://dx.doi.org/10.1113/jphysiol.1963.sp007079>.
39. D.E. Rumelhart, G.E. Hinton, R.J. Williams, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, MIT Press, 1986, pp. 318–362.
40. Long-Ji Lin, Reinforcement Learning for Robots Using Neural Networks, PhD thesis, Pittsburgh, PA, USA, 1992.
41. Razvan Ioan Ionasec, Ingmar Voigt, Bogdan Georgescu, Yang Wang, Helene Houle, Fernando Vega-Higuera, Nassir Navab, Dorin Comaniciu, Patient-specific modeling and quantification of the aortic and mitral valves from 4-D cardiac CT and TEE, *IEEE Trans. Med. Imaging* 29 (9) (2010) 1636–1651.
42. Xiaoguang Lu, Bogdan Georgescu, Marie-Pierre Jolly, Jens Guehring, Alistair Young, Brett Cowan, Arne Littmann, Dorin Comaniciu, Cardiac anchoring in MRI through context modeling, in: Tianzi Jiang, Nassir Navab, Josien P.W. Pluim, Max A. Viergever (Eds.), *MICCAI 2010, Part I*, in: *Lect. Notes Comput. Sci.*, vol. 6361, Springer, Heidelberg, 2010, pp. 383–390.
43. Xiaoguang Lu, Marie-Pierre Jolly, Discriminative context modeling using auxiliary markers for LV landmark detection from a single MR image, in: Oscar Camara, Tommaso Mansi, Mihaela Pop, Kawal Rhode, Maxime Sermesant, Alistair Young (Eds.), *STACOM 2013*, in: *Lect. Notes Comput. Sci.*, vol. 7746, Springer, Heidelberg, 2013, pp. 105–114.

This page intentionally left blank

# Multi-Instance Multi-Stage Deep Learning for Medical Image Recognition

Zhennan Yan<sup>\*</sup>, Yiqiang Zhan<sup>†</sup>, Shaoting Zhang<sup>‡</sup>, Dimitris Metaxas<sup>\*</sup>,  
Xiang Sean Zhou<sup>†</sup>

Rutgers University, Piscataway, NJ, United States<sup>\*</sup> Siemens Healthcare, Malvern, PA,  
United States<sup>†</sup> University of North Carolina at Charlotte, Charlotte, NC, United States<sup>‡</sup>

## CHAPTER OUTLINE

<b>4.1</b>	<b>Introduction</b>	83
<b>4.2</b>	<b>Related Work</b>	85
<b>4.3</b>	<b>Methodology</b>	87
4.3.1	Problem Statement and Framework Overview	87
4.3.2	Learning Stage I: Multi-Instance CNN Pre-Train	88
4.3.3	Learning Stage II: CNN Boosting	90
4.3.4	Run-Time Classification	92
<b>4.4</b>	<b>Results</b>	93
4.4.1	Image Classification on Synthetic Data	93
4.4.2	Body-Part Recognition on CT Slices	95
<b>4.5</b>	<b>Discussion and Future Work</b>	99
<b>References</b>		100

## 4.1 INTRODUCTION

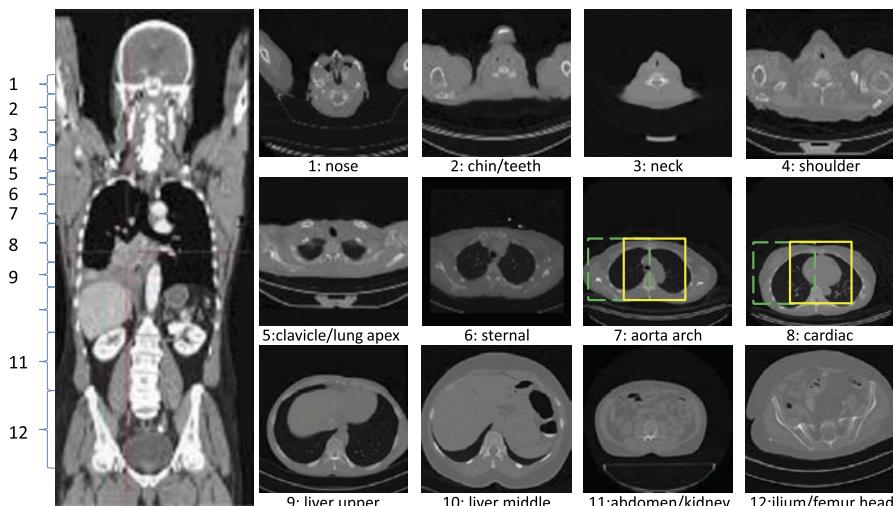
In medical image analysis, clinical information representation and extraction is the primary goal, and the basis of many complicated frameworks. Many automatic or semi-automatic algorithms have been developed in this community [1–4]. They were designed to assist clinicians or researchers to interpret and assess medical images in different applications, from fundamental tasks, e.g. anatomical landmark detection [5,6] and organ segmentation [7–9], to complicated computer aided diagnosis (CAD) systems [10–12]. Since different organ systems have highly diverse characteristics, medical image analysis models are usually trained or designed for specific anatomies to incorporate appropriate prior knowledge. Such ad-hoc models using handcraft features are not easily extensible to different use cases.

In the last decade, deep learning [13] methods have shown successful outcomes in different applications, including signal processing [14], object recognition [15], natural language processing [16], etc. Their attraction comes from the automatic feature learning ability in a deep network architecture. The deep network architecture uses multiple layers of simple but nonlinear activation functions to transform the input data to multiple levels of feature representation, from low (detail) level to high (abstract) level. The network is able to learn such hierarchical feature representations in unsupervised or supervised way from a large amount of training data by itself. Such learned hierarchical features have been proved to be superior to ad-hoc designed ones in a wide range of practical applications [17]. Image recognition [15] is one of the most promising applications, where deep learning learns good representation of visual features. Thus, deep learning should be able to benefit the medical image recognition as well. The goal of this chapter is to introduce recent progress of deep learning based methods in medical image recognition, in particular a medical image recognition algorithm using multi-stage multi-instance deep learning.

A common form of medical data is imaging scans, e.g. CT and MR. A typical imaging scan is a 3D volume image consisted of a series of 2D slices. Although image scans are usually acquired for specific body parts in most clinical activities, whole body scans are also used to give a holistic view of diseases. For large scale medical image analysis, the data are very likely collected from different institutes with various hardware and protocols. These complicated data set always need complex preprocessing, such as data selection, alignment and anatomy localization, before applying some automatic frameworks for a research study. In such cases, it is important to have a reliable automatic module that can recognize image contents in the first place. For example, we would like to have a program, which can act like a medical professional who can tell quickly anatomies and their rough positions in an image at a glance. With the correct first impression, automated workflow can conduct easy preprocessing and other higher level jobs (for example, detection and segmentation) using appropriate method or model.

In this chapter, we mainly focus on the medical image recognition task to identify anatomies contained in the input image, namely “body-part recognition”. Although organ segmentation and landmark detection topics have been extensively investigated, the automatic body-part recognition (identifying the human body parts contained in the medical image) is still less explored. This task can be easily defined as an image classification problem in 2D setting. For example, assuming the human body is divided into continuous sections according to anatomical context as shown in Fig. 4.1, the task is, in fact, a multi-class image classification problem. Given a 2D transversal slice, the goal is to identify what body section the image belongs to. Although 3D volume always contain more comprehensive anatomy information, a 2D slice-based anatomy recognition algorithm provides the foundation of 3D recognition, and is efficient and practical when 3D data is not available. This chapter does not intend to introduce a new method but to present our existing work on deep learning [18,19].

It is worth noting that although DICOM header includes anatomy information, text-based retrieval is not an ideal choice due to three major challenges. First, it may

**FIGURE 4.1**

Definition of body sections. Human body is divided into 12 continuous parts. Each part may cover different ranges due to the variability of anatomies. Yellow boxes indicate the discriminative local regions in aorta arch class and cardiac class, while green boxes indicate ambiguous local regions for this classification task.

contain 15% of errors in DICOM headers [20]. Second, text information in DICOM is sometimes too abstract to precisely describe the anatomies contained in the scan. Third, the multi-language supporting of DICOM header becomes another barrier for text-based retrieval. On the contrary, a reliable image-based anatomy recognition algorithm can tackle all these three challenges by learning the intrinsic anatomical appearance information.

## 4.2 RELATED WORK

Deep learning utilizes massive amounts of computational power and achieves state-of-the-art results on various challenging tasks [17,13]. Among different deep learning methods, convolutional neural network (CNN) [21] based algorithms are more suitable in image related tasks, since images have highly correlated intensities in local regions and some local signals or statistics are invariant to location. In computer vision community, different CNN based methods [15,22,23] have shown their superiority in image classification tasks compared to conventional approaches with carefully designed/selected features, e.g. SVM and logistic regression [24,25] with SIFT [26] or HOG [27] features. Despite this success, their application in medical image analysis remains to be fully explored.

Recently, researchers have began to apply deep learning in CAD tasks, including detection [28–30], segmentation [31–34], disease classification [35], etc. These methods are proposed for specific anatomies and tasks. As discussed before, it would be useful to have an image-based anatomy recognition method in the CAD system. Roth et al. [36] presented a method for anatomy-specific classification of medical images using CNN. They applied a standard deep CNN on 2D axial CT images to classify 5 anatomies (neck, lungs, liver, pelvis, and legs) and obtained the state-of-the-art accuracy (5.9% error). In this slice-based anatomy recognition, the standard CNN is conducted as a *global* learning scheme, which takes the entire image as input. The CNN successfully learned feature representations to capture the diverse appearances in the five body sections. However, this standard CNN is not easily scalable to handle more detailed anatomy recognition (as shown in Fig. 4.1) effectively, since distinctive information often comes from *local* patches and these local patches are distributed “inconsistently” at different positions of the slices. As shown in Fig. 4.1, aorta arch section and cardiac section have globally similar appearance characteristics, while the discriminative information only resides in the local mediastinum region (indicated by the yellow boxes). The other areas are just “non-informative” or misleading for classification purposes. One may argue that CNN can still learn local features through its convolutional layers. However, this situation only holds while local features always appear at the similar location across different images, which is not the case of body-part recognition.

In fact, this problem also exists in general image classification/recognition applications in computer vision. Researchers are trying to leveraging local region information to train CNN for recognition or classification. For example, in face recognition [37], the authors first detect and align face regions properly before training CNN. In another pioneer work, Wei et al. [23] applied an existing objectness detector [38] to produce some local region proposals from a given image, and used them to train multi-label CNN classifier. Similarly, Girshick [39] trained a region-based CNN (fast R-CNN) based on existing object proposals and used a multi-task loss function to learn the classifier and bounding-box regressor for efficient object detection. Then, Ren et al. [40] proposed a faster R-CNN detection by using a region proposal network which shares convolutional features with the detection network. Despite promising results these methods generate, they all require manual annotations of local regions of objects in images for training. However, the discriminative local regions for body section recognition are not easy to define, not to mention that the effort to build these local detectors might be quite large. Note that organ/landmark detector based anatomy recognition approaches are limited due to the tedious manual annotation efforts in the training stage and complicated inference efforts in testing stage [41–43].

To avoid explicit local region or object annotation, several studies [44–46] have emerged to incorporate multi-instance learning (MIL) [47] with CNN to better utilize local information in weakly supervised learning fashion. Yan et al. [18,19] proposed a fine-grained body-part recognition method based on CNN and MIL. It only requires image level label to “discover” the discriminative local regions automatically and use multi-stage learning to utilize the discovered local information to generate state-of-

the-art results for image classification. In this way, the annotation efforts for local regions in the training stage are totally eliminated. This is in particular meaningful for medical image applications, since the annotations in medical images always require clinical expertise and high cost.

In this chapter, we introduce the technical details of the multi-stage multi-instance deep learning for medical image classification, specifically with the use case of body-part recognition in image slices.

## 4.3 METHODOLOGY

### 4.3.1 PROBLEM STATEMENT AND FRAMEWORK OVERVIEW

**Definitions.** Slice-based body-part recognition is a typical multi-class image classification problem for a learning algorithm. Denote by  $\mathbf{X}$  the input slice/image, by  $K$  the number of body sections (classes), and by  $l \in \{1, \dots, K\}$  the corresponding class label of  $\mathbf{X}$ . The learning algorithm aims to find a function  $\mathcal{O} : \mathbf{X} \rightarrow l$ . In traditional image classification frameworks,  $\mathcal{O}$  is often defined as  $\mathbb{C}(\mathbb{F}(\mathbf{X}))$ , where  $\mathbb{F}(\mathbf{X})$  and  $\mathbb{C}(\cdot)$  denote the feature extractors and classifiers, respectively.

In the context of convolutional neural network (CNN),  $\mathcal{O}$  becomes a multi-layer neural network. An example of standard CNN is shown in Fig. 4.2 (similar to LeNet [21]), it has two convolutional layers (C1, C3), each followed by a max-pooling layer (S2, S4), one fully connected hidden layer (H5) receiving outputs of the last pooling layer, and one logistic regression (LR) layer (O6) as the output layer. In CNN,  $\mathbb{F}(\mathbf{X})$  becomes multiple nonlinear layers, which aim to extract image features in a local-to-global fashion.  $\mathbb{C}(\cdot)$  is implemented by the LR layer, whose output is a  $K$ -dimension vector  $R(k)$ ,  $k \in \{1, \dots, K\}$ , representing the probability of  $\mathbf{X}$  belonging to each class  $k$ . Mathematically,  $R(k)$  can be described as a conditional probability  $R(k) = \mathbf{P}(k|\mathbf{X}; \mathbf{W})$ . Here,  $\mathbf{W}$  denotes the CNN coefficients, which include the weights of convolutional filters, hidden nodes, LR nodes, as well as the bias vectors. The final predicted label  $l$  is determined by the argument of the maximum element (class with the highest probability) in  $R$ .

Given a set of training images  $\mathcal{T} = \{\mathbf{X}_m, m = 1, \dots, M\}$ , with corresponding discrete labels  $l_m \in \{1, \dots, K\}$ , the training algorithm of CNN aims to minimize the loss function

$$L_1(\mathbf{W}) = \sum_{\mathbf{X}_m \in \mathcal{T}} -\log(\mathbf{P}(l_m|\mathbf{X}_m; \mathbf{W})), \quad (4.1)$$

where  $\mathbf{P}(l_m|\mathbf{X}_m; \mathbf{W})$  indicates the probability of image  $\mathbf{X}_m$  being correctly classified as class  $l_m$  using network coefficients  $\mathbf{W}$ .

Here, a multi-instance multi-stage deep learning framework is designed to “discover” discriminative and non-informative local regions for precise image classification without time-consuming local manual annotations, and learn classifier in the meantime. In the first stage, a CNN is learned in a multi-instance learning fashion

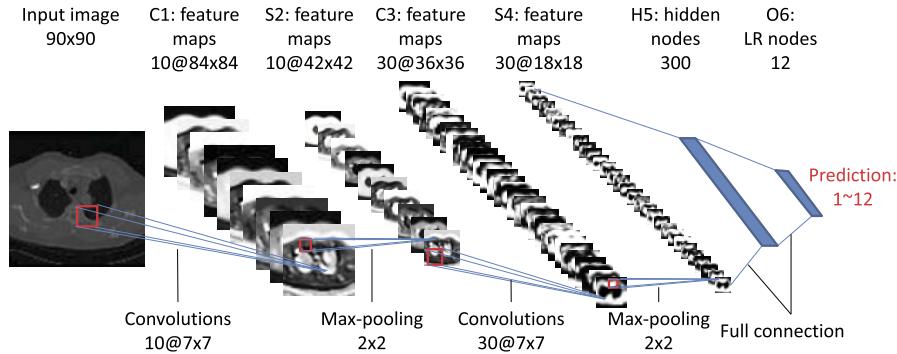
**FIGURE 4.2**

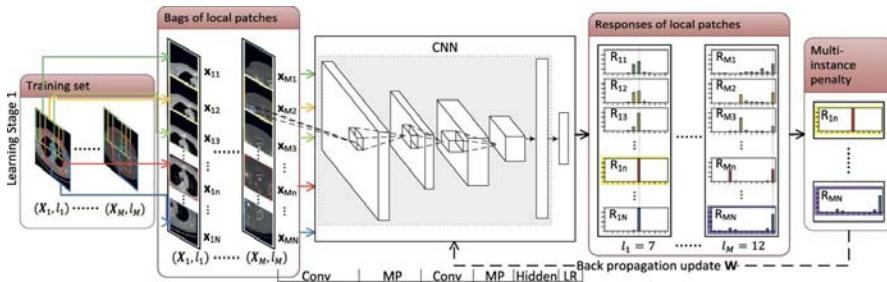
Illustration of one standard CNN architecture and the outputs of each layer.

to discover the most discriminative local patches. Specifically, each image is divided into several local patches. The deep network thus receives a set of labeled images (bags), each containing multiple local patches (instances). The loss function of the CNN is chosen in a way that as long as one local patch (instance) is correctly classified, the labeling of corresponding image slice (bag) is considered to be correct. In this way, the pre-trained CNN will be more sensitive (responds with significantly higher probability at the correct label) to the discriminative local patches than others. Based on the responses of the pre-trained CNNs, discriminative and non-informative local patches are selected to further boost the pre-trained CNN in the second stage (namely boosting stage) of our learning scheme. At run-time, a sliding window approach is employed to apply the boosted CNN to the target image. As the CNN is sensitive to the discriminative local patches, it essentially identifies a body part by focusing on the most distinctive local information and discarding non-informative local regions.

### 4.3.2 LEARNING STAGE I: MULTI-INSTANCE CNN PRE-TRAIN

In order to exploit the local information, CNN should take *discriminative* local patches instead of the entire slice as its input. Here, the key problem is how to automatically *discover* these local patches through learning. This is a major task of the first stage of our CNN learning framework. A multi-instance learning strategy is designed to achieve this goal.

Given a training set  $\mathcal{T} = \{\mathbf{X}_m, m = 1, \dots, M\}$  with corresponding labels  $l_m$ . Each training image,  $\mathbf{X}_m$ , is divided into a set of local patches defined as  $\mathcal{L}(\mathbf{X}_m) = \{\mathbf{x}_{mn}, n = 1, \dots, N\}$ . These local patches become the basic training samples of the CNN and their labels are inherited from the original images, i.e., all  $\mathbf{x}_{mn} \in \mathcal{L}(\mathbf{X}_m)$  share the same label  $l_m$ . While the structure of CNN is still the same as the standard



**FIGURE 4.3**

Illustration of the pre-train stage. In this stage, the CNN is trained in a multi-instance fashion. For instance, yellow highlighted response of instance  $x_{1n}$  from image  $X_1$  and purple highlighted response of instance  $x_{MN}$  from image  $X_M$  are picked to compute the loss to update the CNN parameters in a training iteration. They are picked because they have higher response on the correct label than other instances from the same image. Those local patches which can be easily and correctly classified are considered as the discriminative information for image classification. After training, the CNN will become sensitive to those discriminative local regions.

one, the loss function is

$$L_2(\mathbf{W}) = \sum_{\mathbf{X}_m \in \mathcal{T}} -\log(\max_{\mathbf{x}_{mn} \in \mathcal{L}(\mathbf{X}_m)} \mathbf{P}(l_m | \mathbf{x}_{mn}; \mathbf{W})), \quad (4.2)$$

where  $\mathbf{P}(l_m | \mathbf{x}_{mn}; \mathbf{W})$  is the probability that the local patch  $\mathbf{x}_{mn}$  is correctly classified as  $l_m$  using CNN coefficients  $\mathbf{W}$ .

The new loss function is different from Eq. (4.1) by adopting a multi-instance learning criterion. Here, each original training slice  $\mathbf{X}_m$  is treated as a bag consisting of multiple instances (local patches),  $\{\mathbf{x}_{mn}\}$ . Within each bag (slice), only the instance with the highest probability to be correctly classified is counted in the loss function. This instance is considered as the most discriminative local patch of the image slice. Let  $R_{mn}$  be the output vector of the CNN on local patch  $\mathbf{x}_{mn}$ . The  $l_m$ th component of  $R_{mn}$  represents the probability of  $\mathbf{x}_{mn}$  being correctly classified. As illustrated in Fig. 4.3, for each training image  $\mathbf{X}_m$ , only the local patch that has the highest response at the  $l_m$ th component of  $R_{mn}$  (indicated by the yellow and purple boxes for two training images, respectively), contributes to the loss function and drives the update of network coefficients  $\mathbf{W}$  during the backward propagation. Accordingly, the learned CNN is expected to have high responses on discriminative local patches. In other words, the most discriminative local patches for each image class are automatically discovered after the CNN training.

We design a toy example to illustrate this discovery ability. As shown in Fig. 4.4A, four types of geometry elements, namely, square, circle, triangle, and diamond, are randomly positioned and combined to generate two classes of binary images. While circle and diamond are allowed to appear in any classes, triangle and square are ex-

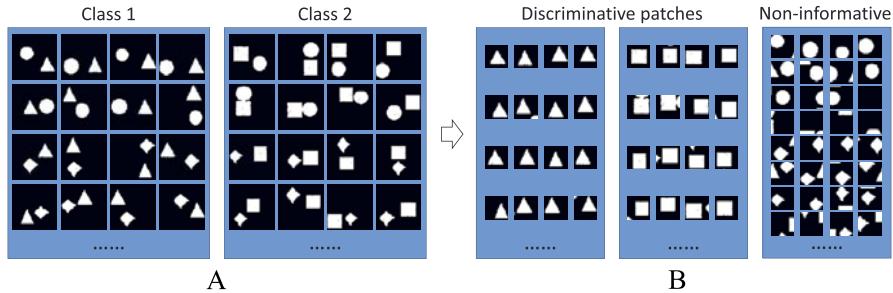


FIGURE 4.4

A synthetic toy example. (A) Synthetic images of two classes. (B) The discriminative and non-informative local patches selected by the pre-trained CNN model. Note that we never “tell” the algorithm that these two classes are differentiable by triangle and square.

clusively owned by Class 1 and Class 2, respectively. Fig. 4.4B shows the discovered discriminative patches (containing triangle or square) for the image classification task in toy example. This is exactly in accordance with the fact that these two classes are only distinguishable by “triangle” and “square”. It proves that our method is able to *discover* the key local patches without manual annotation. Of course, this problem would become trivial if we have prior knowledge of the discriminative local patches and build specific classifiers on them. However, in real-world recognition tasks, it is not easy to figure out the most discriminative local patches for different classes. In addition, even with *ad hoc* knowledge, annotating local patches and training local classifiers often takes large effort. The solution thus becomes non-scalable.

To ensure that the learned CNN will have stable high responses on discriminative local patches, a spatial continuity factor is further incorporated into the loss function as

$$L_3(\mathbf{W}) = \sum_{\mathbf{x}_m \in \mathcal{T}} -\log(\max_{\mathbf{x}_{mn} \in \mathcal{L}(\mathbf{X}_m)} \sum_{\mathbf{x} \in \mathfrak{N}(\mathbf{x}_{mn})} \mathbf{P}(l_m | \mathbf{x}; \mathbf{W})). \quad (4.3)$$

Here,  $\mathfrak{N}(\mathbf{x}_{mn})$  denotes the local patches in the neighborhood of  $\mathbf{x}_{mn}$ . Based on Eq. (4.3), for each training slice, the local patch to be counted in the loss function is not the most *individually* discriminative one (i.e., with the highest probability of being correctly classified), but the one whose neighboring patches and itself are *overall* most discriminative. In this way, the selected discriminative local patches will be robust to image translation and artifacts.

### 4.3.3 LEARNING STAGE II: CNN BOOSTING

In the second stage of our learning framework, the main task is to boost the pre-trained CNN using selected local patches, which is illustrated in Fig. 4.5.

The first type of selected local patches are the discriminative ones, i.e., these local patches on which the pre-trained CNN have high responses at the corresponding

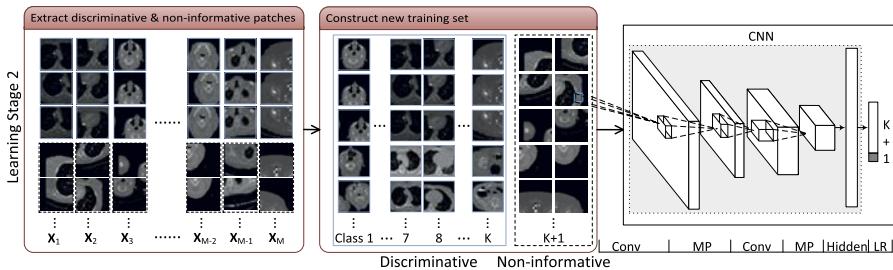
**FIGURE 4.5**

Illustration of boosting stage. In this stage, CNN architecture is modified by adding a non-informative class in output layer. The parameters are inherited from the pre-trained CNN and fine-tuned using the discriminative and non-informative local regions extracted from each class by the pre-trained CNN.

classes. For each image  $\mathbf{X}_m$ , we select  $D$  discriminative local patches as

$$\mathbf{A}_m = \underset{\mathbf{x}_{mn} \in \mathcal{L}(\mathbf{X}_m)}{\operatorname{argmax}_D} \mathbf{P}(l_m | \mathbf{x}_{mn}; \hat{\mathbf{W}}). \quad (4.4)$$

Here,  $\hat{\mathbf{W}}$  is the coefficients of the pre-trained CNN.  $\mathbf{P}(l_m | \mathbf{x}_{mn}; \hat{\mathbf{W}})$  denotes the response of the pre-trained CNN on the local patch  $\mathbf{x}_{mn}$  corresponding to the correct class  $l_m$ .  $\operatorname{argmax}_D(\cdot)$  is the operator that returns the arguments of the largest  $D$  elements.

We noticed that apart from the discriminative local patches, the remaining regions cannot be completely ignored in the boosting stage for two reasons. First, only selecting discriminative patches to boost classifier may lead to overfitting problems. Second, some “confusing” local patches may mislead the body-part recognition. For example, the patches containing lung regions (green dashed boxes in Fig. 4.1) appear in both aortic arch and cardiac sections. For these “confusing” patches, CNN may generate similarly high responses for both aortic arch and cardiac classes. (Note that since the pre-trained CNN is only ensured to correctly classify one local patch per slice, the responses of the remaining patches are not guaranteed.) At run-time, when CNN is applied to the confusing patches, the high responses on multiple classes may induce wrong body-part identification. Therefore, the algorithm should select these “confusing” regions as the second type of local patches in boosting stage to suppress their responses for *all* classes (body sections).

To this end, we introduce a new “non-informative” class (patches in dashed box in Fig. 4.5) besides the existing training classes. This class includes two kinds of local patch: (i) local patch where the pre-trained CNN has higher responses on wrong classes, and (ii) local patch where the pre-trained CNN has “flat” responses across all classes. Denote by  $\mathbf{P}(k | \mathbf{x}_{mn}; \hat{\mathbf{W}})$  the  $k$ th output of the pre-trained CNN on  $\mathbf{x}_{mn}$ , i.e., the probability of  $\mathbf{x}_{mn}$  belonging to class  $k$ , the non-informative local patches of

a training slice  $\mathbf{X}_m$  are defined as:

$$\begin{aligned}\mathbf{B}_m = \{\mathbf{x}_{mn} | \underset{k \in \{1, \dots, K\}}{\operatorname{argmax}} \mathbf{P}(k|\mathbf{x}_{mn}; \hat{\mathbf{W}}) \neq l_m\} \\ \cup \{\mathbf{x}_{mn} | \underset{k \in \{1, \dots, K\}}{\operatorname{entropy}} \mathbf{P}(k|\mathbf{x}_{mn}; \hat{\mathbf{W}}) > \theta\}.\end{aligned}\quad (4.5)$$

Recalling the toy example, Fig. 4.4B shows the selected discriminative and non-informative local patches. When the discriminative patches from Class 1 and Class 2 only contain triangle or square, respectively, the non-informative patches may include circle, diamond, or background. This is exactly in accordance to the fact that these two classes are only distinguishable by “triangle” and “square”, and other components are misleading. It demonstrates the discovery ability of the method.

After introducing the additional non-informative class, the CNN structure keeps the same as the pre-trained CNN, except the LR layer has an additional output (see shadowed box in the rightmost diagram of Fig. 4.5) and the corresponding connections to the hidden layer. Since the pre-trained CNN already captured some discriminative local appearance characteristics, all network layers except the last one are initialized by inheriting their coefficients from the pre-trained CNN. These coefficients are further adapted by minimizing Eq. (4.6):

$$L_4(\mathbf{W}) = \sum_{\mathbf{x} \in \mathbf{A} \cup \mathbf{B}} -\log(\mathbf{P}(l|\mathbf{x}; \mathbf{W})).\quad (4.6)$$

Here,  $\mathbf{A} = \bigcup_{\{m=1, \dots, M\}} \mathbf{A}_m$  and  $\mathbf{B} = \bigcup_{\{m=1, \dots, M\}} \mathbf{B}_m$  denote the discriminative and non-informative local patches selected from all training images, respectively. Note that since the non-informative local patches do not belong to any body section class now, their responses on any body section class can be effectively suppressed during the CNN boosting stage.

#### 4.3.4 RUN-TIME CLASSIFICATION

At runtime, the boosted CNN is applied for body-part recognition in a sliding window fashion. The sliding window partitions a testing image  $\mathbf{X}$  into  $N$  overlapping local patches  $\mathcal{L}(\mathbf{X}) = \{\mathbf{x}_n, n = 1, \dots, N\}$ . For each local patch  $\mathbf{x}_n$ , the boosted CNN outputs a response vector with  $K+1$  components  $\{\mathbf{P}(k|\mathbf{x}_n; \mathbf{W}^{opt}) | k = 1, \dots, K+1\}$ , where  $\mathbf{W}^{opt}$  denotes the optimal coefficients of Eq. (4.6). The class of the local patch  $\mathbf{x}_n$  is then determined as

$$c(\mathbf{x}_n) = \underset{k \in \{1, \dots, K+1\}}{\operatorname{argmax}} \mathbf{P}(k|\mathbf{x}_n; \mathbf{W}^{opt}).\quad (4.7)$$

Since the class  $K+1$  is an artificially constructed non-informative one, local patches belong to this class should be ignored in body section determination. The most discriminative patch  $\mathbf{x}_{n^*}$  in the image is selected as the most peaky correctly

labeled one excluding the non-informative patches:

$$\mathbf{x}_{n^*} = \underset{\mathbf{x}_n \in \mathcal{L}(\mathbf{X}); c(\mathbf{x}_n) \neq K+1}{\operatorname{argmax}} \mathbf{P}(c(\mathbf{x}_n) | \mathbf{x}_n; \mathbf{W}^{opt}). \quad (4.8)$$

To generate reliable classification of the image  $\mathbf{X}$ , the effect of possible outlier  $\mathbf{x}_{n^*}$  with different prediction of its neighbors can be suppressed by a label fusion of patches to label the image. A simple choice of label fusion is combining the class probabilities in the neighborhood around the most discriminative patch:

$$C(\mathbf{X}) = \underset{k \in \{1, \dots, K\}}{\operatorname{argmax}} \sum_{\mathbf{x}_n \in \mathfrak{N}(\mathbf{x}_{n^*})} \mathbf{P}(k | \mathbf{x}_n; \mathbf{W}^{opt}). \quad (4.9)$$

## 4.4 RESULTS

In this study, we mainly compare the multi-instance multi-stage CNN with standard CNN on image classification tasks. In implementation of CNNs, Rectified Linear Units (ReLUs) [48] and Dropout strategy [49] are employed. Dropout rate is 0.5. Data is augmented by up to 10% (relate to image size) random translations to increase training samples. The image patches for multi-instance learning are extracted by fixed-size sliding window with overlapping. As the training set may be too large to load into memory at one time, the models are learned using a mini batch of samples at each iteration. The optimization is implemented by stochastic gradient descent with a momentum term  $\beta$  [50] and a weight decay. The learning process is conducted on a training subset and a validation subset. It will stop if either the error rate on validation subset drops below a threshold or a predefined maximum number of epochs is reached. The framework is implemented in Python using Theano on a 64-bit desktop with i7-2600 (3.4 GHz) CPU, 16 GB RAM and NVIDIA GTX-660 3 GB GPU. Classification accuracies are reported in terms of recall, precision and  $F_1$  score as

$$\text{recall} = \frac{TP}{TP + FN}, \quad \text{precision} = \frac{TP}{TP + FP}, \quad (4.10)$$

$$F_1 = 2 \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}, \quad (4.11)$$

where  $TP$  (true positive) denote the number of samples belonging to class  $k$  and correctly classified;  $FN$  (false negative) denote the number of samples belonging to class  $k$  but misclassified;  $FP$  (false positive) denote the number of samples not belonging to class  $k$  but misclassified as class  $k$ .

### 4.4.1 IMAGE CLASSIFICATION ON SYNTHETIC DATA

A synthetic data set, which has been briefly introduced as a toy example in Section 4.3.2, is constructed by 4 types of geometry elements: triangle, square, circle,

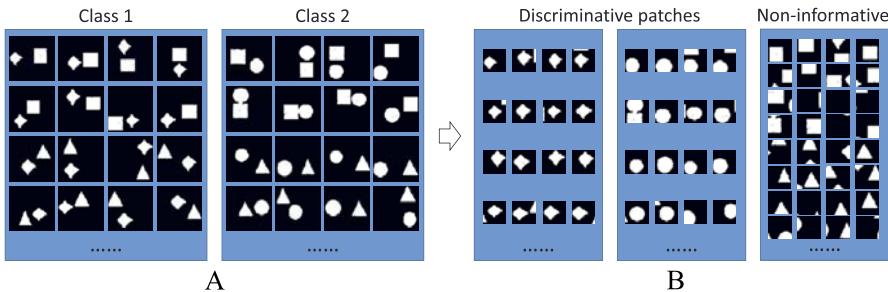
**Table 4.1** Classification accuracies (%) on synthetic data set as shown in Fig. 4.4. Class 1 contains triangle; Class 2 contains square

Class	Triangle and square								
	Recall			Precision			$F_1$		
	1	2	Total	1	2	Total	1	2	Total
SCNN	84.2	82.4	83.3	82.7	83.9	83.3	83.5	83.2	83.3
PCNN	99.6	99.7	99.7	99.7	99.6	99.7	99.7	99.7	99.7
BCNN1	98.4	99.7	99.1	99.7	98.4	99.1	99.0	99.1	99.1
BCNN2	100	100	100	100	99.9	100	100	100	100

and diamond. Each synthetic image ( $60 \times 60$ ) contains two of the geometry elements at random positions on black background (intensity value 0). The basic geometry elements are roughly  $20 \times 20$  with variance in height and width. They have random intensity values in [1, 255]. In constructing the two image classes, we ensure that the triangle and square are the “distinctive” element and only appear in Class 1 and Class 2, respectively. Circle or diamond is evenly picked as the second element in each image. Some examples of the synthetic images are shown in Fig. 4.4A. Overall, we create 2000 training, 2000 validation, and 2000 for testing samples (balanced distribution for each class).

Four different variants of CNN are compared: (i) standard CNN, as shown in Fig. 4.2, trained on whole image (SCNN); (ii) local patch-based CNN without boost, i.e., the CNN trained by pre-train stage only (PCNN); (iii) local patch-based CNN boosted without additional non-informative class (BCNN1); (iv) local patch-based CNN boosted with both discriminative and non-informative patches (BCNN2). Method (i) represents standard CNN learning (using features extracted from whole image). Methods (ii) and (iii) are two variants of our proposed method (iv), which are presented to verify the effects of each component of our method. All CNNs use the same intermediate architecture: one convolutional layer with  $10 5 \times 5$  filters, one max-pooling layer with a  $2 \times 2$  kernel, one hidden layer of 300 nodes, and finally it is followed by an LR layer to output response. The patch size for all patch-based CNNs is  $30 \times 30$ . There are 36 patches extracted from each  $60 \times 60$  image through a sliding window with 6-pixel step size.

As shown in Table 4.1, by leveraging the local discriminative information, PCNN gets  $\approx 16\%$  improvement from SCNN. It implies that standard CNN does not fully discover and learn the discriminative local patches, “triangle” and “square”. On the contrary, the most discriminative and non-informative local patches are effectively discovered by the CNN with multi-instance learning as shown in Fig. 4.4B. Among our local patch-based CNNs (PCNN, BCNN1, and BCNN2), BCNN1 is worse than PCNN due to overfitting on discriminative patches (because the parameters of BCNN1 are initialized by those of PCNN, and refined by training with the extracted discriminative patches only). BCNN2 is similar to BCNN1 but refined by training with discriminative as well as non-informative patches and achieves the best performance.



**FIGURE 4.6**

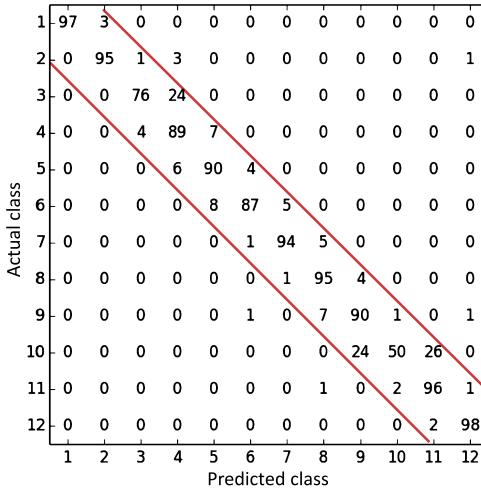
The second toy example. (A) Synthetic images of two classes distinguished by diamond and circle. It is important to note that we used the same image samples as in Fig. 4.3, but re-labeled the images into two classes based on different rules. (B) The discriminative and non-informative local patches discovered by the pre-trained CNN model.

To further prove the key discovery ability of our algorithm, we re-labeled the synthetic data using diamond and circle as distinctive elements in Class 1 and Class 2, respectively (see Fig. 4.6A). In other words, although the synthetic data are exactly the same, the local patches to distinguish the two classes become different. This is in analogy to real-world problems where the datasets are identical but the classification goal is changed. After conducting the first stage learning, the extracted local patches from the learned CNN are shown in Fig. 4.6B. Again, the extracted local patches contain the most discriminative information, diamond and circle. This result demonstrates that our multi-instance CNN learning can *adaptively* learn discriminative local regions for specific classification task without any local level annotations.

#### 4.4.2 BODY-PART RECOGNITION ON CT SLICES

A dataset of 7489 transversal CT slices was collected from whole body scans of 675 patients with very different ages (1–90 year old). The imaging protocols were different: 31 different reconstruction kernels, 0.281–1.953 mm in-slice pixel resolution. This dataset with large variance is good to validate the robustness of the proposed method in practice. As shown in Fig. 4.1, transversal slices of CT scans are categorized into 12 body sections (classes). The body part recognition problem is defined as image classification of the transversal CT slices. The whole dataset is divided into 2413 (225 patients) training, 656 (56 patients) validation, and 4043 (394 patients) testing subsets. We augment data by applying up to 10% random translations in training and validation subsets to make them three times larger.

Our preprocessing includes two different steps: image sub-sampling and image cropping. First, all images are re-sampled to have 4 mm × 4 mm pixel resolution and 90 × 90 in size. Then, cropping operation (for multi-instance learning) extracts 50 × 50 local patches from each image with 10-pixel step size. Thus, 25 local patches

**FIGURE 4.7**

Confusion matrix of BCNN2 on CT data. Values are normalized to 0–100 in each row. Classes are defined in Fig. 4.1.

are extracted per image. Our CNN has similar structure as in Fig. 4.2. C1 layer has 20  $9 \times 9$  filters. C3 layer has 40  $9 \times 9$  filters. Two sub-sampling layer, S2 and S4, use  $2 \times 2$  max-pooling. H5 layer has 600 hidden nodes. LR layer, O6, has 12 output nodes in pre-train stage, or 13 output nodes in boosting stage. The learning process takes 440 epochs ( $\approx 25$  hours) in first stage and 70 epochs ( $\approx 1$  hour) in the second stage.

Fig. 4.7 shows detailed classification performance by the confusion matrix. Most errors appear close to the diagonal line, which means most misclassifications happen between the neighboring body sections. Quantitatively, the classification error is 7.79%, 90% of which being “less-than-one neighboring class error” (within the red line corridor of Fig. 4.7). In practice, this type of error is acceptable for some use cases. The remaining gross error (0.8%) can be further suppressed by a simple label smoothing after classifications of a series of continuous slices for 3D body-part identification.

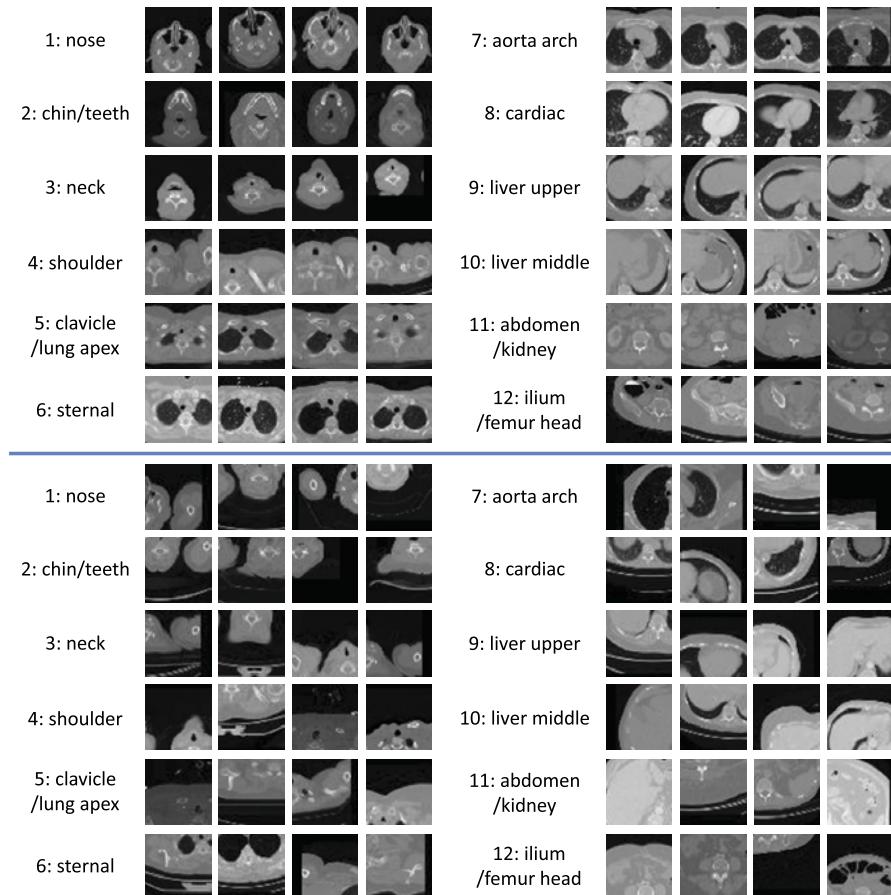
For quantitative comparison (in Table 4.2), tested CNN models include: (i) CaffeNet, (ii) SCNN, (iii) SCNN\_a, (iv) PCNN, (v) BCNN1, and (vi) our proposed BCNN2. SCNN method is the standard CNN that takes the whole slice as input. SCNN\_a method is the same as SCNN except trained by six times more augmented data samples with random transformations, rotations and scalings. Methods (iv) and (v) are the variants of (vi) as described in Section 4.4.1. Similar network structure is used in methods (ii)–(vi), except for different input and output sizes. CaffeNet [51] has a similar structure as AlexNet [15] with a minor variation, which is trained on whole images without cropping. We noticed that training of CaffeNet with  $50 \times 50$

**Table 4.2** Classification accuracies on CT data (%). Classes are defined in Fig. 4.1

Class	F <sub>1</sub> score					
	CaffeNet	SCNN	SCNN_a	PCNN	BCNN1	BCNN2
1	83.15	86.14	88.42	91.84	70.44	<b>92.39</b>
2	78.13	90.38	86.77	93.01	88.48	<b>94.32</b>
3	56.38	76.85	55.62	84.44	61.22	<b>84.47</b>
4	85.75	88.11	86.56	90.44	84.59	<b>90.98</b>
5	82.74	83.89	84.06	87.61	82.88	<b>88.63</b>
6	81.96	81.6	84.59	84.35	87.54	<b>88.42</b>
7	69.65	85.71	86.64	92.37	92.64	<b>92.84</b>
8	89.12	93.89	91.32	95.4	95.31	<b>95.68</b>
9	72.73	77.21	63.56	80.18	<b>81.17</b>	80.38
10	78.13	76.33	69.21	82.46	77.31	<b>84.55</b>
11	84.85	80.89	76.39	83.57	82	<b>89.75</b>
12	98.31	96.52	95.38	95.99	96.91	<b>98.99</b>
Total	85.78	87.73	85.25	90.45	88.2	<b>92.23</b>

cropping doesn't converge. This observation shows that our proposed method is not merely a kind of data augmentation via image cropping. The discriminative and non-informative patches discovered by multi-instance learning are the key to success. BCNN1 is trained on extracted discriminative (without non-informative) patches from learning stage I. Although the trained classifier focuses more on discriminative patches, ambiguous local patches across different classes (e.g. upholding arms may look similar to neck) are completely ignored and thereby mislead the classifier at runtime. Thus, the performance of BCNN1 is worse than PCNN and close to the SCNN. Compared to its variants, the proposed BCNN2 achieves the best performance in the last column of Table 4.2 (significantly better than much deeper CNN, CaffeNet), which proves the necessity of using all strategies designed in our method. In addition, we noted that the SCNN\_a trained with more augmented data is even inferior to the SCNN due to overfitting (training error, SCNN\_a 4.4% vs. SCNN 5%; testing error, SCNN\_a 14.7% vs. SCNN 12.3%). It shows that the global CNN cannot learn the anatomy characteristics from more augmented data and tends to overfit them. As shown in Table 4.2, the overfitting problem is more severe in neck (column 3) and liver upper (column 9) sections. These two sections happen to have subtle global appearance differences compared to their neighboring sections and are thus prone to overfitting. The online classification time is about 3, 4, 4, 10, 11, and 11 ms per image for methods (i) through (vi), respectively. A more detailed comparison can be found in [19].

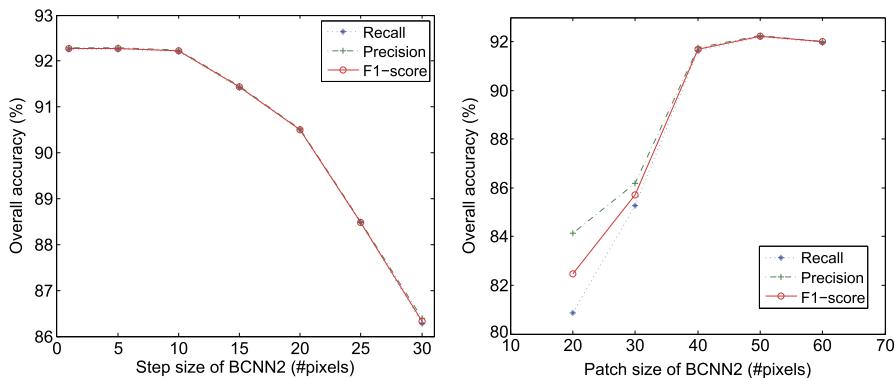
The discovered discriminative patch samples and non-informative (kind of misleading) patches for each class in the CT dataset are shown in Fig. 4.8. From this figure, we observe that the proposed method "magically" extracts meaningful local patches for each class without any prior information, and these discovered local in-

**FIGURE 4.8**

Automatically discovered discriminative and non-informative patches from each class through multi-instance learning.

formation can significantly improve the classification task comparing with the global image information. It is also noted that the discriminative patches of liver middle class contain only a small part of the liver and a narrow band of the left lung bottom. The corresponding non-informative patches may contain a large piece of the liver, since it can appear in different classes (e.g. liver upper and abdomen/kidney). This observation indicates that the proposed method just finds the discriminative information from the data, and does not guarantee that some particular “desired” element will be considered as discriminative.

As one of the important parameters in BCNN2 method, step size of sliding window testing is investigated regarding to the accuracies (shown in Fig. 4.9, left). The

**FIGURE 4.9**

Performance analyses on the sensitivity of parameters in BCNN2. (Left) Classification accuracies vs. step size of sliding window. (Right) Classification accuracies vs. patch size.

running times for step sizes 1, 5, 10, 15, 20, 25, and 30 pixels are 541.1, 30.6, 11.7, 5.3, 5.2, 3.6, and 3.4 ms per image, respectively. Considering the balance of running time and accuracy, step size 10 or 15 is a reasonable choice in this task. The effect of patch size to the classification accuracy is also investigated as shown in Fig. 4.9, right. We can see from the plot that (i) the patch size should not be too small to capture the discriminative information (size 20 or 30); (ii) the performance is not very sensitive to the local patch size once it is big enough to include discriminative information (sizes from 40 to 60 in this task).

## 4.5 DISCUSSION AND FUTURE WORK

In this chapter, we introduced a multi-instance multi-stage deep learning framework for medical image classification. The framework discovers the discriminative and non-informative local patches via multi-instance learning, and employs multi-stage learning strategy to learn CNN for the image classification task. From the validations on synthetic dataset and a large scale CT dataset, we observed clear improvements compared with other state-of-the-art methods. It is proved that the success of the proposed method against the standard CNN does not result from more augmented training samples (see the results of SCNN\_a vs SCNN), but rather results from its capability of *discovering* local characteristics of different body parts.

In fact, the method may benefit radiological workflow in different aspects. For example, a very reliable and fast anatomy recognition algorithm can make the planning of the scanning range in topogram or scout scans be conducted on-the-fly to significantly save scanning time. In another example, an automatic image-based recognition will enable content-based image retrieval and improve the query precision in PACS system. Besides, it can serve as an initialization module for other higher level medical

image interpretation tasks, e.g. anatomy detection or segmentation, to make the workflow more efficient. With the precise search ranges, detection/segmentation speed and robustness can be benefited. Moreover, it can help in medical image preprocessing by gating the applicable auto-algorithms before being loaded for manual reading. In this way, the meaningful and automatic results can be displayed instantaneously in the reading room to speed up radiologists' reading process.

It is worth noting that since no manual annotations are required to label these local patches, our method becomes very scalable. This weakly supervised discriminative patch discovery and classification method can be easily applied to other image classification tasks where local information is critical to distinguish different classes. It can be used to discover and extract discriminative information in different classes to help in obtaining better insight in some problems. One limitation of this method is the identical patch size for different classes [19]. It requires prior knowledge to ensure that the chosen patch size is able to include the discriminative information in a fixed size region. It could be better to incorporate strategies like multi-scale convolution [52] or multi-scale image patch [53] to discover and recognize different-size discriminative local regions in different classes. Another limitation is the sliding-window and multiple stage pipeline. Like the improvement from R-CNN [54] to fast R-CNN [39] and faster R-CNN [40], it may be possible to simplify the multi-stage pipeline to single-stage training, and use part of the convolutional features of image for local regions to avoid the sliding-window strategy in training and testing.

This 2D slice based body part recognition can be trivially applied in 3D image data by labeling each slice one by one. Considering that no more than 7% error locating between continuous sections is acceptable in practice, the only gross error (less than 1%) can be easily eliminated by a smoothing filter on the predicted label distribution. This framework can also be extended to handle 3D cases. One way is to treat multiple slice as a multi-channel input (like the RGB image). Another way is using 3D convolutional filters in the CNN.

Another possible direction of improvement would be the multi-modal deep learning [55,56]. It is not limited in just multiple image modalities [57]. Since shape prior models [58–60] have shown the benefits in many applications, the 3D geometric information should be able to act as a complementary modality besides visual appearance to be incorporated in deep learning frameworks [61,62].

---

## REFERENCES

1. T. McInerney, D. Terzopoulos, Deformable models in medical image analysis: a survey, *Med. Image Anal.* 1 (2) (1996) 91–108.
2. J.A. Maintz, M.A. Viergever, A survey of medical image registration, *Med. Image Anal.* 2 (1) (1998) 1–36.
3. D.L. Pham, C. Xu, J.L. Prince, Current methods in medical image segmentation, *Annu. Rev. Biomed. Eng.* 2 (1) (2000) 315–337.
4. D.L. Hill, P.G. Batchelor, M. Holden, D.J. Hawkes, Medical image registration, *Phys. Med. Biol.* 46 (3) (2001) R1.

5. M. Betke, H. Hong, D. Thomas, C. Prince, J.P. Ko, Landmark detection in the chest and registration of lung surfaces with an application to nodule registration, *Med. Image Anal.* 7 (3) (2003) 265–281.
6. Y. Zheng, M. John, R. Liao, J. Boese, U. Kirschstein, B. Georgescu, S.K. Zhou, J. Kempfert, T. Walther, G. Brockmann, et al., Automatic aorta segmentation and valve landmark detection in C-arm CT: application to aortic valve implantation, in: *Medical Image Computing and Computer-Assisted Intervention*, Springer, 2010, pp. 476–483.
7. D. Shen, S. Moffat, S.M. Resnick, C. Davatzikos, Measuring size and shape of the hippocampus in MR images using a deformable shape model, *NeuroImage* 15 (2) (2002) 422–434.
8. H. Ling, S.K. Zhou, Y. Zheng, B. Georgescu, M. Suehling, D. Comaniciu, Hierarchical, learning-based automatic liver segmentation, in: *IEEE International Conference on Computer Vision and Pattern Recognition*, IEEE, 2008, pp. 1–8.
9. C. Li, R. Huang, Z. Ding, J.C. Gatenby, D.N. Metaxas, J.C. Gore, A level set method for image segmentation in the presence of intensity inhomogeneities with application to MRI, *IEEE Trans. Image Process.* 20 (7) (2011) 2007–2016.
10. R. Bellotti, F. De Carlo, G. Gargano, S. Tangaro, D. Cascio, E. Catanzariti, P. Cerello, S.C. Cheran, P. Delogu, I. De Mitri, et al., A CAD system for nodule detection in low-dose lung CTs based on region growing and a new active contour model, *Med. Phys.* 34 (12) (2007) 4901–4910.
11. L.A. Meinel, A.H. Stolpen, K.S. Berbaum, L.L. Fajardo, J.M. Reinhardt, Breast MRI lesion classification: improved performance of human readers with a backpropagation neural network computer-aided diagnosis (CAD) system, *J. Magn. Reson. Imaging* 25 (1) (2007) 89–95.
12. K. Doi, Current status and future potential of computer-aided diagnosis in medical imaging, *Br. J. Radiol.* 78 (suppl\_1) (2005) s1–s19.
13. Y. LeCun, Y. Bengio, G. Hinton, Deep learning, *Nature* 521 (7553) (2015) 436–444.
14. G. Hinton, L. Deng, D. Yu, G.E. Dahl, A.r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T.N. Sainath, et al., Deep neural networks for acoustic modeling in speech recognition: the shared views of four research groups, *IEEE Signal Process. Mag.* 29 (6) (2012) 82–97.
15. A. Krizhevsky, I. Sutskever, G.E. Hinton, ImageNet classification with deep convolutional neural networks, in: *Advances in Neural Information Processing Systems*, 2012, pp. 1097–1105.
16. I. Sutskever, O. Vinyals, Q.V. Le, Sequence to sequence learning with neural networks, in: *Advances in Neural Information Processing Systems*, 2014, pp. 3104–3112.
17. Y. Bengio, A. Courville, P. Vincent, Representation learning: a review and new perspectives, *IEEE Trans. Pattern Anal. Mach. Intell.* 35 (8) (2013) 1798–1828.
18. Z. Yan, Y. Zhan, Z. Peng, S. Liao, Y. Shinagawa, D.N. Metaxas, X.S. Zhou, Bodypart recognition using multi-stage deep learning, in: *International Conference on Information Processing in Medical Imaging*, Springer, 2015, pp. 449–461.
19. Z. Yan, Y. Zhan, Z. Peng, S. Liao, Y. Shinagawa, S. Zhang, D. Metaxas, X. Zhou, Multi-instance deep learning: discover discriminative local anatomies for bodypart recognition, *IEEE Trans. Med. Imaging* (2016) 1332–1343.
20. M.O. Gueld, M. Kohnen, D. Keysers, H. Schubert, B.B. Wein, J. Bredno, T.M. Lehmann, Quality of DICOM header information for image categorization, in: *Medical Imaging, International Society for Optics and Photonics*, 2002, pp. 280–287.

21. Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, *Proc. IEEE* 86 (11) (1998) 2278–2324.
22. C. Szegedy, A. Toshev, D. Erhan, Deep neural networks for object detection, in: *Advances in Neural Information Processing Systems*, 2013, pp. 2553–2561.
23. Y. Wei, W. Xia, J. Huang, B. Ni, J. Dong, Y. Zhao, S. Yan, CNN: single-label to multi-label, *arXiv:1406.5726*, 2014.
24. C. Cortes, V. Vapnik, Support-vector networks, *Mach. Learn.* 20 (3) (1995) 273–297.
25. D.W. Hosmer Jr., S. Lemeshow, *Applied Logistic Regression*, John Wiley & Sons, 2004.
26. D.G. Lowe, Distinctive image features from scale-invariant keypoints, *Int. J. Comput. Vis.* 60 (2) (2004) 91–110.
27. N. Dalal, B. Triggs, Histograms of oriented gradients for human detection, in: *IEEE International Conference on Computer Vision and Pattern Recognition*, vol. 1, IEEE, 2005, pp. 886–893.
28. D. Yang, S. Zhang, Z. Yan, C. Tan, K. Li, D. Metaxas, Automated anatomical landmark detection on distal femur surface using convolutional neural network, in: *IEEE International Symposium on Biomedical Imaging*, IEEE, 2015, pp. 17–21.
29. H.C. Shin, H.R. Roth, M. Gao, L. Lu, Z. Xu, I. Nogues, J. Yao, D. Mollura, R.M. Summers, Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics and transfer learning, *IEEE Trans. Med. Imaging* 35 (5) (2016) 1285–1298.
30. S. Albarqouni, C. Baur, F. Achilles, V. Belagiannis, S. Demirci, N. Navab, AggNet: deep learning from crowds for mitosis detection in breast cancer histology images, *IEEE Trans. Med. Imaging* 35 (5) (2016) 1313–1321.
31. Y. Guo, G. Wu, L.A. Commander, S. Szary, V. Jewells, W. Lin, D. Shen, Segmenting hippocampus from infant brains by sparse patch matching with deep-learned features, in: *Medical Image Computing and Computer-Assisted Intervention*, Springer, 2014, pp. 308–315.
32. O. Ronneberger, P. Fischer, T. Brox, U-Net: convolutional networks for biomedical image segmentation, in: *Medical Image Computing and Computer-Assisted Intervention*, Springer, 2015, pp. 234–241.
33. T. Brosch, L. Tang, Y. Yoo, D. Li, A. Traboulsee, R. Tam, Deep 3D convolutional encoder networks with shortcuts for multiscale feature integration applied to multiple sclerosis lesion segmentation, *IEEE Trans. Med. Imaging* 35 (5) (2016) 1229–1239.
34. P. Moeskops, M.A. Viergever, A.M. Mendrik, L.S. de Vries, M.J. Benders, I. Isgum, Automatic segmentation of MR brain images with a convolutional neural network, *IEEE Trans. Med. Imaging* 35 (5) (2016) 1252–1261.
35. M. Anthimopoulos, S. Christodoulidis, L. Ebner, A. Christe, S. Mougiakakou, Lung pattern classification for interstitial lung diseases using a deep convolutional neural network, *IEEE Trans. Med. Imaging* 35 (5) (2016) 1207–1216.
36. H.R. Roth, C.T. Lee, H.C. Shin, A. Seff, L. Kim, J. Yao, L. Lu, R.M. Summers, Anatomy-specific classification of medical images using deep convolutional nets, in: *IEEE International Symposium on Biomedical Imaging*, 2015, pp. 101–104.
37. Y. Taigman, M. Yang, M. Ranzato, L. Wolf, DeepFace: closing the gap to human-level performance in face verification, in: *IEEE International Conference on Computer Vision and Pattern Recognition*, IEEE, 2014, pp. 1701–1708.
38. M.M. Cheng, Z. Zhang, W.Y. Lin, P. Torr, BING: binarized normed gradients for objectness estimation at 300 fps, in: *IEEE International Conference on Computer Vision and Pattern Recognition*, IEEE, 2014, pp. 3286–3293.

39. R. Girshick, Fast R-CNN, in: IEEE International Conference on Computer Vision, 2015, pp. 1440–1448.
40. S. Ren, K. He, R. Girshick, J. Sun, Faster R-CNN: towards real-time object detection with region proposal networks, in: Advances in Neural Information Processing Systems, 2015, pp. 91–99.
41. Y. Zhan, X.S. Zhou, Z. Peng, A. Krishnan, Active scheduling of organ detection and segmentation in whole-body medical images, in: Medical Image Computing and Computer-Assisted Intervention, Springer, 2008, pp. 313–321.
42. A. Criminisi, J. Shotton, D. Robertson, E. Konukoglu, Regression forests for efficient anatomy detection and localization in CT studies, in: Medical Computer Vision. Recognition Techniques and Applications in Medical Imaging, Springer, 2011, pp. 106–117.
43. R. Donner, B.H. Menze, H. Bischof, G. Langs, Global localization of 3D anatomical structures by pre-filtered Hough Forests and discrete optimization, *Med. Image Anal.* 17 (8) (2013) 1304–1314.
44. P.O. Pinheiro, R. Collobert, From image-level to pixel-level labeling with convolutional networks, in: IEEE International Conference on Computer Vision and Pattern Recognition, 2015, pp. 1713–1721.
45. G. Papandreou, L.C. Chen, K. Murphy, A.L. Yuille, Weakly- and semi-supervised learning of a DCNN for semantic image segmentation, arXiv:1502.02734, 2015.
46. J. Wu, Y. Yu, C. Huang, K. Yu, Deep multiple instance learning for image classification and auto-annotation, in: IEEE International Conference on Computer Vision and Pattern Recognition, IEEE, 2015, pp. 3460–3469.
47. O. Maron, T. Lozano-Pérez, A framework for multiple-instance learning, in: Advances in Neural Information Processing Systems, 1998, pp. 570–576.
48. V. Nair, G.E. Hinton, Rectified linear units improve restricted Boltzmann machines, in: International Conference on Machine Learning, 2010, pp. 807–814.
49. N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: a simple way to prevent neural networks from overfitting, *J. Mach. Learn. Res.* 15 (1) (2014) 1929–1958.
50. N. Qian, On the momentum term in gradient descent learning algorithms, *Neural Netw.* 12 (1) (1999) 145–151.
51. Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, T. Darrell, Caffe: convolutional architecture for fast feature embedding, in: Proceedings of the ACM International Conference on Multimedia, ACM, 2014, pp. 675–678.
52. P. Sermanet, Y. LeCun, Traffic sign recognition with multi-scale convolutional networks, in: The 2011 International Joint Conference on Neural Networks (IJCNN), IEEE, 2011, pp. 2809–2813.
53. P. Felzenszwalb, D. McAllester, D. Ramanan, A discriminatively trained, multiscale, deformable part model, in: IEEE International Conference on Computer Vision and Pattern Recognition, IEEE, 2008, pp. 1–8.
54. R. Girshick, J. Donahue, T. Darrell, J. Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, in: IEEE International Conference on Computer Vision and Pattern Recognition, 2014, pp. 580–587.
55. J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, A.Y. Ng, Multimodal deep learning, in: International Conference on Machine Learning, 2011, pp. 689–696.
56. A. Karpathy, F.F. Li, Deep visual-semantic alignments for generating image descriptions, in: IEEE International Conference on Computer Vision and Pattern Recognition, 2015, pp. 3128–3137.

57. W. Zhang, R. Li, H. Deng, L. Wang, W. Lin, S. Ji, D. Shen, Deep convolutional neural networks for multi-modality isointense infant brain image segmentation, *NeuroImage* 108 (2015) 214–224.
58. T.F. Cootes, C.J. Taylor, D.H. Cooper, J. Graham, Active shape models – their training and application, *Comput. Vis. Image Underst.* 61 (1) (1995) 38–59.
59. S. Zhang, Y. Zhan, M. Dewan, J. Huang, D.N. Metaxas, X.S. Zhou, Towards robust and effective shape modeling: sparse shape composition, *Med. Image Anal.* 16 (1) (2012) 265–277.
60. S. Zhang, Y. Zhan, D.N. Metaxas, Deformable segmentation via sparse representation and dictionary learning, *Med. Image Anal.* 16 (7) (2012) 1385–1396.
61. Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, J. Xiao, 3D shapenets: a deep representation for volumetric shapes, in: *IEEE International Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1912–1920.
62. Y. Fang, J. Xie, G. Dai, M. Wang, F. Zhu, T. Xu, E. Wong, 3D deep shape descriptor, in: *IEEE International Conference on Computer Vision and Pattern Recognition*, 2015, pp. 2319–2328.

# Automatic Interpretation of Carotid Intima–Media Thickness Videos Using Convolutional Neural Networks

# 5

Nima Tajbakhsh\*, Jae Y. Shin\*, R. Todd Hurst<sup>†</sup>, Christopher B. Kendall<sup>†</sup>,  
Jianming Liang\*

*Arizona State University, Scottsdale, AZ, United States\* Mayo Clinic, Scottsdale, AZ,  
United States<sup>†</sup>*

## CHAPTER OUTLINE

<b>5.1</b>	<b>Introduction</b>	106
<b>5.2</b>	<b>Related Work</b>	107
<b>5.3</b>	<b>CIMT Protocol</b>	109
<b>5.4</b>	<b>Method</b>	109
5.4.1	Convolutional Neural Networks (CNNs)	109
5.4.2	Frame Selection	110
5.4.3	ROI Localization	112
5.4.4	Intima–Media Thickness Measurement	115
<b>5.5</b>	<b>Experiments</b>	117
5.5.1	Pre- and Post-Processing for Frame Selection	118
5.5.2	Constrained ROI Localization	118
5.5.3	Intima–Media Thickness Measurement	121
5.5.4	End-to-End CIMT Measurement	123
<b>5.6</b>	<b>Discussion</b>	124
<b>5.7</b>	<b>Conclusion</b>	128
<b>Acknowledgement</b>		128
<b>References</b>		128
<b>Notes</b>		131

## CHAPTER POINTS

- A framework based on CNNs that automates the entire CIMT interpretation process
- A novel method that selects the end-diastolic frames from an ultrasound video

- A new contextually-constrained method that localizes an ROI in an end-diastolic frame
- A framework that enables accurate interface segmentation in an ROI
- The suggested CNN-based system outperforms our previous handcrafted approach for CIMT interpretation

---

## 5.1 INTRODUCTION

Cardiovascular disease (CVD) is the number one killer in the United States [1]: every 40 seconds, one person in the United States dies of CVD [35]. More striking, nearly one-half of these deaths occur suddenly as the initial manifestation, and one-third of them occur in patients younger than 65 years [35,42]. Nevertheless, CVD is largely preventable [1]. However, the key is to identify at-risk persons before coronary events occur [42], so that scientifically proven and efficacious preventive care can be prescribed appropriately.

Carotid intima–media thickness (CIMT) is a noninvasive ultrasonography method that has proven to be valuable for predicting individual CVD risk [42]. It quantifies subclinical atherosclerosis, adds predictive value to traditional risk factors (e.g., the Framingham risk score [9]), and has several advantages over computed tomography (CT) coronary artery calcium score by being safer (no radiation exposure), more sensitive in a young population, and more accessible to the primary care setting. However, as illustrated in Fig. 5.1, interpretation of CIMT videos involves 3 manual operations: (i) selection of 3 end-diastolic ultrasonographic frames (EUFs) in each video; (ii) localization of a region of interest (ROI) in each selected EUF; and (iii) identification of the intima–media boundaries within each ROI to measure CIMT. These 3 operations, and in particular the CIMT measurement, are not only tedious and laborious, but also subjective to large interoperator variability [29] if guidelines are not properly followed. Therefore, many semi-automated computer-aided measurement systems have been suggested for CIMT (e.g., [26,7,37,47,16,3]). In this study, we present a *fully-automated* system to accelerate CIMT video interpretation while still offering a highly user-friendly interface that allows sonographers to offer their indispensable expertise.

The first appearance of CNNs dates back to 1989 when they were used for handwritten ZIP code recognition [21]. However, they had not demonstrated superior performance in image analysis until the ImageNet competition in 2012, a success that has triggered a revolution in computer vision through the efficient use of graphics processing units (GPUs) and other new technologies. This breakthrough widely inspired research on exploring the use of CNNs as a promising alternative to the existing handcrafted approaches. Recently, CNNs have been successfully used in medical image analysis, resulting in a new generation of computer-aided detection and diagnosis systems [5,41,8,46,43] with superior accuracy over the existing solutions. In this study, we show that with proper preprocessing and post-processing, our

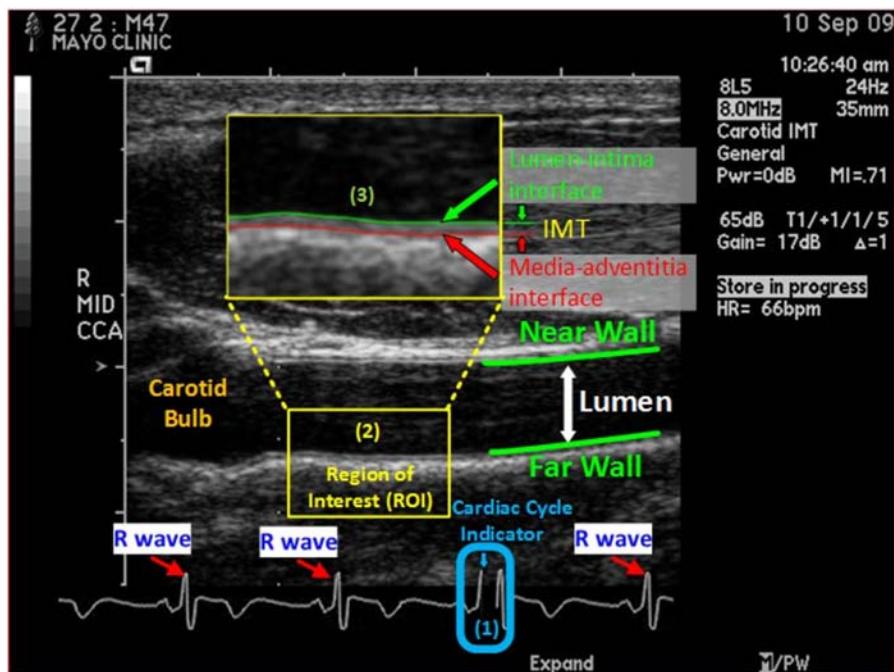
CNN-based approach can substantially outperform our previous carefully engineered method for CIMT image interpretation, making the following specific contributions:

- A framework based on CNNs that automates the entire CIMT interpretation process. This is in contrast to the prior works where only the very last step of the CIMT interpretation process was automated. The performance of the suggested system significantly outperforms our previous handcrafted approach, which, to our knowledge, is the only system in the literature that aimed to automate all the above three tasks.
- A novel frame selection method based on the electrocardiogram signals displayed at the bottom of ultrasound frames. The suggested method utilizes effective pre-processing of patches and post processing of CNN outputs, enabling a significant increase in the performance of a baseline CNN.
- A new method that localizes the ROI for CIMT interpretation. The suggested method combines the discriminative power of a CNN with a contextual constraint to accurately localize the ROIs in the selected frames. We demonstrate that the suggested contextually-constrained CNN outperforms the performance of a baseline CNN.
- A framework that combines CNNs with active contour models for accurate boundary segmentation. Specifically, given a localized ROI, the CNN initializes two open snakes, which further deform to acquire the shapes of intima–media boundaries. We show that the segmentation accuracy of the suggested method is greater than our previous handcrafted approach.

---

## 5.2 RELATED WORK

Given the clinical significance of CIMT as an early and reliable indicator of cardiovascular risk, several methods have been developed for CIMT image interpretation. The CIMT is defined as the distance between the lumen–intima and media–adventitia interfaces at the far wall of the carotid artery (Fig. 5.1). Therefore, to measure CIMT, the lumen–intima and media–adventitia interfaces must be identified. Hence, earlier approaches were focused on analyzing the intensity profile and distribution, computing the gradient [38,44,11], or combining various edge properties through dynamic programming [24,6,39]. The recent approaches [26,7,37,47,16,3] are mostly based on active contours (a.k.a., snakes) or their variations [18]. Some of these approaches require user interaction; other approaches aim for complete automation through integration of various image processing algorithms, such as Hough transform [34] and dynamic programming [39]. Most recently, a committee of standard multilayer perceptrons was used in [31], and a single standard, multilayer perceptron with an autoencoder was used in [30] for CIMT image interpretation, but both methods did not outperform a snake-based approach from the same research group [3,4]. A more complete survey of methods for automatic CIMT measurements can be found in the review studies conducted by Molinari et al. [32] and Loizou et al. [25].

**FIGURE 5.1**

Longitudinal view of the carotid artery in an ultrasonographic B-scan image. CIMT is defined as the distance between the lumen–intima interface and the media–adventitia interface, measured approximately 1 cm from the carotid bulb on the far wall of the common carotid artery at the end of the diastole. Therefore, interpretation of a CIMT video involves 3 operations: (i) selecting 3 EUFs in each video (the cardiac cycle indicator shows to where in the cardiac cycle the current frame corresponds); (ii) localizing an ROI approximately 1 cm distal from the carotid bulb in the selected EUF; and (iii) measuring the CIMT within the localized ROI. In this paper, we aim to automate these 3 operations simultaneously through the use of CNNs.

However, nearly all the methods explained above are focused on only the third operation, CIMT measurement, and ignore the 2 preceding operations of frame selection and ROI localization. To our knowledge, the only system that simultaneously automates the 3 operations is our prior work [40], an extension of [48], which automatically selects the EUF, localizes the ROI in each selected EUF, and provides the CIMT measurement in the selected ROI. However, as with other works, this method is based on handcrafted algorithms, which often lack the desired robustness for routine clinical use [40]. Our new proposed method aims to overcome this weakness through CNNs. As demonstrated in Sections 5.5 and 5.6, our new system outperforms our previous handcrafted method in all aspects, including frame selection, ROI localization, and CIMT measurements.

---

### 5.3 CIMT PROTOCOL

Each of the CIMT examinations was performed with high-resolution B-mode ultrasonography using a 15 MHz linear array transducer with fundamental frequency only (Acuson Sequoia, Mountain View, CA, USA). The carotid screening protocol begins with scanning up from the lower neck in the transverse manner to the carotid artery and then further to the carotid bulb and to internal and external carotid arteries. The probe is then turned to obtain the longitudinal view of the common carotid artery. The sonographer optimizes the 2-dimensional images of the lumen–intima and media–adventitia interfaces at the level of the common carotid artery by adjusting overall gain, time gain, compensation, and focus position. After the parameters are adjusted, the sonographer captures 2 CIMT videos focused on the common carotid artery from 2 optimal angles of incidence. The same procedure is repeated for the other side of neck, resulting in a total of 4 CIMT videos for each patient. The videos are recorded at 24 frames/second and consist of  $640 \times 480$  ultrasound images. The pixel spacing is 0.09 mm/pixel along both  $x$  and  $y$  directions.

---

### 5.4 METHOD

We propose a solution based on CNNs for automating all 3 tasks in CIMT interpretation. Herein, we first briefly explain CNNs and then detail each stage of the suggested system.

#### 5.4.1 CONVOLUTIONAL NEURAL NETWORKS (CNNS)

CNNs are deep learning machines that were originally proposed by LeCun in 1989 for handwritten ZIP code recognition [21]. Although CNNs were a breakthrough, they did not achieve much popularity at the time, mainly because of slow central processing units (CPUs) and limited memory resources. However, the advent of powerful GPUs in scientific computing, together with effective regularization techniques [12, 13, 19, 45, 22], has once again revived the CNNs, allowing researchers to break records in many computer vision and medical image analysis challenges. The main power of CNNs stems from a deep learning architecture that allows for learning a hierarchical set of features. One major advantage of CNNs is that they are end-to-end learning machines where the input images are directly mapped to the target labels or target bounding box coordinates. This direct mapping eliminates the need for designing suboptimal handcrafted features, which often provide a noisy image representation with insufficient discriminative power.

CNNs are an extension of multilayer perceptrons, in which fully connected hidden layers are replaced with convolutional layers. In a convolutional layer, each unit is connected to only a small subset of spatially connected units in the previous layer and the weights of such connections are shared among all units in the convolution layer. Weight sharing dramatically decreases the network's width (i.e., number of

parameters in each layer), enabling the design of deeper architectures. To reduce computational complexity and achieve a hierarchical image representation, each sequence of convolution layers is followed by a pooling layer, a workflow reminiscent of simple and complex cells in the primary visual cortex [14]. The pooling layer subsamples the feature maps of the preceding convolutional layer by computing the average response or selecting the maximum response in small overlapping or non-overlapping neighborhoods. A CNN typically consists of several pairs of convolutional and pooling layers, followed by a number of consecutive  $1 \times 1$  convolutional layers (a.k.a., fully connected layers), and finally a softmax layer or a regression layer to generate the desired outputs.

If  $D$  denotes a set of training images,  $W$  denotes a matrix containing the weights of the convolutional layers, and  $f_W(D^{(i)})$  denotes the loss for the  $i$ th training image, the loss over the entire training set is computed with the following equation:

$$\mathcal{L}(W) = \frac{1}{|D|} \sum_i^{|D|} f_W(X^{(i)}). \quad (5.1)$$

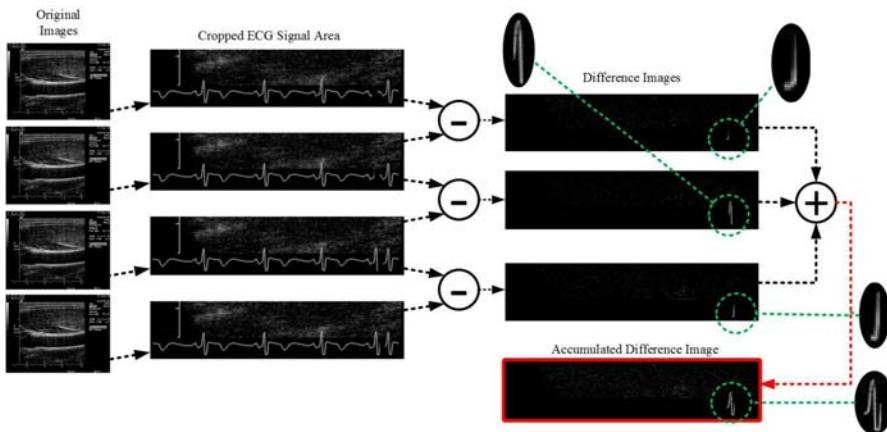
Gradient descent [2] is commonly used for minimizing the above loss function with respect to the unknown weights  $W$ . However, the modern, massively parallelized implementations of CNNs are limited by the amount of memory on GPUs; therefore, the loss function cannot be evaluated on the basis of the entire training set  $D$  at once. Instead, the loss function is approximated with the loss over the mini-batches of training images of size  $N \ll |D|$ . A common choice of the mini-batch size is 128, which is a reasonable trade-off between loss noise suppression and memory management. Given the size of mini-batches, the loss function can be approximated as  $\mathcal{L}(W) \approx \frac{1}{N} \sum_{i=1}^N f_W(X^{(i)})$  and the weights of the network iteratively updated with the following equations:

$$\begin{aligned} \gamma_t &= \gamma^{\lfloor \frac{tN}{|D|} \rfloor}, \\ V_{t+1} &= \mu V_t - \gamma_t \alpha \Delta L(W_t), \\ W_{t+1} &= W_t + V_{t+1} \end{aligned} \quad (5.2)$$

where  $\alpha$  is the learning rate,  $\mu$  is the momentum that indicates the contribution of the previous weight update in the current iteration, and  $\gamma$  is the scheduling rate that decreases the learning rate  $\alpha$  at the end of each epoch.

### 5.4.2 FRAME SELECTION

For a CIMT video, the first step in cardiovascular risk assessment is to select 3 EUFs. The CIMT test is routinely performed with electrocardiogram (ECG) test, and the operator normally selects these frames on the basis of the ECG signal that is printed at the bottom of ultrasonographic frames. Each frame in the CIMT video corresponds to a particular location in the printed ECG signal. To establish this correspondence, as

**FIGURE 5.2**

An accumulated difference image is generated by adding up 3 neighboring difference images.

shown in Fig. 5.1, a black line indicator is displayed on the ECG signal, indicating to where in the cardiac cycle the current frame corresponds. In routine clinical practice, the operator selects the EUFs so that the corresponding black line indicator coincides with the R wave in the QRS complex of the printed ECG signals.

We aim to automate the frame selection process by automatically identifying the frames that correspond to the R wave in the QRS complex of a printed ECG signal. Our idea is to first reconstruct the segment of the ECG signal that is masked by a black line indicator in the current frame and then determine whether the restored wavelet<sup>1</sup> resembles the appearance of an R wave or not. For this purpose, we introduce the notion of accumulated difference images that allows us to capture the missing wavelets and then use a CNN to classify these captured wavelets into R wave or non-R wave categories.

Let  $I^t$  denote an image subregion selected from the bottom 20% of an ultrasonicographic frame so that it contains the displayed ECG signal. We first construct a set of difference images  $d^t$  by subtracting every consecutive pair of images,  $d^t = |I^t - I^{t+1}|$ , and then form accumulated difference images by adding up every 3 neighboring difference images,  $D^t = \sum_{i=0}^2 d^{t-i}$ . Accumulated difference image  $D^t$  can capture the segment of the ECG signal that is masked by the black line indicator at frame  $t$ . Fig. 5.2 illustrates how an accumulated difference image is generated.

Second, we determine the location of the restored wavelet in each accumulated difference image. For this purpose, we find the weighted centroid  $c = [c_x, c_y]$  of each accumulated difference image  $D^t$  as follows:

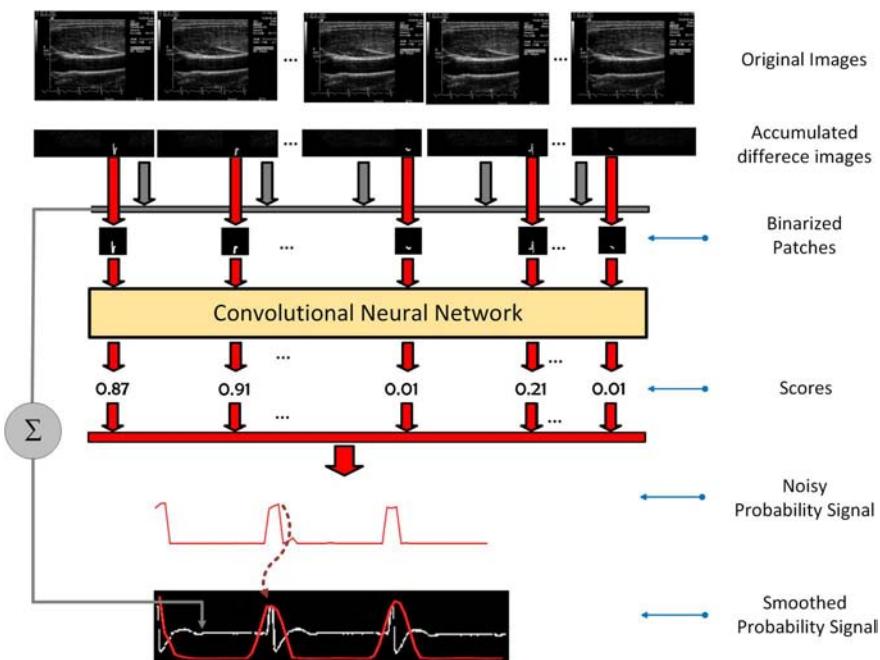
$$c = \frac{1}{Z_t} \sum_{p \in D^t} D^t(p_x, p_y) \times p$$

where  $p = [p_x, p_y]$  is a pixel in the accumulated difference image and  $Z_t = \sum_{p \in D^t} D^t(p_x, p_y)$  is a normalization factor that ensures the weighted centroid stays within the image boundary. After centroids are identified, we extract patches of size  $32 \times 32$  around the centroid locations. Specifically, we extract patches with up to 2 pixel translations from each centroid. However, we do not perform data augmentation by scaling the patches because doing so would inject label noise in the training set. For instance, a small restored wavelet may take the appearance of an R wave after expanding or an R wave may look like a non-R wave after shrinking. We also do not perform rotation-based patch augmentation because we do not expect the restored wavelets to appear with rotation in the test image patches. After collection, the patches are binarized with Otsu's method [36]. In Section 5.5.1, we discuss the choice of binarization method through an extensive set of experiments. Each binary patch is then labeled as positive if it corresponds to an EUF (or an R wave); otherwise, it is labeled as negative. Basically, given a patch, we initially determine the accumulated difference image from which the patch is extracted. We then trace back to the underlying difference images and check whether they are related to the EUF or not. After the patches are labeled, we form a stratified set with 96,000 patches to train a CNN for distinguishing between R waves and non-R waves.

[Fig. 5.3](#) shows our frame selection system for a test video. We compute an accumulated difference image for each frame in the video. We then extract image patches from the weighted centroids of the accumulated difference images. The probability of each frame being the EUF is measured as the average probabilities assigned by the CNN to the corresponding patches. By concatenating the resulting probabilities for all frames in the video, we obtain a probability signal whose local maxima indicate the locations of the EUFs. However, the generated probability signals often exhibit abrupt changes, which can cause too many local maxima along the signal. We therefore first smooth the probability signal using a Gaussian function and then find the EUFs by locating the local maxima of the smoothed signals. In [Fig. 5.3](#), for illustration purposes, we also show the reconstructed ECG signal, which is computed as the average of the accumulated difference images,  $\frac{1}{N} \sum_{t=1}^N D^t$  with  $N$  being the number of frames in the video. As seen, the probability of a frame being the EUF reaches a maximum around the R waves of the QRS complexes (as desired) and then smoothly decays as it distances from the R waves. By mapping the locations of the local maxima to the frame numbers, we can identify the EUFs in the test video.

### 5.4.3 ROI LOCALIZATION

Accurate localization of the ROI is challenging because, as seen in [Fig. 5.1](#), no notable differences can be observed in image appearance among the ROIs on the far wall of the carotid artery. To overcome this challenge, we use the location of the carotid bulb as a contextual constraint. We choose this constraint for 2 reasons: (i) the carotid bulb appears as a distinct dark area in the ultrasonographic frame and thus can be uniquely identified; and (ii) according to the consensus statement of American Society of Electrocardiography for cardiovascular risk assessment [42], the ROI should

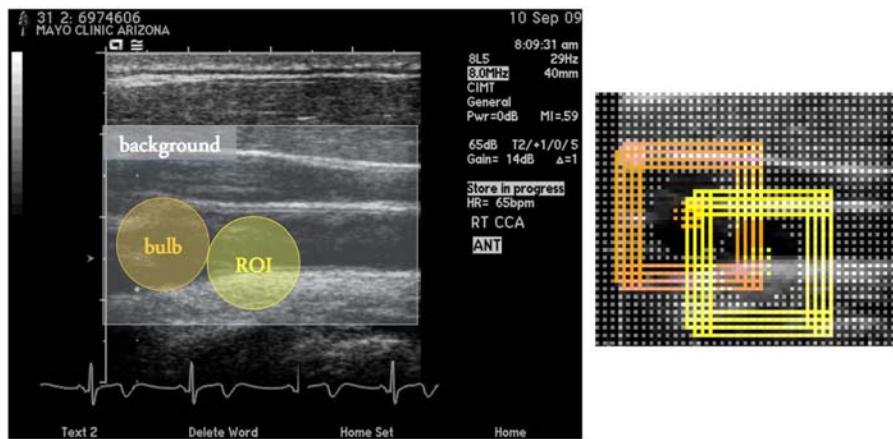
**FIGURE 5.3**

The test stage of our automatic frame selection scheme.

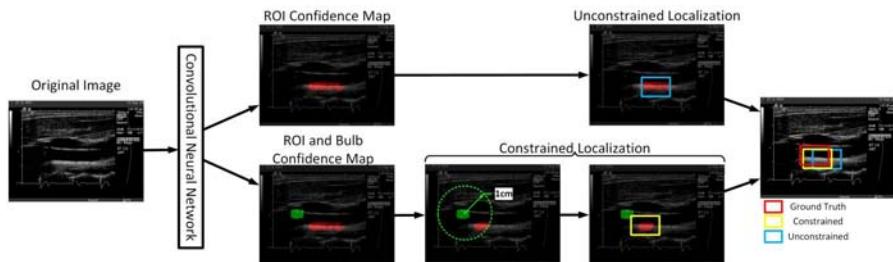
be placed approximately 1 cm from the carotid bulb on the far wall of the common carotid artery. The former motivates the use of the carotid bulb location as a constraint from a technical point of view, and the latter justifies this constraint from a clinical standpoint.

We incorporate this constraint in the suggested system by training a CNN for 3-class classification that simultaneously localizes both ROI and carotid bulb and then refines the estimated location of the ROI given the location of the carotid bulb. Fig. 5.4 shows how the image patches are extracted from a training frame. We perform data augmentation by extracting the training patches within a circle around the locations of the carotid bulbs and the ROIs. The background patches are extracted from a grid of points sufficiently far from the locations of the carotid bulbs and the ROIs. Of note, the described translation-based data augmentation is sufficient for this application because our database provides a relatively large number of training EUFs, from which a large set of training patches can be collected. After the patches are collected, we form a stratified training set with approximately 410,000 patches to train a CNN for constrained ROI localization.

In referring to Fig. 5.5, the trained CNN is applied during the test stage to all the pixels in the EUF, generating 2 confidence maps with the same size as the EUF. The

**FIGURE 5.4**

For constrained ROI localization, we use a CNN for 3-class classification whose training image patches are extracted from a grid of points on the background and around the ROI and the carotid bulb locations.

**FIGURE 5.5**

The test stage of our ROI localization method. In the unconstrained scenario, we use only the ROI confidence map, which results in a relatively large localization error. In the constrained mode, given the estimated location of the carotid bulb, we localize the ROI more accurately.

first confidence map shows the probability of a pixel being the carotid bulb, and the second confidence map shows the probability of a pixel being the ROI. One way to localize the ROI is to find the center of the largest connected component within the ROI confidence map without considering the detected location of the carotid bulb. However, this naive approach may fail to accurately localize the ROI. For instance, a long-tail connected component along the far wall of the carotid artery may cause substantial ROI localization error. To compound the problem, the largest connected component of the ROI confidence map may appear far from the actual location of the ROI, resulting in a complete detection failure. To overcome these limitations,

we constrain the ROI location  $l_{roi}$  by the location of the carotid bulb  $l_{cb}$ . For this purpose, we determine the location of the carotid bulb as the centroid of the largest connected component within the first confidence map and then localize the ROI using the following formula:

$$l_{roi} = \frac{\sum_{p \in C^*} M(p) \cdot p \cdot I(p)}{\sum_{p \in C^*} M(p) \cdot I(p)} \quad (5.3)$$

where  $M$  denotes the confidence map of being the ROI,  $C^*$  is the largest connected component in  $M$  that is nearest to the carotid bulb, and  $I(p)$  is an indicator function for pixel  $p = [p_x, p_y]$  that is defined as

$$I(p) = \begin{cases} 1, & \text{if } \|p - l_{cb}\| < 1 \text{ cm}, \\ 0, & \text{otherwise.} \end{cases}$$

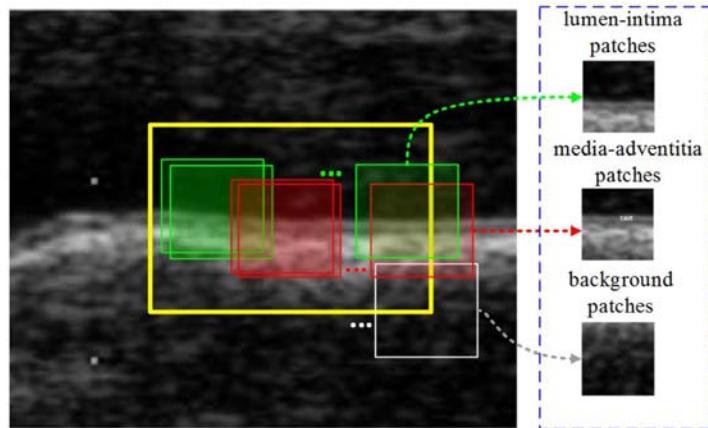
Basically, the indicator function excludes the pixels located farther than 1 cm from the carotid bulb location. This choice of the distance threshold is motivated by the fact that the ROI is located within 1 cm to the right of the carotid bulb. [Fig. 5.5](#) illustrates how the location of the carotid bulb as a contextual clue improves the accuracy of ROI localization.

#### 5.4.4 INTIMA–MEDIA THICKNESS MEASUREMENT

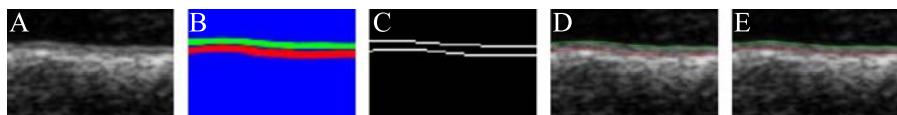
To automatically measure intima–media thickness, the lumen–intima and media–adventitia interfaces of the carotid artery need to be detected within the ROI. Although the lumen–intima interface is relatively easy to detect, the detection of the media–adventitia interface is challenging because of the faint image gradients around its boundary. We formulate this interface segmentation problem as a 3-class classification task where the goal is to classify each pixel within the ROI into 3 categories: (i) a pixel on the lumen–intima interface, (ii) a pixel on the media–adventitia interface, and (iii) a background pixel.

We use a 3-way CNN to segment the lumen–intima and media–adventitia interfaces. To train the CNN, we collect image patches from the lumen–intima interface and media–adventitia interface, as well as from other random locations far from the desired interfaces. [Fig. 5.6](#) shows how the training patches are collected from 1 ROI. For the positive patches, we choose not to perform data augmentation. This decision is made because  $92 \times 60$  ROIs allow us to collect a large number of patches around the lumen–intima and media–adventitia interfaces. Furthermore, given the relatively small distance between the 2 interfaces, translation-based data augmentation would inject a large amount of label noise in the training set, which would negatively impact the convergence and the overall performance of the CNN. When the patches are collected, we form a stratified training set with approximately 380,000 patches to train a 3-way CNN for interface segmentation.

[Fig. 5.7](#) illustrates how our system measures intima–media thickness in a test ROI. The trained CNN is applied to a given test ROI in a convolutional manner, generating

**FIGURE 5.6**

For lumen–intima and media–adventitia interface segmentation, we use a CNN whose training image patches are extracted from the background and around the lumen–intima and media–adventitia interfaces.

**FIGURE 5.7**

The test stage of lumen–intima and media–adventitia interface detection. (A) A test region of interest. (B) The trained CNN generates a confidence map where the green and red colors indicate the likelihood of lumen–intima interface and media–adventitia interface, respectively. (C) The thick probability band around each interface is thinned by selecting the largest probability for each interface in each column. (D) The step-like boundaries are refined through 2 open snakes. (E) The ground truth made as the consensus of two experts.

2 confidence maps with the same size as the ROI. The first confidence map shows the probability of a pixel being on the lumen–intima interface; the second confidence map shows the probability of a pixel being on the media–adventitia interface. We have shown the 2 confidence maps in Fig. 5.7 where the green and red colors indicate the likelihood of being the lumen–intima interface and the media–adventitia interface, respectively. A relatively thick high-probability band is apparent along each interface, which hinders the accurate measurement of intima–media thickness. To thin the detected interfaces, we scan the confidence map column by column, searching for the rows with the maximum response for each of the 2 interfaces. By doing so, we obtain a 1-pixel-thick boundary with a step-like shape around each interface as shown in Fig. 5.7C. To further refine the boundaries, we use 2 active contour models (a.k.a.,

**Table 5.1** The CNN architecture used in our experiments. Of note,  $C$  is the number of classes, which is 2 for frame selection and 3 for both ROI localization and intima–media thickness measurements

Layer	Type	Input	Kernel	Stride	Pad	Output
0	input	$32 \times 32$	N/A	N/A	N/A	$32 \times 32$
1	convolution	$32 \times 32$	$5 \times 5$	1	0	$64 \times 28 \times 28$
2	max pooling	$64 \times 28 \times 28$	$3 \times 3$	2	0	$64 \times 14 \times 14$
1	convolution	$64 \times 14 \times 14$	$5 \times 5$	1	0	$64 \times 10 \times 10$
2	max pooling	$64 \times 10 \times 10$	$3 \times 3$	2	0	$64 \times 5 \times 5$
2	fully connected	$64 \times 5 \times 5$	$5 \times 5$	1	0	$250 \times 1$
2	fully connected	$250 \times 1$	$1 \times 1$	1	0	$C \times 1$

snakes) [23], one for the lumen–intima interface and one for the media–adventitia interface. The open snakes are initialized with the current step-like boundaries and then deform solely based on the probability maps generated by the CNN rather than the original image content. Fig. 5.7D shows the converged snakes for the test ROI. We determine intima–media thickness as the average of vertical distance between the 2 open snakes.

## 5.5 EXPERIMENTS

We use UFL MCAEL CIMT research database [15] with an average population age of 27.0 and standard deviation of 2.8 years. We select 23 patients from this database using systematic random sampling, resulting in a total of 92 CIMT videos (4 videos per patient). The number of frames in each video ranges between 49 to 119. Each video covers at least 3 cardiac cycles and thus a minimum of 3 EUFs. The ground truth for each video consists of the locations of EUFs, the locations of ROIs, and the segmentation of lumen–intima and media–adventitia interfaces. The ground truth is created by a sonographer using a spline-based graphical user interface system and is then refined as the consensus of the first sonographer and a new expert. For consistency among the 3 detection tasks, we use the same training set for tuning the parameters and the same test set (no overlap with training) for evaluating the detection accuracy of each task. For this purpose, we randomly divide the CIMT videos at the patient level into training and test sets. The training set contains 48 CIMT videos of 12 patients with a total of 4456 frames; the test set contains 44 CIMT videos of 11 patients with a total of 3565 frames. For each task, we perform leave-1-patient-out cross-validation on the basis of the *training patients* to tune the parameters and then we evaluate the performance of the tuned system using the test patients.

For all 3 detection tasks, we use a widened version of LeNet [20], that is, we have used the same number of layers but employed a larger number of kernels in the convolutional layers. This is because wider CNNs tend to perform better than narrower CNNs [10]. As summarized in Table 5.1, the selected architecture has 2 convolutional

layers with rectified linear unit activation functions, 2 subsampling layers, and 2 fully connected layers. We also append a softmax layer to the last fully connected layer to generate probabilistic confidence values for each class. Since the selected architecture is designed for input patches of size  $32 \times 32$ , we resize the collected patches to  $32 \times 32$  before the training process. For the CNNs used in our experiments, we use a learning rate of  $\alpha = 0.001$ , a momentum of  $\mu = 0.9$ , and a constant scheduling rate of  $\gamma = 0.95$ . For our experiments, we also use an open source GPU implementation of CNNs.<sup>2</sup>

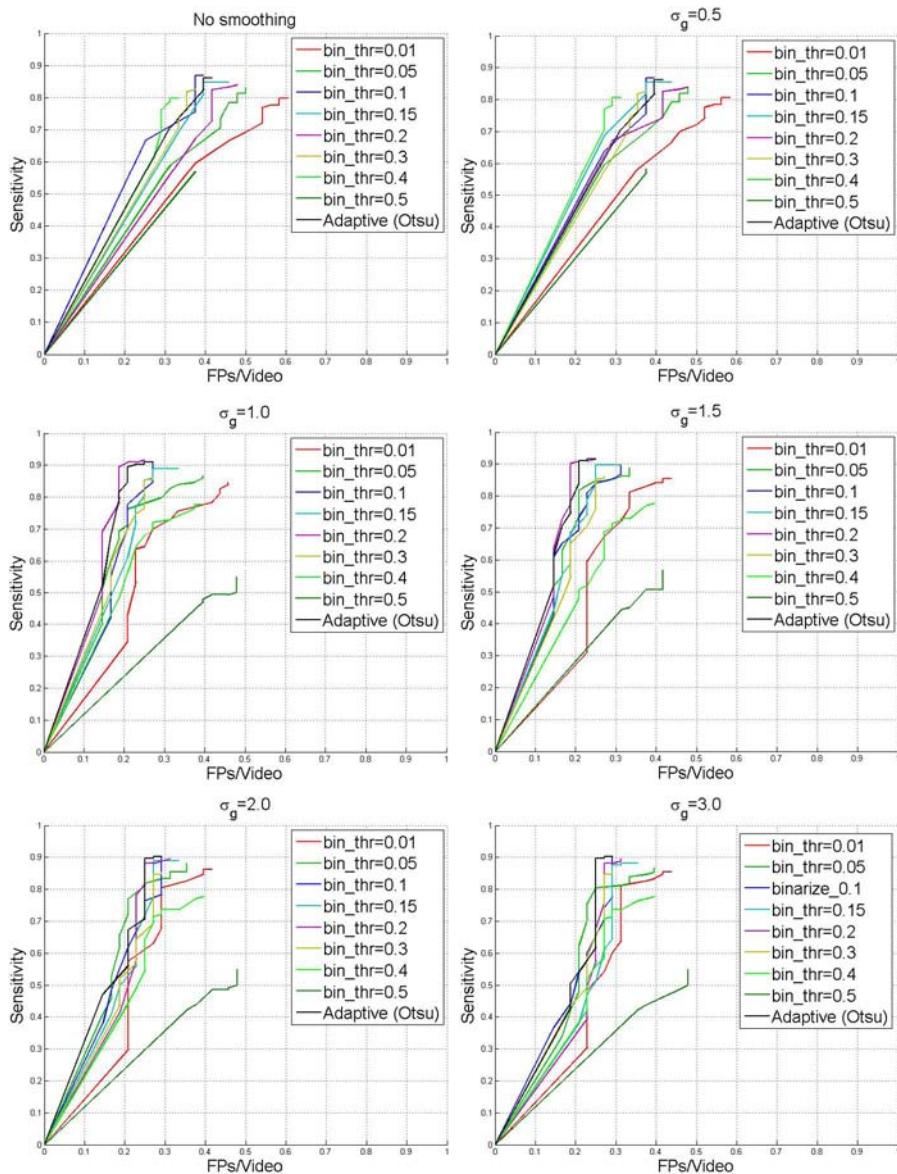
### 5.5.1 PRE- AND POST-PROCESSING FOR FRAME SELECTION

We have experimentally determined that binarized image patches improve the quality of convergence and the accuracy of frame selection. Furthermore, we have observed that the standard deviation of the Gaussian function used for smoothing the probability signals can also substantially influence frame selection accuracy. Therefore, we have conducted leave-1-patient-out cross-validation based on the training patients to find the best binarization method and the optimal standard deviation of the Gaussian function. For binarization, we have considered a fixed set of thresholds and an adaptive thresholding scheme with Otsu's method. For smoothing, we have considered a Gaussian function with different standard deviation ( $\sigma_g$ ), as well as the scenario where no smoothing is applied. For each configuration of parameters, we have done a free-response receiver operating characteristic (FROC) analysis. We consider a selected frame a true positive when it is found within 1 frame from the expert-annotated EUF; otherwise, it is considered a false positive.

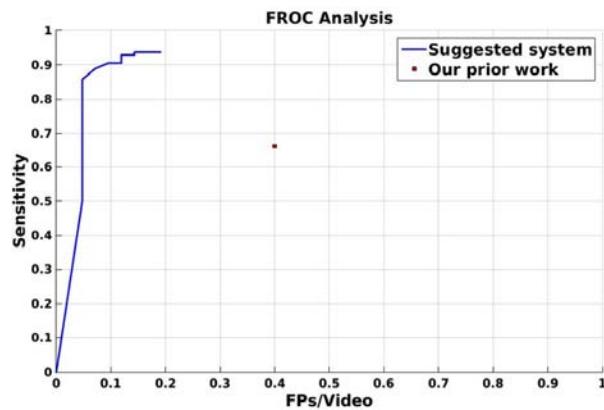
[Fig. 5.8](#) shows the FROC curves for the training patients using leave-1-patient-out cross-validation. As seen, no smoothing or a small degree of Gaussian smoothing leads to relatively low frame selection accuracy. This is because a trivial level of smoothing may not properly handle the fluctuations in the probability signals, causing a large number of false positives around an EUF. Yet, a large degree of smoothing may decrease the sensitivity of frame selection because the locations of the local maxima may be found more than 1 frame away from the expert-annotated EUFs. We therefore use a Gaussian function with  $\sigma_g = 1.5$  for smoothing the probability signals. Our results also indicate that the adaptive thresholding method and a fixed threshold of 0.2 achieve the highest frame selection accuracy. However, we choose to use adaptive thresholding because it decreases the parameters of our system by one and that it performs more consistently at different levels of Gaussian smoothing. [Fig. 5.9](#) shows the FROC curve of our system for the test patients, using the tuned parameters. For comparison, we also show the operating point of our prior work [40], which is outperformed by the suggested system.

### 5.5.2 CONSTRAINED ROI LOCALIZATION

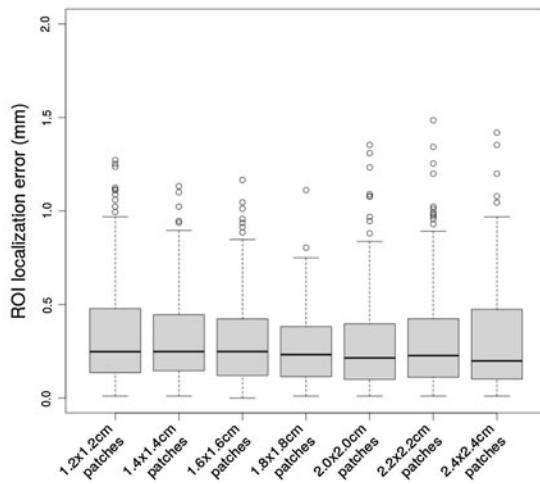
We conduct a leave-1-patient-out cross-validation study based on the training patients to find the optimal size of the training patches. [Fig. 5.10](#) shows the box plots of

**FIGURE 5.8**

FROC curves of our system for automatic frame selection. Each plot shows FROC curves for different binarization thresholds and different levels of Gaussian smoothing. The results are obtained using leave-1-patient-out cross-validation based on the training patients.

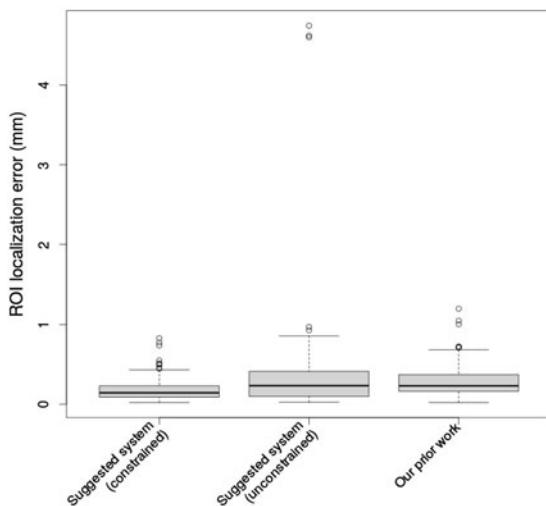
**FIGURE 5.9**

FROC curve of our frame selection system for the test patients with use of tuned parameters. For comparison, we also show the operating point of our prior work [40], which is outperformed by the suggested system.

**FIGURE 5.10**

ROI localization error of our system for different sizes of patches. The results are obtained using leave-1-patient-out cross-validation based on the training patients.

ROI localization error for different sizes of patches. In our analyses, we measure the localization error as the Euclidean distance between the estimated ROI location and the one provided by the expert. According to our cross-validation results, the use of  $1.8 \times 1.8$  cm patches achieves the most stable performance, yielding low ROI localization error with only a few outliers.

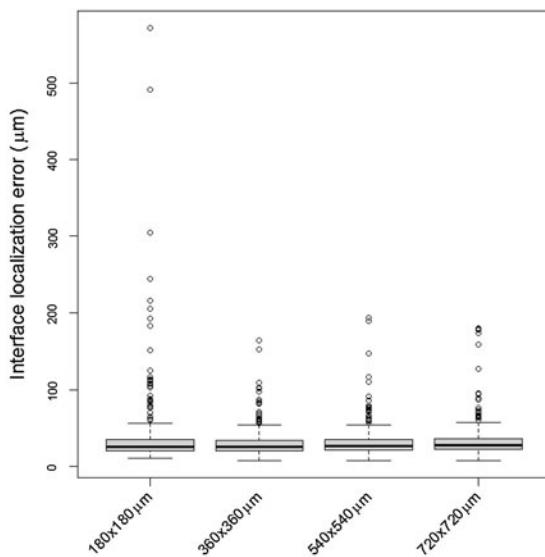
**FIGURE 5.11**

ROI localization error for the test patients. Our method in the constrained mode outperforms both the unconstrained counterpart and our prior work [40].

[Fig. 5.11](#) shows the ROI localization error of our system for the test patients, using the optimal size of training patches. To demonstrate the effectiveness of our constrained ROI localization method, we include the performance of the unconstrained counterpart. In the constrained mode, we use Eq. (5.3) for ROI localization; in the unconstrained mode, we localize the ROI as the center of the largest connected component in the corresponding confidence map without considering the location of the carotid bulb. Our method achieves an average localization error of 0.19 and 0.35 mm in the constrained and unconstrained modes, respectively. The decrease in localization error is statistically significant ( $p < 0.01$ ). Furthermore, as shown in [Fig. 5.11](#), our method in the unconstrained mode has resulted in 3 complete localization failures (outliers), which have been corrected in the constrained mode. Compared with our prior work [40], our system in the constrained mode shows a decrease of 0.1 mm in ROI localization error, which is statistically significant ( $p < 0.00001$ ).

### 5.5.3 INTIMA–MEDIA THICKNESS MEASUREMENT

We conducted a leave-1-patient-out cross-validation study based on the training patients to find the optimal size of the training patches. The results are depicted in [Fig. 5.12](#), where each box plot shows the combined localization error of lumen–intima and media–adventitia interfaces for a different size of patches. In our analyses, we determine the localization error as the average of absolute vertical distances between our detected boundaries and the expert-annotated boundaries for the interfaces. Although our system shows a high degree of robustness against different sizes of

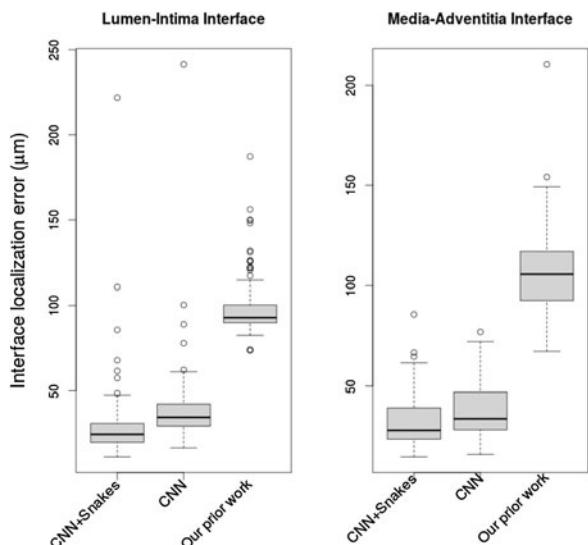
**FIGURE 5.12**

Combined interface localization error for different sizes of patches. The results are produced through a leave-1-patient-out cross-validation study based on the training patients.

input patches, the use of patches of size  $360 \times 360 \mu\text{m}$  achieves slightly lower localization error and fewer outliers. Furthermore, this choice of patches yields greater computational efficiency than the larger counterpart patches.

[Fig. 5.13](#) shows the interface localization error of our system with and without snakes on the test patients. For a more accurate analysis, we break down the overall localization error into the localization error of lumen–intima interface and the localization error of the media–adventitia interface. We have also included the localization error of our prior work [40] for each of the 2 interfaces. As shown, our system yields a median error of  $2.68 \mu\text{m}$  for the lumen–intima interface and a median error of  $3.49 \mu\text{m}$  for the media–adventitia interface, which is significantly lower than a CNN without snakes and our prior work (paired t-test,  $p < 0.0001$ ). The higher error for the media–adventitia interface is due to lower contrast and faint gradients around its boundary.

We further analyzed agreement between our system and the expert for the assessment of intima–media thickness. To this end, we use the Bland–Altman plot, which is a well-established technique to measure agreement among different observers. The Bland–Altman plot for the test patients is shown in [Fig. 5.14](#), where each circle represents a pair of thickness measurements, one from our method and one from the expert. The majority of circles are within 2 standard deviations from the mean error, which suggests a large agreement between the automatically computed thickness measurements and those of the expert. Furthermore, Pearson product–moment corre-



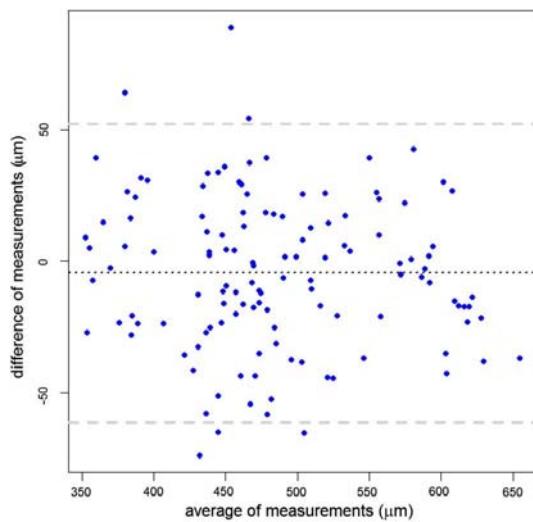
**FIGURE 5.13**

Localization error of the lumen–intima and media–adventitia interfaces for the suggested system (CNN and CNN+Snakes) and our prior work [40]. The results are obtained for the test patients.

lation coefficient for the average and difference measurements is  $-0.097$ , indicating that the agreement between our method and the expert does not depend on intima–media thickness.

#### 5.5.4 END-TO-END CIMT MEASUREMENT

In the previous analyses, we evaluated each stage of the suggested system independently. Herein, we analyze the performance of our system as a whole. Specifically, the input to our system is an ultrasound video and the output is the measured CIMT as the average of measurements made in the EUFs of the input video. This analysis mimics the actual clinical scenario, permitting us to investigate how the error of each stage of the suggested system accumulates through the pipeline. For this analysis, we asked 3 new experts to measure CIMT for the 44 videos from the 11 test patients. The difference in CIMT measurements between readers 1 and 2 was  $16 \pm 12 \mu\text{m}$ , between readers 1 and 3 was  $22 \pm 15 \mu\text{m}$ , and between reader 2 and 3 was  $28 \pm 18 \mu\text{m}$ . We obtained an interclass correlation coefficient of  $0.986$  (95% CI  $0.977$ – $0.991$ ), suggesting an excellent level of agreement among the 3 experts. The ANOVA test also yielded  $p$ -values around  $0.50$ , suggesting that there is a lack of evidence to show a difference among the CIMT measurements made by our system and those made by the 3 experts. Our results suggest that the suggested system is robust against variability in CIMT measurements made by different experts.

**FIGURE 5.14**

The Bland–Altman plot shows high agreement between our system and the expert for the assessment of intima–media thickness. Each circle in the plot represents a pair of thickness measurements from our method and the expert for a test region of interest. The plot shows a total of 126 circles corresponding to 44 test videos.

## 5.6 DISCUSSION

Our computer-aided measurement system consisted of 3 components, each designed according to the guidelines suggested for a quality CIMT exam [42]. Specifically, we used the ECG signal for EUF selection because the guidelines for the CIMT exams recommend the use of echocardiography signals for accurate and consistent selection of EUFs. We utilized the location of carotid bulb for more accurate ROI localization, because the guidelines recommend the measurement of CIMT approximately 1 cm from the carotid bulb. Finally, our method for CIMT measurement implicitly assumed the horizontal appearance of the common carotid artery in the ultrasound images, which was in accordance with the guidelines for a quality CIMT exam. However, our method could also accommodate a range of incidence angles by rotating the ultrasound images according to the orientation of the common carotid artery.

Throughout the experiment section, we demonstrated that our CNN-based system outperformed our previous handcrafted approach. Here, we discuss the reasons behind the observed superiority. Our previous approach relies on altitude of the ECG signal to find the R waves of the QRS complexes, but the CNN-based approach explicitly learns the appearance of the QRS complex. The former is affected by the variability of peak altitude across different vendors but the latter is robust against such variations in the displayed ECG signals. Our previous approach identifies the

bulb area (ROI) by computing curvature along the boundary of the carotid artery. However, curvature calculation requires a clean segmentation of the carotid artery, which is not a trivial task in low-contrast ultrasound images. On the other hand, the CNN-based approach explicitly learns the appearance of the ROI, eliminating the need for artery segmentation. For CIMT measurements, our previous method initializes the open snakes based on image gradient, which is often affected by spurious edges and low contrast areas in the ultrasound frames. However, our CNN-based approach initializes the snakes closer to the desired boundaries by recognizing the upper and lower interfaces of the carotid arteries.

In Section 5.5.1, we investigated how the choice of patch binarization and degree of Gaussian smoothing affect the accuracy of frame selection. Herein, we discuss our findings and provide insights about our choices. We choose to binarize the patches because the binarization reduces appearance variability and suppresses the low-magnitude noise content in the patches. Without patch binarization, a large amount of variability can be expected in the appearance of wavelets that, as shown in Fig. 5.8, can deteriorate the performance of the subsequent CNN. The choice of the binarization threshold is another important factor. The use of a high threshold leads to the partial appearance of the wavelets in the resulting binary patches, but a low threshold can intensify noise content in the images. It turns out that Otsu's adaptive threshold strikes a balance between suppressing the noise content and keeping the shapes of the restored wavelets intact in all collected patches. For patches with intensity values between 0 and 1, the adaptive thresholds have a mean of 0.15 and standard deviation of 0.05. The wide range of adaptive thresholds explains why a constant threshold may not perform as desirably.

Gaussian smoothing of the probability signals is also essential for accurate frame selection because the high-frequency fluctuations of raw probability signals may complicate the localization of EUFs. These high-frequency changes are caused when the weighted centroid deviates from the center of the restored wavelet. This type of error can manifest as a sudden change in the CNN output and, as a result, in the corresponding probability signal. The second cause of high-frequency changes is the inherited high variance of CNNs. Use of ensemble of CNNs and data augmentation can alleviate this problem at a substantial computation cost. Alternatively, we choose to mitigate these undesirable fluctuations using Gaussian smoothing, which allows for both time and computational efficiency.

As described in Section 5.4.3, we constrain our ROI localization method by the location of the carotid bulb because the bulb area appears as a relatively distinct dark area in the ultrasonographic frame. The distinct appearance of the carotid bulb is also confirmed by our experiments, where we obtain the average bulb localization error of 0.26 mm for the test patients and with only 1 failure case, which is more favorable than the average unconstrained ROI localization error of 0.38 mm with 3 failure cases. Therefore, the localization of the bulb area can be done more reliably than the localization of the ROI, which motivates the use of the bulb location as a guide for more accurate ROI localization. Alternatively, we could train a regression CNN where each pixel in the image directly votes for the location of the ROI. However, this

approach may be hindered by the lack of stable anatomical structures in ultrasonography images. We will explore the use of a regression CNN for ROI localization in our future work.

In Fig. 5.7, we presented how the use of open snakes as a post-processing scheme further improved the interface localization accuracy of the CNN. Herein, we emphasize that the two open snakes employed in our system do not see the actual images during the deformation process, but rather the probability maps that are produced by the CNN. By using snakes in the CNN probability maps, we can smooth the initial step-like boundary while maximizing the probability by which each snake point lies on the desired interface. Therefore, the snake deformation process strikes a balance between boundary smoothness and adherence to CNN classification scores. A by-product of our approach is that the snake deformation process will not be directly affected by inherently large amount of noise and artifacts that are present in ultrasound images. This indeed distinguishes our approach from the literature where the snake deformation process is mainly governed by the image content. In Fig. 5.13, we demonstrated the superiority of our hybrid CNN–Snake approach over our prior work [40], wherein image gradient information together with vertical intensity scanning are used to initialize 2 image-based open snakes.

In Section 5.5.3, we showed a high level of agreement between our system and the expert for the assessment of intima–media thickness. The suggested system achieves a mean absolute error of 23.4  $\mu\text{m}$  with a standard deviation of 17.3  $\mu\text{m}$  for intima–media thickness measurements. However, this level of measurement error cannot hurt the interpretation of the vascular age, because a minimum difference of 400  $\mu\text{m}$  exists between the average intima–media thickness of the healthy and high-risk populations (600  $\mu\text{m}$  for healthy population and  $\geq 1000 \mu\text{m}$  for high-risk population) [17]. To further put the performance of our system into perspective, in Table 5.2, we show the accuracy of intima–media thickness measurements produced by our system and those of the other automatic methods recently suggested in the literature. Of note, the studies listed in Table 5.2 are evaluated using different non-public databases, and thus the tabulated results are not directly comparable.

We used a LeNet-like CNN architecture in our study, but it does not limit the suggested framework to this architecture. In fact, we have experimented with deeper CNN architectures such as AlexNet [19] in both training and fine-tuning modes; however, we did not observe a substantial performance gain. This result was probably because the higher-level semantic features detected by the deeper networks are not relevant to the tasks in our CIMT applications. Meanwhile, the concomitant computational cost of deep architectures may hinder the applicability of our system, because it lowers the speed, a key usability factor of our system. We also do not envision that a shallower architecture can offer the performance required for clinical practice. This is because a network shallower than the LeNet has only 1 convolutional layer and thus is limited to learning primitive edge-like features. Detecting the carotid bulb and the ROI and segmenting intima–media boundaries are relatively challenging tasks, requiring more than primitive edge-like features. Similarly, for frame selection, classifying the restored wavelets into R wave and non-R wave categories is similar to digit

**Table 5.2** CIMT error ( $\mu \pm \sigma$ ) for our system and the other state-of-the-art methods. Note that the listed studies are evaluated using different non-public databases, and thus the tabulated results are not directly comparable

Author	Thickness error ( $\mu\text{m}$ )	Image size	Scan resolution (mm/pixel)	Cohort
Current work	$23.4 \pm 17.3$	$640 \times 480$	0.09	23 asymptomatic patients
Bastida [4]	$13.8 \pm 31.9$	$600 \times 800$	0.029 to 0.081	27 mostly healthy patients
Ilea [16]	$80 \pm 40$	$600 \times 800$	0.029 to 0.095	23 asymptomatic patients
Loizou [27]	$30 \pm 30$	$576 \times 768$	0.060	20 symptomatic patients
Molinari [33]	$43 \pm 93$	$576 \times 768$	0.060	150 asymptomatic patients

recognition, for which LeNet is a common choice of architecture. Therefore, LeNet-like CNN architecture seems to represent an optimal balance between efficiency and accuracy for the CIMT video interpretation.

On a desktop computer with a 3.6-GHz quad core Intel and a GTX 970 GPU, our system detects each EUF in 2.9 seconds, localizes each ROI in 12.1 seconds, and measures intima–media thickness in 8.2 seconds. Although the current speed is suitable for offline processing of the CIMT videos, further performance speedup is required for an interactive use in clinical practice. We note that use of CNNs does not hinder the interactive use of our system; rather, extracting a large number of patches from a dense set of locations in the ultrasonography images causes a computational bottleneck. Therefore, substantial performance speedup can be achieved by using fully convolutional networks [28], which eliminate the need for computationally expensive image patch extraction. Further performance speedup can also be obtained using more dedicated graphics cards.

We also note that throughout this chapter, all performance evaluations were performed without involving any user interactions. However, our goal is not to exclude the user (sonographer) from the loop, but rather to relieve the sonographer from the 3 tedious, laborious, and time-consuming operations. We accomplish this relief by automating the operations while still offering a highly user-friendly interface to bring the sonographer’s indispensable expertise onto CIMT interpretation through refining the automatic results easily at the end of each of the automated operations. For instance, the user can simply press an arrow key to move 1 frame forward or backward in case the EUF is mislocalized. From our experience, the automatically localized ROI is acceptable even with a small distance from the ground truth location, but the user still can easily drag the ROI and move it around as desired. Finally, the user can adjust the CIMT measurement by changing “movable” hard constraints [23] on the snakes.

---

## 5.7 CONCLUSION

In this study, we presented a computer-aided measurement system to fully automate and accelerate CIMT video interpretation. Specifically, we suggested a computer-aided CIMT measurement system with 3 components: (i) automatic frame selection in CIMT videos, (ii) automatic ROI localization within the selected frames, and (iii) automatic intima–media boundary segmentation within the localized ROIs. We based each of these components on a CNN with a LeNet-like architecture and then boosted the performance of the CNNs with effective pre- and post-processing techniques. For frame selection, we demonstrated how patch binarization as a pre-processing step and smoothing the probability signals as a post-processing step improve the results generated by the CNN. For ROI localization, we experimentally proved that the location of the carotid bulb as a constraint in a post-processing setting, substantially improves ROI localization accuracy. For intima–media boundary segmentation, we used open snakes as a post-processing step to further improve the segmentation accuracy. We compared the results produced by the suggested system with those of our prior work, demonstrating more accurate frame selection, ROI localization, and CIMT measurements. This superior performance is attributed to the effective use of CNNs coupled with pre- and post-processing steps, uniquely designed for each of these 3 tasks.

---

## ACKNOWLEDGEMENT

This research has been partially supported by the Mayo Clinic Discovery Translation Program.

---

## REFERENCES

1. World Health Organization, Global atlas on cardiovascular disease prevention and control, September 19, 2011, available at [www.who.int/cardiovascular\\_diseases/publications](http://www.who.int/cardiovascular_diseases/publications).
2. Shun-ichi Amari, Backpropagation and stochastic gradient descent method, Neurocomputing 5 (4–5) (1993) 185–196.
3. M. Consuelo Bastida-Jumilla, Rosa M. Menchón-Lara, Juan Morales-Sánchez, Rafael Verdú-Monedero, Jorge Larrey-Ruiz, José Luis Sancho-Gómez, Segmentation of the common carotid artery walls based on a frequency implementation of active contours, J. Digit. Imaging 26 (1) (2013) 129–139.
4. M.C. Bastida-Jumilla, R.M. Menchón-Lara, J. Morales-Sánchez, R. Verdú-Monedero, J. Larrey-Ruiz, J.L. Sancho-Gómez, Frequency-domain active contours solution to evaluate intima–media thickness of the common carotid artery, Biomed. Signal Process. Control 16 (2015) 68–79.
5. Hao Chen, Qi Dou, Dong Ni, Jie-Zhi Cheng, Jing Qin, Shengli Li, Pheng-Ann Heng, Automatic fetal ultrasound standard plane detection using knowledge transferred recurrent neural networks, in: International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer, 2015, pp. 507–514.

6. Da-Chuan Cheng, Xiaoyi Jiang, Detections of arterial wall in sonographic artery images using dual dynamic programming, *IEEE Trans. Inf. Technol. Biomed.* 12 (6) (2008) 792–799.
7. Silvia Delsanto, Filippo Molinari, Pierangela Giustetto, William Liboni, Sergio Badalamenti, Jasjit S. Suri, Characterization of a completely user-independent algorithm for carotid artery segmentation in 2-D ultrasound images, *IEEE Trans. Instrum. Meas.* 56 (4) (2007) 1265–1274.
8. Q. Dou, H. Chen, L. Yu, L. Zhao, J. Qin, D. Wang, V.C. Mok, L. Shi, P.A. Heng, Automatic detection of cerebral microbleeds from MR images via 3D convolutional neural networks, *IEEE Trans. Med. Imaging* 35 (5) (2016) 1182–1195.
9. Ralph B. D'Agostino, Ramachandran S. Vasan, Michael J. Pencina, Philip A. Wolf, Mark Cobain, Joseph M. Massaro, William B. Kannel, General cardiovascular risk profile for use in primary care the Framingham heart study, *Circulation* 117 (6) (2008) 743–753.
10. David Eigen, Jason Rolfe, Rob Fergus, Yann LeCun, Understanding deep architectures using a recursive convolutional network, arXiv:1312.1847, 2013.
11. Francesco Faita, Vincenzo Gemignani, Elisabetta Bianchini, Chiara Giannarelli, Lorenzo Ghiadoni, Marcello Demi, Real-time measurement system for evaluation of the carotid intima–media thickness with a robust edge operator, *J. Ultrasound Med.* 27 (9) (2008) 1353–1361.
12. Ian J. Goodfellow, David Warde-Farley, Mehdi Mirza, Aaron Courville, Yoshua Bengio, Maxout networks, arXiv:1302.4389, 2013.
13. Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, Ruslan R. Salakhutdinov, Improving neural networks by preventing co-adaptation of feature detectors, arXiv:1207.0580, 2012.
14. David H. Hubel, Torsten N. Wiesel, Receptive fields of single neurones in the cat's striate cortex, *J. Physiol.* 148 (3) (1959) 574–591.
15. R. Todd Hurst, Robert F. Burke, Erik Wissner, Arthur Roberts, Christopher B. Kendall, Steven J. Lester, Virend Somers, Martin E. Goldman, Qing Wu, Bijoy Khandheria, Incidence of subclinical atherosclerosis as a marker of cardiovascular risk in retired professional football players, *Am. J. Cardiol.* 105 (8) (2010) 1107–1111.
16. E. Dana Ilea, Caoimhe Duffy, Liam Kavanagh, Alice Stanton, Paul F. Whelan, Fully automated segmentation and tracking of the intima media thickness in ultrasound video sequences of the common carotid artery, *IEEE Trans. Ultrason. Ferroelectr. Freq. Control* 60 (1) (2013).
17. Douglas S. Jacoby, Emile R. Mohler III, Daniel J. Rader, Noninvasive atherosclerosis imaging for predicting cardiovascular events and assessing therapeutic interventions, *Curr. Atheroscler. Rep.* 6 (1) (2004) 20–26, <http://dx.doi.org/10.1007/s11883-004-0112-8>.
18. Michael Kass, Andrew Witkin, Demetri Terzopoulos, Snakes: active contour models, *Int. J. Comput. Vis.* 1 (4) (1988) 321–331.
19. Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton, ImageNet classification with deep convolutional neural networks, in: *Advances in Neural Information Processing Systems*, 2012, pp. 1097–1105.
20. B. Boser, Y. LeCun, John S. Denker, D. Henderson, Richard E. Howard, W. Hubbard, Lawrence D. Jackel, Handwritten digit recognition with a back-propagation network, in: *Advances in Neural Information Processing Systems*, 1990.
21. Yann LeCun, Bernhard Boser, John S. Denker, Donnie Henderson, Richard E. Howard, Wayne Hubbard, Lawrence D. Jackel, Backpropagation applied to handwritten zip code recognition, *Neural Comput.* 1 (4) (1989) 541–551.

22. Yann LeCun, Yoshua Bengio, Geoffrey Hinton, Deep learning, *Nature* 521 (7553) (2015) 436–444.
23. Jianming Liang, Tim McInerney, Demetri Terzopoulos, United snakes, *Med. Image Anal.* 10 (2) (2006) 215–233.
24. Quan Liang, Inger Wendelhag, John Wikstrand, Tomas Gustavsson, A multiscale dynamic programming procedure for boundary detection in ultrasonic artery images, *IEEE Trans. Med. Imaging* 19 (2) (2000) 127–142.
25. Christos P. Loizou, A review of ultrasound common carotid artery image and video segmentation techniques, *Med. Biol. Eng. Comput.* 52 (12) (2014) 1073–1093.
26. Christos P. Loizou, Constantinos S. Pattichis, Marios Pantziaris, Andrew Nicolaides, An integrated system for the segmentation of atherosclerotic carotid plaque, *IEEE Trans. Inf. Technol. Biomed.* 11 (6) (2007) 661–667.
27. Christos P. Loizou, Takis Kasparis, Christina Spyrou, Marios Pantziaris, Integrated system for the complete segmentation of the common carotid artery bifurcation in ultrasound images, in: Harris Papadopoulos, Andreas S. Andreou, Lazaros Iliadis, Ilias Maglogiannis (Eds.), *Artificial Intelligence Applications and Innovations*, in: IFIP Adv. Inf. Commun. Technol., vol. 412, Springer, Berlin, Heidelberg, ISBN 978-3-642-41141-0, 2013, pp. 292–301.
28. Jonathan Long, Evan Shelhamer, Trevor Darrell, Fully convolutional networks for semantic segmentation, *arXiv:1411.4038*, 2014.
29. Kristen M. Meiburger, Filippo Molinari, Justin Wong, Luis Aguilar, Diego Gallo, David A. Steinman, Umberto Morbiducci, Validation of the carotid intima–media thickness variability: can manual segmentations be trusted as ground truth?, *Ultrasound Med. Biol.* 42 (7) (2016) 1598–1611.
30. Rosa-María Menchón-Lara, José-Luis Sancho-Gómez, Fully automatic segmentation of ultrasound common carotid artery images based on machine learning, *Neurocomputing* 151 (2015) 161–167.
31. Rosa-María Menchón-Lara, María-Consuelo Bastida-Jumilla, Antonio González-López, José Luis Sancho-Gómez, Automatic evaluation of carotid intima–media thickness in ultrasounds using machine learning, in: *Natural and Artificial Computation in Engineering and Medical Applications*, Springer, 2013, pp. 241–249.
32. Filippo Molinari, Guang Zeng, Jasjit S. Suri, A state of the art review on intima–media thickness (IMT) measurement and wall segmentation techniques for carotid ultrasound, *Comput. Methods Programs Biomed.* 100 (3) (2010) 201–221.
33. Filippo Molinari, Kristen M. Meiburger, Guang Zeng, Andrew Nicolaides, Jasjit S. Suri, CAUDLES-EF: carotid automated ultrasound double line extraction system using edge flow, *J. Digit. Imaging* 24 (6) (2011) 1059–1077, <http://dx.doi.org/10.1007/s10278-011-9375-0>.
34. Filippo Molinari, Kristen M. Meiburger, Luca Saba, Guang Zeng, U. Rajendra Acharya, Mario Ledda, Andrew Nicolaides, Jasjit S. Suri, Fully automated dual-snake formulation for carotid intima–media thickness measurement a new approach, *J. Ultrasound Med.* 31 (7) (2012) 1123–1136.
35. Dariush Mozaffarian, Emelia J. Benjamin, Alan S. Go, Donna K. Arnett, Michael J. Blaha, Mary Cushman, Sarah de Ferranti, Jean-Pierre Despres, Heather J. Fullerton, Virginia J. Howard, et al., Heart disease and stroke statistics – 2015 update: a report from the American Heart Association, *Circulation* 131 (4) (2015) e29.
36. Nobuyuki Otsu, A threshold selection method from gray-level histograms, *Automatica* 11 (285–296) (1975) 23–27.

37. Styliani Petroudi, Christos Loizou, Marios Pantziaris, Constantinos Pattichis, Segmentation of the common carotid intima–media complex in ultrasound images using active contours, *IEEE Trans. Biomed. Eng.* 59 (11) (2012) 3060–3069.
38. P. Pignoli, T. Longo, Evaluation of atherosclerosis with b-mode ultrasound imaging, *J. Nucl. Med. Allied Sci.* 32 (3) (1987) 166–173.
39. Alessandro C. Rossi, Peter J. Brands, Arnold P.G. Hoeks, Automatic localization of intimal and adventitial carotid artery layers with noninvasive ultrasound: a novel algorithm providing scan quality control, *Ultrasound Med. Biol.* 36 (3) (2010) 467–479.
40. Haripriya Sharma, Ramsri G. Golla, Yu Zhang, Christopher B. Kendall, R. Todd Hurst, Nima Tajbakhsh, Jianming Liang, ECG-based frame selection and curvature-based ROI detection for measuring carotid intima–media thickness, in: *SPIE Medical Imaging, International Society for Optics and Photonics*, 2014, 904016.
41. Hoo-Chang Shin, Holger R. Roth, Mingchen Gao, Le Lu, Ziyue Xu, Isabella Nogues, Jianhua Yao, Daniel Mollura, Ronald M. Summers, Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics and transfer learning, *IEEE Trans. Med. Imaging* 35 (5) (2016) 1285–1298.
42. J.H. Stein, C.E. Korcarz, R.T. Hurst, E. Lonn, C.B. Kendall, E.R. Mohler, S.S. Najjar, C.M. Rembold, W.S. Post, American Society of Echocardiography carotid intima–media thickness task force. Use of carotid ultrasound to identify subclinical vascular disease and evaluate cardiovascular disease risk: a consensus statement from the American Society of Echocardiography carotid intima–media thickness task force. Endorsed by the Society for Vascular Medicine, *J. Am. Soc. Echocardiogr.* 21 (2) (2008) 93–111.
43. Nima Tajbakhsh, Jae Y. Shin, Suryakanth R. Gurudu, R. Todd Hurst, Christopher B. Kendall, Michael B. Gotway, Jianming Liang, Convolutional neural networks for medical image analysis: full training or fine tuning?, *IEEE Trans. Med. Imaging* 35 (5) (2016) 1299–1312.
44. Pierre-Jean Touboul, Patrizio Prati, Pierre-Yves Scarabin, Valérie Adrai, Emmanuel Thibout, Pierre Ducimetière, Use of monitoring software to improve the measurement of carotid wall thickness by b-mode imaging, *J. Hypertens.* 10 (1992) S37–S42.
45. Li Wan, Matthew Zeiler, Sixin Zhang, Yann L. Cun, Rob Fergus, Regularization of neural networks using DropConnect, in: *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, 2013, pp. 1058–1066.
46. Jun Xu, Xiaofei Luo, Guanhao Wang, Hannah Gilmore, Anant Madabhushi, A deep convolutional neural network for segmenting and classifying epithelial and stromal regions in histopathological images, *Neurocomputing* 191 (2016) 214–223.
47. Xiangyang Xu, Yuan Zhou, Xinyao Cheng, Enmin Song, Guokuan Li, Ultrasound intima–media segmentation using hough transform and dual snake model, *Comput. Med. Imaging Graph.* 36 (3) (2012) 248–258.
48. Xiangjun Zhu, Christopher B. Kendall, R. Todd Hurst, Jianming Liang, A user friendly system for ultrasound carotid intima–media thickness image interpretation, in: *SPIE Medical Imaging, International Society for Optics and Photonics*, 2011, 79681G.

## NOTES

1. This should not to be confused with the wavelet transform in signal processing. A wavelet, in this context, refers to a small part of the ECG signal.
2. <https://github.com/sdemyanov/ConvNet>.

This page intentionally left blank

# Deep Cascaded Networks for Sparsely Distributed Object Detection from Medical Images

# 6

Hao Chen\*, Qi Dou\*, Lequan Yu\*, Jing Qin<sup>†</sup>, Lei Zhao\*, Vincent C.T. Mok\*,  
Defeng Wang\*, Lin Shi\*, Pheng-Ann Heng\*

The Chinese University of Hong Kong, Hong Kong, China\*

The Hong Kong Polytechnic University, Hong Kong, China<sup>†</sup>

## CHAPTER OUTLINE

<b>6.1</b>	<b>Introduction</b>	134
<b>6.2</b>	<b>Method</b>	136
6.2.1	<i>Coarse Retrieval Model</i>	136
6.2.1.1	<i>Score Mask Generation</i>	137
6.2.1.2	<i>Object Candidate Localization</i>	138
6.2.2	<i>Fine Discrimination Model</i>	139
<b>6.3</b>	<b>Mitosis Detection from Histology Images</b>	139
6.3.1	<i>Background</i>	139
6.3.2	<i>Transfer Learning from Cross-Domain</i>	140
6.3.3	<i>Dataset and Preprocessing</i>	140
6.3.4	<i>Quantitative Evaluation and Comparison</i>	141
6.3.5	<i>Computation Cost</i>	142
<b>6.4</b>	<b>Cerebral Microbleed Detection from MR Volumes</b>	143
6.4.1	<i>Background</i>	143
6.4.2	<i>3D Cascaded Networks</i>	144
6.4.2.1	<i>3D Convolutional Neural Network</i>	144
6.4.2.2	<i>Screening Stage</i>	144
6.4.2.3	<i>Discrimination Stage</i>	145
6.4.3	<i>Dataset and Preprocessing</i>	146
6.4.4	<i>Quantitative Evaluation and Comparison</i>	147
6.4.4.1	<i>CMB Candidate Localization</i>	147
6.4.4.2	<i>True CMB Discrimination</i>	147
6.4.5	<i>System Implementation</i>	149
<b>6.5</b>	<b>Discussion and Conclusion</b>	149

Acknowledgements .....	150
References .....	150
Notes .....	154

## CHAPTER POINTS

---

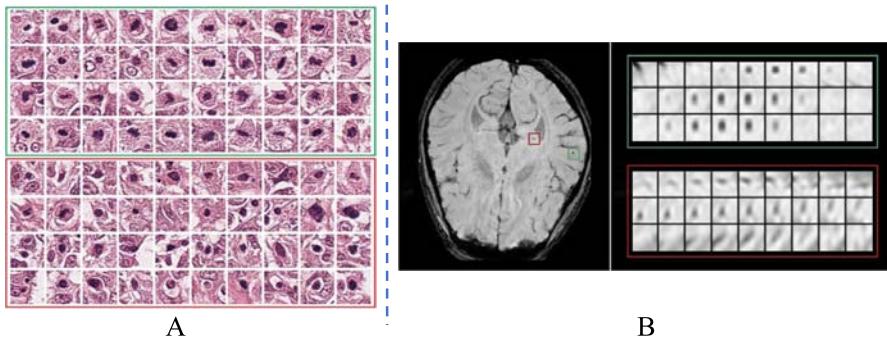
- A cascaded deep learning framework, composed of a coarse retrieval model and a fine discrimination model, is proposed to efficiently and accurately detect sparsely distributed objects from medical images
- We demonstrate the importance of volumetric feature representations by exploiting 3D deep learning on volumetric imaging modalities
- Extensive experimental results on two challenging applications including both 2D and 3D imaging modalities corroborated the outstanding performance and generalization capability of our method in terms of speed and accuracy

---

## 6.1 INTRODUCTION

Computer-aided diagnosis (CAD) has become one of the major research subjects in medical image computing and clinical diagnosis [1]. In the pipeline of a CAD system, clinically related object detection from medical images plays an important role, which has a wide range of applications, such as mitosis detection [2], nuclei detection [3], cerebral microbleed detection [4], lung nodule detection [5], cell tracking, lesion localization [6], etc. In general, the clinical routine to annotate targets of interest is based on visual inspection and manual localization [1], which suffers from a low reproducibility among different observers and could be laborious and time-consuming, especially within the context of large numbers of subjects. Robust object detection from medical images can release doctors from considerable workload, provide reliable quantitative assessments, and accelerate the diagnosis procedure.

However, how to efficiently and effectively detect the sparsely distributed objects from large-scale data remains a challenging problem. Although the situation may vary depending on the imaging modalities and confronting detection tasks, there are several common challenges: (i) There is a high intra-class variation of object appearance in medical images. Take the mitosis for example, it is characterized by a large variety of shape configurations, which are related to the high variation of biological structures, as shown in the green rectangle in Fig. 6.1A. Furthermore, the different conditions of histology image acquisition process, including sampling, cutting, staining, and digitalizing, increase the variabilities of mitosis appearance [2,7]. This is common when the tissue samples are acquired from different patients or at different time slots. (ii) The sparsely distributed objects require the algorithm to be efficient and robust when applied to large-scale data. In the example of cerebral microbleeds (CMBs), the widespread distributed locations of CMBs make complete and accurate detection very challenging [8,9]. (iii) There are strong mimics that hinder the detection process, therefore the automatic methods should be robust and discriminative to

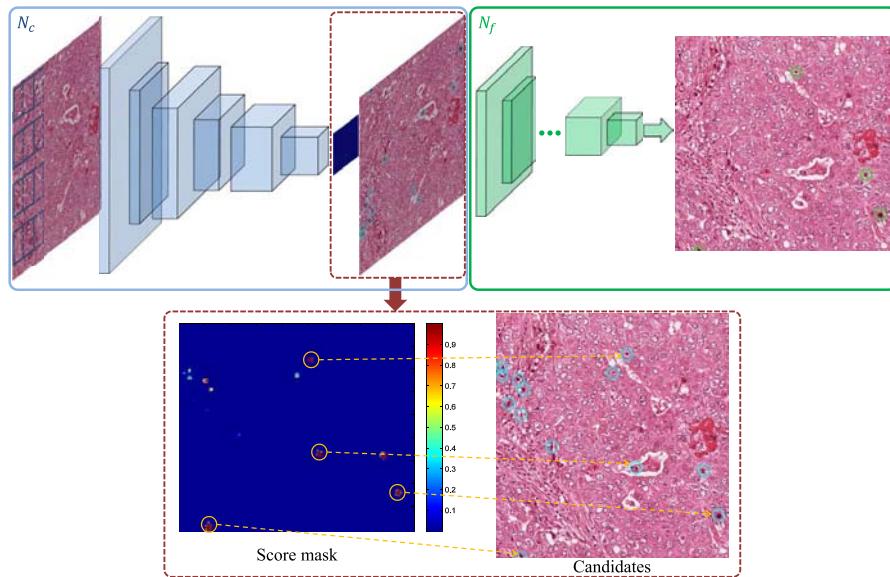
**FIGURE 6.1**

Examples of target objects and mimics. (A) Mitosis detection (green rectangle encloses the true mitoses and red rectangle encloses the mimics that carry similar appearance); (B) CMB and mimics are denoted with green and red rectangles, respectively (in each of the rectangle, from top to down, the rows show adjacent slices in axial, sagittal and coronal planes).

single out the true objects from hard mimics. In the case of mitosis detection, some cell types (e.g., apoptotic cells) carry similar morphological appearance with mitosis, as shown in the red rectangle in Fig. 6.1A, resulting in lots of false positives in the detection process. Similarly, in the task of CMB detection, there exist many sources of hard CMB mimics, e.g., flow voids, calcification and cavernous malformations, see the red rectangle in Fig. 6.1B, which would resemble the appearance of CMBs in magnetic resonance (MR) scans and heavily impede the detection process [10].

Recently, deep neural networks have made breakthroughs in image recognition related tasks including object detection [11], semantic segmentation [12], and image classification [13,14]. Inspired by this and aiming to tackle the aforementioned challenges, we propose a fast and accurate method to detect sparsely distributed objects from medical images by designing deep cascaded neural networks (CasNN). Extensive experimental results on two typical applications including mitosis detection from histopathological images and CMB detection from MR volumes demonstrated promising performance and high generalization capability of our framework.

The remainder of this chapter is organized as follows. In Section 6.2, we describe the cascaded framework based on deep convolutional neural networks. Extensive experimental results on 2D histopathological images in Section 6.3, and volumetric data in Section 6.4 demonstrate the robustness and efficiency of the proposed method. Meanwhile, Section 6.4 elaborates the 3D CNN and verifies the effectiveness of volumetric feature representations for 3D detection tasks. This chapter concludes with a discussion in Section 6.5. An early version of this work was published in [4] and [15].

**FIGURE 6.2**

An overview of deep cascaded networks for sparsely distributed object detection. The cascaded framework consists of a coarse retrieval model  $N_c$  and a fine discrimination model  $N_f$  (we take the task of mitosis detection from histopathological images for illustration).

## 6.2 METHOD

[Fig. 6.2](#) shows the architecture of the proposed method (we take the example of mitosis detection for illustration), which consists of two models of convolutional neural networks integrated in a cascaded manner. The first model quickly retrieves the object candidates while preserving a high sensitivity by taking the advantage of the fully convolutional network. We call it the *coarse retrieval model*  $N_c$ , which outputs a score map indicating the probability of candidates. The second model is effectively constructed by a deeper convolutional neural network. We call it the *fine discrimination model*  $N_f$ , which has a higher capability of feature representation, hence it accurately discriminates the true targets and mimics. Note that as the  $N_f$  performs only on the candidates generated by  $N_c$  instead of the whole image, hence the detection process can be significantly accelerated.

### 6.2.1 COARSE RETRIEVAL MODEL

Considering that objects are sparsely distributed in the whole image, a step of retrieving the regions of interest (ROI), e.g., mitosis candidates, could reduce the detection time dramatically, as the subsequent detection process could focus only on

the candidates. Previous studies obtained the candidates relying on the pre-defined measurements on domain-specific morphological textures, color ratios, or histogram distribution [16–19]. However, these methods were prone to losing targets, as these handcrafted features could not accurately describe the complicated characteristics of objects. Recently, there have been studies that explored the fast scanning approach with deep max-pooling convolutional neural networks for the mitosis detection [20, 21]. Although these methods are more accurate than previous studies based on low-level features, they are computation-intensive and time-consuming in a pixel-wise classification way.

Different from previous methods, we utilize a fully convolutional network (FCN) for fast retrieving of the mitosis candidates. A traditional convolutional neural network (CNN) contains the convolutional (C), sub-sampling, e.g., max-pooling (M), and fully connected (FC) layers. Both C and M layers are translation-invariant and can be operated on input of arbitrary size. However, the introduction of FC layers requires the input with a fixed size,

$$\mathbf{h}_j^l = \sigma(\mathbf{W}_j^l \mathbf{h}^{l-1} + \mathbf{b}_j^l), \quad (6.1)$$

where  $\mathbf{W}_j^l$  is the weight matrix connecting the neurons  $\mathbf{h}^{l-1}$  in  $(l-1)$ th FC layer and  $j$ th index neuron  $\mathbf{h}_j^l$  in the  $l$ th FC layer,  $\mathbf{b}_j^l$  is the bias and  $\sigma(\cdot)$  is the element-wise nonlinear activation function. In fact, the fully connected layers are equivalent to the convolutional layers with kernel size  $1 \times 1$ ,

$$\mathbf{h}_j^l = \sigma\left(\sum_k \mathbf{W}_{jk}^l \otimes \mathbf{h}_k^{l-1} + \mathbf{b}_j^l\right), \quad (6.2)$$

where  $\otimes$  denotes the 2D spatial convolution,  $\mathbf{W}_{jk}^l$  is the convolution kernel connected to  $j$ th feature map  $\mathbf{h}_j^l$  and the  $k$ th feature map in the previous layer  $\mathbf{h}^{l-1}$ . By employing Eq. (6.2), we can convert the fully connected layers into a fully convolutional fashion. Once the filters have been trained, the fully convolutional network can be applied to an input image of arbitrary size.

The proposed coarse retrieval model has several advantages. First, since the FCN can take the whole image as input and generate the score mask with only one pass of forward propagation, it is capable of retrieving candidates efficiently while preserving a high sensitivity. Second, it can also help to build a representative training database for the fine discrimination model  $N_f$ . Specifically, the targeting objects are sparsely distributed and rarely appeared in the whole image, the mimics can be well represented by putting the false positives from  $N_c$  into the training samples of  $N_f$ . In this way, the capability of the model  $N_f$  in distinguishing the true targets from the hard mimics can be greatly enhanced.

### 6.2.1.1 Score Mask Generation

The proposed coarse retrieval model is trained on the training samples with a fixed size input ( $94 \times 94 \times 3$ , architecture of  $N_c$  for mitosis detection is shown in Table 6.1)

**Table 6.1** The architecture of coarse retrieval model  $N_c$ 

Layer	Kernel size	Stride	Output size	Feature maps
Input	—	—	94 × 94	3
C1	5 × 5	1	90 × 90	32
M1	3 × 3	3	30 × 30	32
C2	3 × 3	1	28 × 28	32
M2	2 × 2	2	14 × 14	32
C3	3 × 3	1	12 × 12	32
M3	2 × 2	2	6 × 6	32
FC4	—	—	100	—
FC5	—	—	2	—

by minimizing the cross entropy loss. Once the training is done, the  $N_c$  can be converted into a FCN model by Eq. (6.2). Then, the trained filters can be applied to scan the whole image instead of employing the traditional patch-by-patch manner, which speeds up the detection process dramatically. Hence, the score mask indicating probabilities of candidates can be obtained with only a single forward propagation of the converted coarse retrieval model. Each position of the output score mask corresponds to a specific region (size 94 × 94) in the original image. Actually, it is equivalent to scanning the whole image with a fixed stride, which is mainly determined by the stride of max-pooling layers. We will detail this in the following section.

### 6.2.1.2 Object Candidate Localization

Derived from the proposed model, the candidates can be located by mapping the index with higher scores on the score mask into the original coordinates of input image. Assuming non-overlapping region pooling, index mapping with convolution and max-pooling operations is formulated as

$$\hat{x}_i = \frac{x_i - c_i}{s_i} + 1, \quad (6.3)$$

where  $c_i$  denotes the kernel size of convolutional or max-pooling layer,  $\hat{x}_i$  is the position index after C or M operation on  $x_i$ , and  $s_i$  denotes the stride of convolutional or max-pooling layer. The original position index can be obtained by inverting above operations. For example, based on the network architecture shown in Table 6.1, for each position  $\hat{p}_s$  in the score mask, we can get the index  $p_o$  in the original image as

$$\begin{aligned} p_o &= c_1 - 1 + s_{m_1}(c_2 - 1) + s_{m_1}s_{m_2}(c_3 - 1) \\ &\quad + s_{m_1}s_{m_2}s_{m_3}\hat{p}_s = p_0 + s\hat{p}_s, \end{aligned} \quad (6.4)$$

where  $p_0 = 22$  and  $s = 12$  according to the architecture in Table 6.1. Thus we can retrieve the candidates with a sparse distribution based on the above index mapping. This is quite efficient when the target objects are sparsely distributed in the whole image. Despite with max-pooling layers, the probability maps can give quite dense

predictions considering the equivalent stride 12 compared with the image size  $2048 \times 2048$ . Therefore, this approach can efficiently retrieve the candidates with a high sensitivity while reducing the computational workload. Subsequently, candidates are input into the model  $N_f$  for fine discrimination after local smoothing and non-max suppression.

### 6.2.2 FINE DISCRIMINATION MODEL

In order to increase the representation capability for singling out true targets from mimics, the fine discrimination model is constructed by a deeper convolutional neural network (e.g., we used CaffeNet [22] with 8 layers in mitosis detection). In addition, the training samples are augmented with different transformations to enlarge the training dataset. Finally, the fine discrimination model  $N_f$  is optimized by minimizing the following cross-entropy function with standard back-propagation:

$$\arg \min_{\theta} \sum_{n=1}^N \sum_{k=1}^K -t_k \log p(y_k = 1 | \mathbf{I}_n) + \lambda \|\mathbf{W}\|_2^2, \quad (6.5)$$

where  $\theta = \{\mathbf{W}, \mathbf{b}\}$  denotes the parameters of  $N_f$ ,  $\lambda$  is the parameter for controlling the balance between the data loss term and the regularization term,  $p(y_k = 1 | \mathbf{I}_n)$  is the output probability for  $k$ th class given the input sub-window patch  $\mathbf{I}_n$ ,  $t_k$  is the corresponding ground truth,  $K$  and  $N$  are the total number of classes and training samples, respectively. In the training process, dropout method [23] was utilized to reduce the co-adaption of intermediate features. In order to improve the robustness, we trained multiple models of  $N_f$  for reducing the variance and improving the robustness capability [24]. For example, three architectures with different number of neurons in three fully connected layers (i.e., FC6–FC8), 1024–256–2, 1024–512–2, 512–256–2, were trained in the mitosis detection. The sub-window sample was categorized as a true target when its averaged output posterior probability was above the threshold  $T$  (determined with cross-validation in our experiments), otherwise, categorized as non-target.

---

## 6.3 MITOSIS DETECTION FROM HISTOLOGY IMAGES

### 6.3.1 BACKGROUND

Breast cancer is the most common cancer for women and a major cause of cancer death worldwide [25]. According to the Nottingham Grading System, three morphological features in histology sections, including tubule formation, nuclear pleomorphism and number of mitotic figures, are critical for the diagnosis of breast cancer [26]. Among them, the number of mitoses gives an important aggressiveness indication of the invasive breast carcinoma.

Previous studies employed domain-specific handcrafted features to describe the morphological, statistical, or textural characteristics of mitosis [16–19]. However,

these handcrafted features usually require considerable efforts to design and validate. Furthermore, they cannot sufficiently represent the characteristics of mitoses with a large variation of shapes and textures, therefore resulting in a low detection accuracy. Compared with methods based on handcrafted features, the deep CNN with powerful feature representation learning has achieved state-of-the-art performance in object recognition related problems [13,14,27]. Regarding the mitosis detection, Ciresan et al. utilized a deep CNN as a pixel-wise classifier to detect mitosis and achieved the best performance at 2012 ICPR MITOSIS challenge [21] and 2013 MICCAI challenge [2], respectively. However, the pixel-wise classifier of deep CNN is computation-intensive and time-consuming. Considering a single whole slide that consists of thousands of high-power fields (HPFs), it takes a long time to run across all sub-windows for detection, which may prohibit its potential in clinical practice.

### 6.3.2 TRANSFER LEARNING FROM CROSS-DOMAIN

The deep CNN with powerful feature representation achieved remarkable performance on recognition related tasks with large scale training data available. However, limited data in the field of medical applications increases the difficulties for training a powerful model to discriminate objectives from their mimics. The situation is further deteriorated with existence lots of false positives carrying similar appearance. Although various transformation augmentations could be used to enlarge the training database, the training samples may be still insufficient to train a powerful model. Further improvement can be obtained by transferring knowledge learned from related auxiliary tasks, where the training data can be easily acquired. Previous studies demonstrated that the filters trained on large scale images of ImageNet [28] could be transferred to different applications in other domains empirically [22,29–31]. Therefore, we optimized the new medical task by employing an off-the-shelf model CaffeNet [22]. The parameters of the previous layers (C1–C5) in the  $N_f$  were initialized by the pre-trained filters of CaffeNet model, which was trained on large scale images of ImageNet. Then the whole network was jointly optimized on the specific task.

### 6.3.3 DATASET AND PREPROCESSING

The datasets were obtained from the 2012<sup>1</sup> and 2014 ICPR MITOSIS contests.<sup>2</sup> In our experiments, we evaluated our method on the HPF images acquired by the widely-used Apero-XT scanner. The 2012 ICPR MITOSIS dataset consisted of 50 images (train/test = 35/15) with 326 annotated mitoses, while the 2014 ICPR MITOSIS dataset was extensively enlarged (train/test = 1200/496). The centroids of mitoses were annotated by experienced pathologists and the ground truths of 2014 ICPR testing data were held out by the organizers for evaluation. For each dataset, we split training data with ground truth into two sets for training and validation (about 1/7 of total training data), respectively. Patches extracted from mitotic regions were augmented by different transformations, including translation, rotation and flipping, for enlarging the training dataset.

**Table 6.2** Results of 2012 ICPR MITOSIS dataset

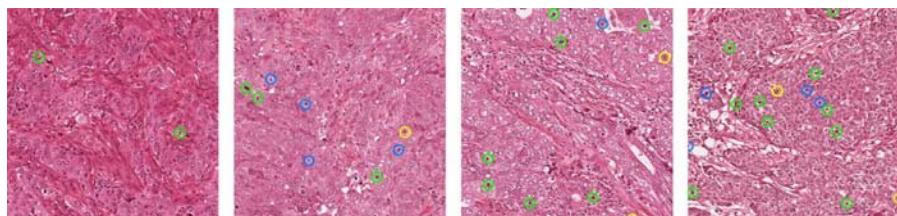
Method	Precision	Recall	<i>F</i> <sub>1</sub> score
SUTECH	0.699	0.720	0.709
IPAL [16]	0.698	0.740	0.718
DNN [21]	<b>0.886</b>	0.700	0.782
Retrieval model $N_c$	0.211	<b>0.891</b>	0.342
RCasNN	0.720	0.713	0.716
CasNN (single)	0.738	0.753	0.745
CasNN (average)	0.804	<b>0.772</b>	<b>0.788</b>

### 6.3.4 QUANTITATIVE EVALUATION AND COMPARISON

According to the challenge evaluation, a detection would be regarded as a true positive if its distance to a ground truth mitosis is less than 8  $\mu\text{m}$ . All the detections that are not fallen within 8  $\mu\text{m}$  of a ground truth are counted as false positives. All the ground truths that do not have detections within 8  $\mu\text{m}$  are counted as false negatives. The evaluation measurements include recall  $R = N_{tp}/(N_{tp} + N_{fn})$ , precision  $P = N_{tp}/(N_{tp} + N_{fp})$ , and  $F_1$ -score  $F_1 = 2RP/(R + P)$ , where  $N_{tp}$ ,  $N_{fp}$ , and  $N_{fn}$  are the number of true positives, false positives, and false negatives, respectively.

In order to build a representative dataset for training the fine discrimination model, false positives from the coarse retrieval model were employed. Note that the size of training data (total 394,275 training samples including 7.4% mitoses, 67.8% random selected negative samples, and 24.8% false positives from  $N_c$ ) is smaller compared to the training dataset used in the state-of-the-art method [21] (1 million training instances with 6.6% as mitoses). Although with less aggressive data augmentation, our method achieved a competitive performance on testing data with a much faster speed. The detailed results are reported in Table 6.2. Compared to the method with the best performance in 2012 ICPR contest [21], our method (CasNN) with model averaging achieved a comparable  $F_1$ -score of 0.788 and a higher recall of 0.772. The single CasNN model outperformed the randomly initialized model (RCasNN) with respect to all the evaluation measurements, demonstrating that the knowledge transferred from deep and rich hierarchies can help to improve the performance. Four typical detection examples on the testing data of 2012 ICPR MITOSIS contest are shown in Fig. 6.3. It is observed that the tissue appearance has large variations, which increases the difficulties for automated mitosis detection. Nevertheless, our method can successfully detect most of mitoses from histology images with a low false negative rate and only a few false positives, which verifies the efficacy of our method qualitatively.

As the organizers from 2012 ICPR MITOSIS contest indicated that the dataset in this contest is too small for a good assessment of reliability and robustness of different algorithms [32], we further evaluated our method on the 2014 ICPR MITOSIS dataset, which was extensively expanded (1200 training images and 496 testing images). One of the most difficult challenges in this dataset is the variability of tissue appearance, mostly resulted from the different conditions during the tissue acqui-

**FIGURE 6.3**

Mitosis detection results of our method on the testing data of 2012 ICPR MITOSIS contest. Yellow, blue, and green circles denote the false negatives, false positives, and true positives, respectively.

**Table 6.3** Results of 2014 ICPR MITOSIS dataset

<b>Method</b>	<b>Precision</b>	<b>Recall</b>	<b><math>F_1</math> score</b>
STRASBOURG	—	—	0.024
YILDIZ	—	—	0.167
MINES-CURIE-INSERM	—	—	0.235
CUHK	—	—	0.356
RCasNN	0.360	0.424	0.389
CasNN (single)	0.411	0.478	0.442
CasNN (average)	<b>0.460</b>	<b>0.507</b>	<b>0.482</b>

tion process. As a result, the dataset is much more challenging than that in 2012 ICPR. The results of different methods are reported in Table 6.3. Our approach achieved the best performance with  $F_1$ -score of 0.482, outperforming other methods by a large margin. The performance of the single CasNN model outperformed the RCasNN model, demonstrating the efficacy of transfer learning strategy consistently.

### 6.3.5 COMPUTATION COST

In histopathological examination of breast cancer diagnosis, a single whole slide usually consists of thousands of HPFs. Thus, the processing time of one HPF should be taken into account in clinical applications [2]. The superior advantage of the proposed cascaded framework is that it can reduce detection time dramatically while achieving a satisfactory accuracy. The coarse retrieval model took about 0.45 s to process per 4 megapixel HPF and the fine discrimination model with 10 input variations cost about 0.49 s using a workstation with a 2.50 GHz Intel(R) Xeon(R) E5-2609 CPU and a NVIDIA GeForce GTX TITAN GPU. Totally, it took about 0.5 s for each input variation and was roughly 60 times faster than the state-of-the-art method [21], which took about 31 s with an optimized GPU implementation. Meanwhile, our approach achieved comparable detection accuracy to [21]. This makes our approach possible for real-world clinical applications.

---

## 6.4 CEREBRAL MICROBLEED DETECTION FROM MR VOLUMES

### 6.4.1 BACKGROUND

Cerebral microbleeds refer to small foci of chronic blood products in (near) normal brain tissues, as shown in Fig. 6.1B, which are considered to be composed of hemosiderin deposits that leak through pathological blood vessels [33]. The existence of CMBs and their distribution patterns have been recognized as important diagnostic biomarkers of cerebrovascular diseases. In addition, CMBs could also structurally damage their nearby brain tissues, and further cause neurologic dysfunction, cognitive impairment, and dementia [34]. In this regard, reliable detection of the presence and number of CMBs is crucial for cerebral diagnosis and guiding physicians in determining necessary treatment.

Previous computer-aided CMB detection methods mainly employed handcrafted features based on shape, size and intensity information. For example, Fazlollahi et al. [35] utilized the radon transform to describe the shape information of CMBs, and Kuijf et al. [36] have applied the radial symmetry transform (RST) to identify spherical regions. To improve the capability of discrimination, Bian et al. [37] proposed to measure the geometric features after performing a 2D fast RST. Ghafaryasl et al. [38] further designed more comprehensive features that integrated the geometry, intensity, scale and local image structures. However, the design of these handcrafted features heavily depends on domain knowledge of the lesion. Furthermore, these low-level features are usually insufficient to capture the complicated characteristics of CMBs. Recently, some investigations have been dedicated to learning features in a data driven way in order to more accurately detect CMBs [39,31]. Among them, convolutional neural network is one of the most promising solutions to meet the challenges of CMB detection by virtue of its high capability in extracting powerful high-level features.

Our objective in this task is to detect CMBs which are sparsely distributed in a 3D brain volume. However, how to effectively employ CNNs on volumetric data still remains an open problem in medical image computing community, even though CNNs have presented outstanding effectiveness on 2D medical imaging applications [40–42]. One straightforward solution is to employ conventional 2D CNNs based on a single slice and process the slices sequentially [43–45]. Apparently, this solution disregards the contextual information along the third dimension, thus its performance would be heavily degraded. Alternatively, some researches aggregate adjacent slices [31] or orthogonal planes (i.e., axial, coronal and sagittal) [46,47] to enhance complementary spatial information. Nevertheless, this solution is still unable to make full use of the volumetric spatial information. Note that the spatial information of all three dimensions is quite important for our CMB detection task. As shown in Fig. 6.1B, the mimic can resemble the CMB in the view of one or two dimensions, but when taking the characteristics of all three dimensions into consideration, it is much easier to distinguish the CMB from the mimic. To the end, a 3D version of CNN is a more promising and reliable solution to take full advantage of spatial contextual information in volumetric data for accurate detection of CMBs.

### 6.4.2 3D CASCADED NETWORKS

We extend the detection framework of 2D cascaded networks into a volumetric version. In the first screening stage, the 3D fully convolutional network takes a whole volumetric data as input and directly outputs a 3D score volume. Each value on the 3D score volume represents the probability of CMB at a corresponding voxel of the input volume. In the second discrimination stage, we further remove a large number of false positive candidates by applying a 3D CNN discrimination model to distinguish the true CMBs from challenging mimics.

#### 6.4.2.1 3D Convolutional Neural Network

Learning feature representations from volumetric contextual information is vitally important for biomarker detection tasks from 3D medical images. In this regard, we propose to employ the 3D convolution kernel, in the pursuit of encoding richer spatial information of the volumetric data. In this case, the feature maps are 3D blocks instead of 2D patches (we call them *feature volumes* hereafter). Given a volumetric image, when we employ a 3D convolution kernel to generate a 3D feature volume, the input to the network is the entire volumetric data. By leveraging the kernel sharing across all three dimensions, the network can take full advantage of the volumetric contextual information. Generally, the following equation formulates the exploited 3D convolution operation in an element-wise fashion:

$$\mathbf{u}_{jk}^l(x, y, z) = \sum_{m,n,t} \mathbf{h}_k^{l-1}(x-m, y-n, z-t) \mathbf{W}_{jk}^l(m, n, t), \quad (6.6)$$

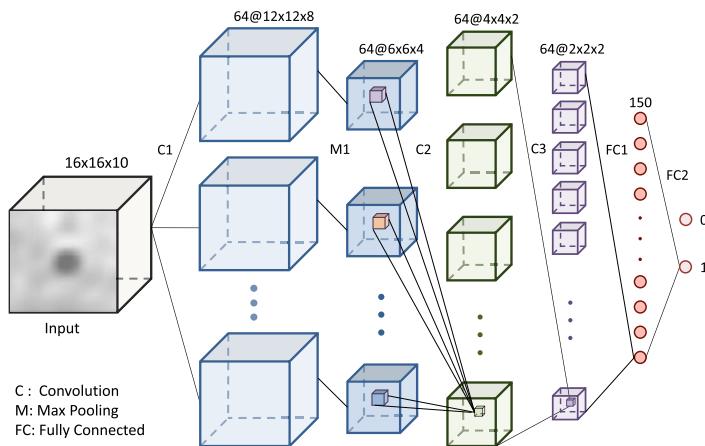
where  $\mathbf{W}_{jk}^l$  is the 3D kernel in the  $l$ th layer which convolves over the 3D feature volume  $\mathbf{h}_k^{l-1}$  in the  $(l-1)$ th layer,  $\mathbf{W}_{jk}^l(m, n, t)$  is the element-wise value of the 3D convolution kernel. The 3D feature volume  $\mathbf{h}_j^l$  is obtained by different 3D convolution kernels as

$$\mathbf{h}_j^l = \sigma(\sum_k \mathbf{u}_{jk}^l + \mathbf{b}_j^l). \quad (6.7)$$

Then a 3D CNN model is constructed by stacking the C, M and FC layers hierarchically, as shown in Fig. 6.4. Specifically, in the C layer, multiple 3D feature volumes are produced. In the M layer, the max-pooling operation is also performed in a 3D fashion, i.e., the feature volumes are sub-sampled based on a cubic neighborhood. In the following FC layer, 3D feature volumes are flattened into a feature vector as its input. The rectifier linear unit (ReLU) [48] is utilized for the nonlinear activation function in the C and FC layers. Finally, the output layer employs the softmax regression to yield the prediction probabilities.

#### 6.4.2.2 Screening Stage

Previous works proposed to screen CMB candidates in a MR volume by employing local statistical information, including size, intensity, shape and other geometric features [49,31,38,39]. However, due to the large variations of CMBs, only relying on

**FIGURE 6.4**

The hierarchical architecture of the 3D CNN model.

these statistic values is difficult to precisely describe the characteristics of CMBs and detach them from the background regions. The results would easily either neglect true CMBs or include a large number of false positives, which can complicate the following discrimination procedure.

We propose to use 3D CNN to robustly screen candidates by leveraging high-level discriminative representations of CMBs learned from a large number of 3D samples. Inspired by the 2D FCN, we extend the strategy into a 3D format, i.e., 3D FCN, for efficient and effective retrieval of CMB candidates from MR volumetric data. During the training phase, the positive samples are extracted from CMB regions and augmented by translation, rotation, and mirroring to expand the training database. During the testing phase, the 3D FCN model takes the whole volume as input and generates the corresponding coarse 3D score volume. Considering that the produced score volume could be noisy, we utilize the local non-max suppression in a 3D fashion as the post-processing. Locations in the 3D score volume are then sparsely traced back to coordinates in the original input space, according to the index mapping and architecture in [Table 6.4](#). Finally, regions with high prediction probabilities are selected as the potential candidates.

#### **6.4.2.3 Discrimination Stage**

The screened 3D candidate regions are classified by a newly constructed 3D CNN model. We notice that the randomly selected non-CMB samples are not strongly representative, especially when we aim to distinguish true CMBs from their mimics. To generate representative samples and therefore improve the discrimination capability of the 3D CNN model, the obtained false positives (which take very similar appearance to true CMBs) on the training set in the screening stage are taken as negative

**Table 6.4** The architecture of 3D FCN screening model

<b>Layer</b>	<b>Kernel size</b>	<b>Stride</b>	<b>Output size</b>	<b>Feature volumes</b>
Input	—	—	$16 \times 16 \times 10$	1
C1	$5 \times 5 \times 3$	1	$12 \times 12 \times 8$	64
M1	$2 \times 2 \times 2$	2	$6 \times 6 \times 4$	64
C2	$3 \times 3 \times 3$	1	$4 \times 4 \times 2$	64
C3	$3 \times 3 \times 1$	1	$2 \times 2 \times 2$	64
FC1	—	—	150	—
FC2	—	—	2	—

**Table 6.5** The architecture of 3D CNN discrimination model

<b>Layer</b>	<b>Kernel size</b>	<b>Stride</b>	<b>Output size</b>	<b>Feature volumes</b>
Input	—	—	$20 \times 20 \times 16$	1
C1	$7 \times 7 \times 5$	1	$14 \times 14 \times 12$	32
M1	$2 \times 2 \times 2$	2	$7 \times 7 \times 6$	32
C2	$5 \times 5 \times 3$	1	$3 \times 3 \times 4$	64
FC1	—	—	500	—
FC2	—	—	100	—
FC3	—	—	2	—

samples when training the 3D CNN in the second stage. The network architecture of the discrimination model is shown in [Table 6.5](#).

### 6.4.3 DATASET AND PREPROCESSING

A large dataset of susceptibility-weighted imaging (SWI) images, referred to as *SWI-CMB*, was built to validate the framework for CMB detection in volumetric data. The *SWI-CMB* includes 320 SWI images acquired from a 3.0T Philips Medical System with 3D spoiled gradient-echo sequence using venous blood oxygen level dependent series with the following parameters: repetition time 17 ms, echo time 24 ms, volume size  $512 \times 512 \times 150$ , in-plane resolution  $0.45 \times 0.45$  mm, slice thickness 2 mm, slice spacing 1 mm and a  $230 \times 230$  mm<sup>2</sup> field of view. The subjects came from two separated groups: 126 subjects with stroke (mean age  $\pm$  standard deviation,  $67.4 \pm 11.3$ ) and 194 subjects of normal aging (mean age  $\pm$  standard deviation,  $71.2 \pm 5.0$ ). The dataset was labeled by an experienced rater and was verified by a neurologist following the guidance of Microbleed Anatomical Rating Scale [50]. Overall, a total of 1149 CMBs were annotated from the whole dataset and regarded as the ground truth in our experiments. We randomly divided the whole dataset into three sections for training (230 subjects), validation (40 subjects), and testing (50 subjects), respectively. In the preprocessing step, we normalized the volume intensities to the range of [0, 1] after trimming the top 1% gray-scale values [31].

**Table 6.6** Comparison of different screening methods

Methods	Recall	FP <sub>avg</sub>	Time/subject (s)
Barnes et al. [49]	85.47%	2548.2	81.46
Chen et al. [31]	90.48%	935.8	<b>12.00</b>
3D FCN model	<b>98.29%</b>	<b>282.8</b>	64.35

#### 6.4.4 QUANTITATIVE EVALUATION AND COMPARISON

##### 6.4.4.1 CMB Candidate Localization

Three commonly used metrics were employed to quantitatively evaluate the performance of the proposed CMB detection method including recall, precision, and the average number of false positives per subject (FP<sub>avg</sub>).

We compared the screening performance of our proposed method with two state-of-the-art approaches which utilize low-level statistical features [49,31]. The results are listed in [Table 6.6](#). The values of sensitivity mean the percentage of successfully retrieved CMBs while the values of FP<sub>avg</sub> describe the number of remaining false positives per subject. The fewer false positives produced, the more powerful discrimination capability a screening method has. The proposed 3D FCN model achieves the highest recall with fewest average number of false positives, which highlights the efficacy of the proposed method. Note that our method outperforms the other two methods by a large margin, thanks to the 3D FCN model.

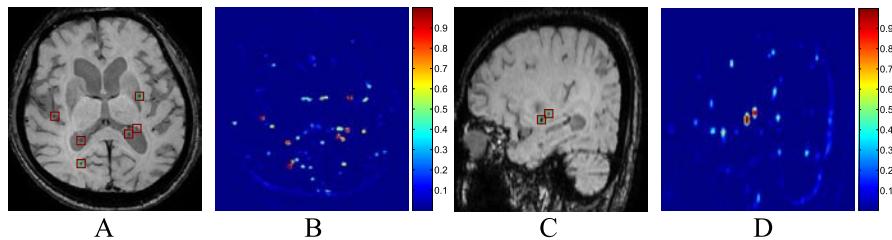
We have also recorded the average time for screening each subject and the results are shown in [Table 6.6](#). From the clinical perspective, the time performance of our method is satisfactory; processing a whole volume with a size of  $512 \times 512 \times 150$  takes around 1 min in our experiments. The method of [49] is slower than ours because it calculates local thresholds using a voxel-wise sliding window strategy. In contrast, the method of [31] merely exploits global thresholding on intensity and size, hence it has a much faster screening speed.

For the candidate screening stage, the retrieval accuracy is vitally important, because we cannot re-find the CMBs that are missed by the screening stage in the following discrimination stage. Although [31] is faster, we achieved around 8% increase in recall and reduced the number of FP<sub>avg</sub> from 935.8 to 282.8, when compared with this method. These results provide a much more reliable basis for further fine discrimination. By employing the 3D FCN, our method achieves a good balance between retrieval accuracy and speed.

Typical candidate screening results by the proposed 3D FCN are shown in [Fig. 6.5](#). It is observed that high values on the score volume mostly correspond to CMB lesions. In addition, most of the backgrounds have been successfully suppressed as zeros. After thresholding, only a small number of candidates are obtained, which dramatically reduces the computational workload in the following stage.

##### 6.4.4.2 True CMB Discrimination

Employing the CMB samples with augmentations and the false positives generated from the training set in the screening stage, we trained the 3D CNN discrimination

**FIGURE 6.5**

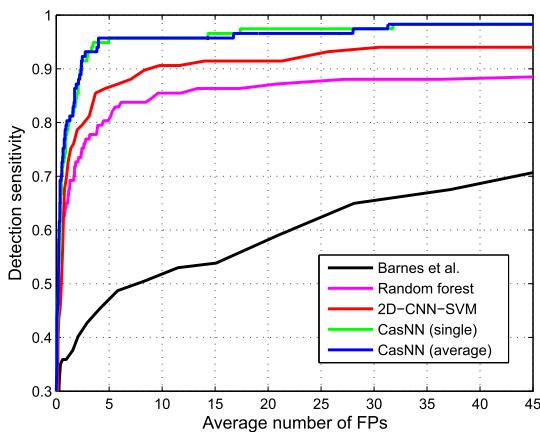
Example results of the 3D FCN screening model. (A), (C) Raw data with true CMBs (green rectangles) and screened candidates (red rectangles) in the axial and sagittal plane, respectively. (B), (D) Corresponding 2D projection of the score volume generated with 3D FCN.

**Table 6.7** Evaluation of detection results on SWI-CMB dataset

Methods	Recall	Precision	FP <sub>avg</sub>
Barnes et al. [49]	64.96%	5.13%	28.10
Random forest [51]	85.47%	17.24%	9.60
2D-CNN-SVM [31]	88.03%	22.69%	7.02
CasNN (single)	92.31%	42.69%	2.90
CasNN (average)	<b>93.16%</b>	<b>44.31%</b>	<b>2.74</b>

model to remove false positive candidates and accurately identify true CMBs. We independently trained three models using the network architecture shown in [Table 6.5](#). The differences of the three models lie in the random weights initialization states and the number of training epochs. This is because that the neural network with a large number of parameters is usually with a low bias and a high variance. By averaging multiple models with different training conditions, we can reduce the model variance, and therefore further boost the discrimination capability [24]. As shown in the last two rows of results in [Table 6.7](#), the performance by averaging the three models was better than that of a single 3D CNN model.

We compared the performance of our method with three typical approaches. These methods were implemented on our dataset for direct comparison. The first one employed handcrafted features based on shape and intensity [49]. We employed its feature extraction procedure on our dataset and utilized a support vector machine (SVM) [52] classifier for prediction. The second one constructed a random forest (RF) classifier based on low-level features, which is commonly used for 3D object detection tasks in medical applications [51]. Specifically, we extracted a total of 159 intensity and geometry-based features. The RF classifier used 500 trees and the maximum depth of each tree is 10. The third one extracted slices from cubic regions as input to the conventional 2D CNN and concatenated the learned features as 3D representations, following a SVM classifier to predict labels [31]. We followed the same network architecture as [31], which is hereinafter referred as 2D-CNN-SVM.

**FIGURE 6.6**

Comparison of FROC curves of different methods.

Table 6.7 shows the comparison results of different methods and the FROC curves are presented in Fig. 6.6. It is clearly observed that our methods outperform the other three comparison methods by a large margin with the highest detection recall and the lowest false positive rates. Although the 2D-CNN-SVM method did not sufficiently leverage the 3D spatial characteristics of the CMBs, the high-level features even with limited spatial information obtained better performance than the other two methods employing low-level features. The comparison results between our methods and the 2D-CNN-SVM method demonstrate that our framework benefits from the high-level representations which can encode richer spatial information by leveraging the 3D convolutional architectures.

#### 6.4.5 SYSTEM IMPLEMENTATION

The proposed framework was implemented based on Theano library [53] using dual Intel Xeon(R) processors E5-2650 2.6 GHz and a GPU of NVIDIA GeForce GTX TITAN Z. The networks were trained with the following hyper-parameters: learning rate = 0.03, momentum = 0.9, dropout rate = 0.3, batch size = 100. The weights of network were randomly initialized from the Gaussian distribution ( $\mu = 0, \sigma = 0.01$ ) and trained with standard back-propagation.

---

## 6.5 DISCUSSION AND CONCLUSION

Detecting objects from medical images that provide clinical significance has been an indispensable component in computer-aided diagnosis. In order to save doctors from backbreaking labor and improve diagnosis reliability as well as efficiency, we pro-

pose an efficient and robust framework for automatic detection of sparsely distributed objects from high-resolution images/volumes. With the design of deep cascaded networks, we keep two aims in mind, namely, efficiency and accuracy. For an automatic object detection system targeting clinical practice, we believe that both of them are equally crucial. In our framework, the first stage efficiently screens the whole image and retrieves a number of potential candidates with a high recall. It can not only speed up the discrimination procedure but also assist the non-experienced trainees by timely promoting the candidates for a closer inspection. The second stage robustly discriminates the true detection targets with only few false positives generated, which can facilitate further quantification measurements. Extensive experimental results on both 2D/3D typical applications corroborated the efficacy and efficiency of our approach, outperforming other state-of-the-art methods by a significant margin.

In medical image computing tasks, especially for volumetric image processing, 3D CNNs hold promising potentials but have not been fully explored yet. Most previous approaches adapted 2D CNN variants for processing 3D volumetric data [54], with difficulties being reported when attempting to employ 3D CNNs. One main concern for employing 3D CNNs is their high computational cost in a sliding window way. In order to address this problem in the detection task, we propose a fast way to narrow down the search range to a limited number of candidates by employing the 3D FCN, which can eliminate the redundant convolutional computations during the forward propagation.

The proposed method is inherently generic and can be easily adapted to different detection tasks in 2D and 3D medical data. Future investigations include assessing the proposed method on more detection tasks and accelerating the algorithm with GPU optimization.

---

## ACKNOWLEDGEMENTS

This work is supported by Hong Kong RGC General Research Fund (No. CUHK412513) and Shenzhen–Hong Kong Innovation Circle Funding Program (No. SGLH20131010151755080 and GHP/002/13SZ). The authors gratefully thank Dr. Ludovic Roux for providing mitosis datasets and helping with the evaluation.

---

## REFERENCES

1. Kunio Doi, Computer-aided diagnosis in medical imaging: historical review, current status and future potential, *Comput. Med. Imaging Graph.* 31 (4) (2007) 198–211.
2. Mitko Veta, Paul J. van Diest, Stefan M. Willems, Haibo Wang, Anant Madabhushi, Angel Cruz-Roa, Fabio Gonzalez, Anders B.L. Larsen, Jacob S. Vestergaard, Anders B. Dahl, et al., Assessment of algorithms for mitosis detection in breast cancer histopathology images, *Med. Image Anal.* 20 (1) (2015) 237–248.

3. Humayun Irshad, Antoine Veillard, Ludovic Roux, Daniel Racoceanu, Methods for nuclei detection, segmentation, and classification in digital histopathology: a review – current status and future potential, *IEEE Rev. Biomed. Eng.* 7 (2014) 97–114.
4. Qi Dou, Hao Chen, Lequan Yu, Lei Zhao, Jing Qin, Defeng Wang, Vincent C.T. Mok, Lin Shi, Pheng-Ann Heng, Automatic detection of cerebral microbleeds from MR images via 3D convolutional neural networks, *IEEE Trans. Med. Imaging* 35 (5) (2016) 1182–1195.
5. Bram van Ginneken, Samuel G. Armato, Bartjan de Hoop, Saskia van Amelsvoort-van de Vorst, Thomas Duindam, Meindert Niemeijer, Keelin Murphy, Arnold Schilham, Alessandra Retico, Maria Evelina Fantacci, et al., Comparing and combining algorithms for computer-aided detection of pulmonary nodules in computed tomography scans: the ANODE09 study, *Med. Image Anal.* 14 (6) (2010) 707–722.
6. Tao Chan, Computer aided detection of small acute intracranial hemorrhage on computer tomography of brain, *Comput. Med. Imaging Graph.* 31 (4) (2007) 285–298.
7. Angel Cruz-Roa, Gloria Díaz, Eduardo Romero, Fabio A. González, Automatic annotation of histopathological images using a latent topic model based on non-negative matrix factorization, *J. Pathol. Inform.* 2 (2011).
8. Andreas Charidimou, David J. Werring, Cerebral microbleeds: detection, mechanisms and clinical challenges, *Future Neurol.* 6 (5) (2011) 587–611.
9. M.W. Vernooij, Aad van der Lugt, Mohammad Arfan Ikram, P.A. Wielopolski, W.J. Niessen, Albert Hofman, G.P. Krestin, M.M.B. Breteler, Prevalence and risk factors of cerebral microbleeds: the Rotterdam scan study, *Neurology* 70 (14) (2008) 1208–1214.
10. Steven M. Greenberg, Meike W. Vernooij, Charlotte Cordonnier, Anand Viswanathan, Rustam Al-Shahi Salman, Steven Warach, Lenore J. Launer, Mark A. Van Buchem, Monique Breteler, Cerebral microbleeds: a guide to detection and interpretation, *Lancet Neurol.* 8 (2) (2009) 165–174.
11. Ross Girshick, Jeff Donahue, Trevor Darrell, Jitendra Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 580–587.
12. Jonathan Long, Evan Shelhamer, Trevor Darrell, Fully convolutional networks for semantic segmentation, in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3431–3440.
13. Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton, ImageNet classification with deep convolutional neural networks, in: *Advances in Neural Information Processing Systems*, 2012, pp. 1097–1105.
14. Karen Simonyan, Andrew Zisserman, Very deep convolutional networks for large-scale image recognition, *arXiv:1409.1556*, 2014.
15. Hao Chen, Qi Dou, Xi Wang, Jing Qin, Pheng Ann Heng, Mitosis detection in breast cancer histology images via deep cascaded networks, in: *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.
16. Humayun Irshad, Automated mitosis detection in histopathology using morphological and multi-channel statistics features, *J. Pathol. Inform.* 4 (2013).
17. Christopher D. Malon, Eric Cosatto, Classification of mitotic figures with convolutional neural networks and seeded blob features, *J. Pathol. Inform.* 4 (2013).
18. Haibo Wang, Angel Cruz-Roa, Ajay Basavanhally, Hannah Gilmore, Natalie Shih, Mike Feldman, John Tomaszewski, Fabio Gonzalez, Anant Madabhushi, Cascaded ensemble of convolutional neural networks and handcrafted features for mitosis detection, in: *SPIE Medical Imaging*, International Society for Optics and Photonics, 2014, 90410B.

19. F. Boray Tek, Mitosis detection using generic features and an ensemble of cascade Adaboosts, *J. Pathol. Inform.* 4 (2013).
20. Alessandro Giusti, Dan C. Cireşan, Jonathan Masci, Luca M. Gambardella, Jürgen Schmidhuber, Fast image scanning with deep max-pooling convolutional neural networks, *arXiv:1302.1700*, 2013.
21. Dan C. Cireşan, Alessandro Giusti, Luca M. Gambardella, Jürgen Schmidhuber, Mitosis detection in breast cancer histology images with deep neural networks, in: *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2013*, Springer, 2013, pp. 411–418.
22. Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, Trevor Darrell, Caffe: convolutional architecture for fast feature embedding, *arXiv:1408.5093*, 2014.
23. Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, Ruslan R. Salakhutdinov, Improving neural networks by preventing co-adaptation of feature detectors, *arXiv:1207.0580*, 2012.
24. Stuart Geman, Elie Bienenstock, René Doursat, Neural networks and the bias/variance dilemma, *Neural Comput.* 4 (1) (1992) 1–58.
25. Peter Boyle, Bernard Levin, et al., *World Cancer Report 2008*, IARC Press, International Agency for Research on Cancer, 2008.
26. C.W. Elston, I.O. Ellis, Pathological prognostic factors in breast cancer. I. The value of histological grade in breast cancer: experience from a large study with long-term follow-up, *Histopathology* 19 (5) (1991) 403–410.
27. Hao Chen, Chiyao Shen, Jing Qin, Dong Ni, Lin Shi, Jack C.Y. Cheng, Pheng-Ann Heng, Automatic localization and identification of vertebrae in spine CT via a joint learning model with deep neural networks, in: *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, Springer, 2015, pp. 515–522.
28. Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al., ImageNet large scale visual recognition challenge, *Int. J. Comput. Vis.* (2014) 1–42.
29. Jason Yosinski, Jeff Clune, Yoshua Bengio, Hod Lipson, How transferable are features in deep neural networks?, in: *Advances in Neural Information Processing Systems*, 2014, pp. 3320–3328.
30. Hao Chen, Dong Ni, Jing Qin, Shengli Li, Xin Yang, Tianfu Wang, Pheng Ann Heng, Standard plane localization in fetal ultrasound via domain transferred deep neural networks, *IEEE J. Biomed. Health Inform.* 19 (5) (2015) 1627–1636.
31. Hao Chen, Lequan Yu, Qi Dou, Lin Shi, Vincent C.T. Mok, Pheng Ann Heng, Automatic detection of cerebral microbleeds via deep learning based 3D feature representation, in: *2015 IEEE 12th International Symposium on Biomedical Imaging (ISBI)*, IEEE, 2015, pp. 764–767.
32. Ludovic Roux, Daniel Racoceanu, Nicolas Loménie, Maria Kulikova, Humayun Irshad, Jacques Klossa, Frédérique Capron, Catherine Genestie, Gilles Le Naour, Metin N. Gurcan, Mitosis detection in breast cancer histological images: an ICPR 2012 contest, *J. Pathol. Inform.* 4 (2013).
33. Andreas Charidimou, Anant Krishnan, David J. Werring, H. Rolf Jäger, Cerebral microbleeds: a guide to detection and clinical relevance in different disease settings, *Neuroradiology* 55 (6) (2013) 655–674.
34. Andreas Charidimou, David J. Werring, Cerebral microbleeds and cognition in cerebrovascular disease: an update, *J. Neurol. Sci.* 322 (1) (2012) 50–55.

35. Amir Fazlollahi, Fabrice Meriaudeau, Victor L. Villemagne, Christopher C. Rowe, Paul Yates, Olivier Salvado, Pierrick Bourgeat, Efficient machine learning framework for computer-aided detection of cerebral microbleeds using the radon transform, in: 2014 IEEE 11th International Symposium on Biomedical Imaging (ISBI), IEEE, 2014, pp. 113–116.
36. Hugo J. Kuijf, Jeroen de Bresser, Mirjam I. Geerlings, Mandy Conijn, Max A. Viergever, Geert Jan Biessels, Koen L. Vincken, Efficient detection of cerebral microbleeds on 7.0 T MR images using the radial symmetry transform, *NeuroImage* 59 (3) (2012) 2266–2273.
37. Wei Bian, Christopher P. Hess, Susan M. Chang, Sarah J. Nelson, Janine M. Lupo, Computer-aided detection of radiation-induced cerebral microbleeds on susceptibility-weighted MR images, *NeuroImage Clin.* 2 (2013) 282–290.
38. Babak Ghafaryasl, Fedde van der Lijn, Marielle Poels, Henri Vrooman, M. Arfan Ikram, Wiro J. Niessen, Aad van der Lugt, Meike Vernooij, Marleen de Bruijne, A computer aided detection system for cerebral microbleeds in brain MRI, in: 2012 9th IEEE International Symposium on Biomedical Imaging (ISBI), IEEE, 2012, pp. 138–141.
39. Qi Dou, Hao Chen, Lequan Yu, Lin Shi, Defeng Wang, Vincent C.T. Mok, Pheng Ann Heng, Automatic cerebral microbleeds detection from MR images via independent subspace analysis based hierarchical features, in: Engineering in Medicine and Biology Society (EMBC), 2015 37th Annual International Conference of the IEEE, IEEE, 2015, pp. 7933–7936.
40. Hao Chen, Qi Dou, Dong Ni, Jie-Zhi Cheng, Jing Qin, Shengli Li, Pheng-Ann Heng, Automatic fetal ultrasound standard plane detection using knowledge transferred recurrent neural networks, in: Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015, Springer, 2015, pp. 507–514.
41. Nima Tajbakhsh, Jae Y. Shin, Suryakanth R. Gurudu, R. Todd Hurst, Christopher B. Kendall, Michael B. Gotway, Jianming Liang, Convolutional neural networks for medical image analysis: full training or fine tuning?, *IEEE Trans. Med. Imaging* 35 (5) (2016) 1299–1312.
42. Hao Chen, Xiao Juan Qi, Jie Zhi Cheng, Pheng Ann Heng, Deep contextual networks for neuronal structure segmentation, in: Thirtieth AAAI Conference on Artificial Intelligence, 2016.
43. Mohammad Havaei, Axel Davy, David Warde-Farley, Antoine Biard, Aaron Courville, Yoshua Bengio, Chris Pal, Pierre-Marc Jodoin, Hugo Larochelle, Brain tumor segmentation with deep neural networks, *arXiv:1505.03540*, 2015.
44. Holger R. Roth, Christopher T. Lee, Hoo-Chang Shin, Ari Seff, Lauren Kim, Jianhua Yao, Le Lu, Ronald M. Summers, Anatomy-specific classification of medical images using deep convolutional nets, in: 2015 IEEE 12th International Symposium on Biomedical Imaging (ISBI), IEEE, 2015, pp. 101–104.
45. Holger R. Roth, Le Lu, Amal Farag, Hoo-Chang Shin, Jiamin Liu, Evrim B. Turkbey, Ronald M. Summers, DeepOrgan: multi-level deep convolutional networks for automated pancreas segmentation, in: Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015, Springer, 2015, pp. 556–564.
46. Adhish Prasoon, Kersten Petersen, Christian Igel, François Lauze, Erik Dam, Mads Nielsen, Deep feature learning for knee cartilage segmentation using a triplanar convolutional neural network, in: Medical Image Computing and Computer-Assisted Intervention – MICCAI 2013, Springer, 2013, pp. 246–253.
47. Holger R. Roth, Le Lu, Ari Seff, Kevin M. Cherry, Joanne Hoffman, Shijun Wang, Jiamin Liu, Evrim Turkbey, Ronald M. Summers, A new 2.5 D representation for lymph node de-

- tection using random sets of deep convolutional neural network observations, in: Medical Image Computing and Computer-Assisted Intervention – MICCAI 2014, Springer, 2014, pp. 520–527.
- 48. Xavier Glorot, Antoine Bordes, Yoshua Bengio, Deep sparse rectifier neural networks, in: International Conference on Artificial Intelligence and Statistics, 2011, pp. 315–323.
  - 49. Samuel R.S. Barnes, E. Mark Haacke, Muhammad Ayaz, Alexander S. Boikov, Wolff Kirsch, Dan Kido, Semiautomated detection of cerebral microbleeds in magnetic resonance images, *Magn. Reson. Imaging* 29 (6) (2011) 844–852.
  - 50. S.M. Gregoire, U.J. Chaudhary, M.M. Brown, T.A. Yousry, C. Kallis, H.R. Jäger, D.J. Werring, The microbleed anatomical rating scale (MARS) reliability of a tool to map brain microbleeds, *Neurology* 73 (21) (2009) 1759–1766.
  - 51. Andy Liaw, Matthew Wiener, Classification and regression by randomForest, *R News* 2 (3) (2002) 18–22, <http://CRAN.R-project.org/doc/Rnews/>.
  - 52. Chih-Chung Chang, Chih-Jen Lin, LIBSVM: a library for support vector machines, *ACM Trans. Intell. Syst. Technol.* 2 (2011) 27:1–27:27, software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>.
  - 53. Theano Development Team, Theano: a Python framework for fast computation of mathematical expressions, arXiv:1605.02688, 2016.
  - 54. Hayit Greenspan, Bram van Ginneken, Ronald M. Summers, Guest editorial deep learning in medical imaging: overview and future promise of an exciting new technique, *IEEE Trans. Med. Imaging* 35 (5) (2016) 1153–1159.

---

## NOTES

1. <http://ipal.cnrs.fr/ICPR2012/>.
2. <http://mitos-atypia-14.grand-challenge.org/>.

# Deep Voting and Structured Regression for Microscopy Image Analysis

7

**Yuanpu Xie, Fuyong Xing, Lin Yang***University of Florida, Gainesville, FL, United States*

## CHAPTER OUTLINE

<b>7.1 Deep Voting: A Robust Approach Toward Nucleus Localization in Microscopy Images.....</b>	156
7.1.1 Introduction .....	156
7.1.2 Methodology .....	158
7.1.2.1 <i>Learning the Deep Voting Model</i> .....	159
7.1.3 Weighted Voting Density Estimation.....	162
7.1.4 Experiments .....	163
7.1.5 Conclusion .....	165
<b>7.2 Structured Regression for Robust Cell Detection Using Convolutional Neural Network .....</b>	165
7.2.1 Introduction .....	165
7.2.2 Methodology .....	166
7.2.3 Experimental Results .....	169
7.2.4 Conclusion .....	171
<b>Acknowledgements .....</b>	172
<b>References.....</b>	172

## CHAPTER POINTS

- In this chapter, we present two novel frameworks for robust cell detection in microscopy images
- The presented methods provide two potential views toward object detection in a crowd scenario. Both methods go beyond the traditional sliding window based classification method
- We demonstrate that the convolutional neural network is a potentially powerful tool, which still needs future exploitation in biomedical image analysis

---

## 7.1 DEEP VOTING: A ROBUST APPROACH TOWARD NUCLEUS LOCALIZATION IN MICROSCOPY IMAGES

### 7.1.1 INTRODUCTION

Accurate localization of nucleus centers in microscopy images is fundamental to many subsequent computer-aided biomedical image analysis such as cell segmentation, cell counting, tracking, etc. Unfortunately, robust nucleus localization, especially in microscopy images exhibiting dense nucleus clutters and large variations in terms of both nucleus sizes and shapes, has proven to be extremely challenging [1]. In the past few years, a large number of methods have been proposed, spatial filters [2] have been applied for automatic nuclei localization, and graph partition methods have been reported in [3,4] to automatically detect cells. Since cells are approximately circular, Parvin et al. [5] have introduced a kernel based radial voting method to iteratively localize cells, which is insensitive to image noises. Several other radial voting-based methods are presented in [6–9] for automatic cell detection on histopathology images. Other localization methods based on gradient vector [10], Laplacian-of-Gaussian (LOG) with adaptive scale selection [2,11] and concave points [12,13] can also be found in the literature. However, because of the large variations in microscopy modality, nucleus morphology, and the inhomogeneous background, it remains to be a challenging topic for these non-learning methods.

Supervised learning based methods have also attracted a lot of interest due to their promising performance. For example, a general and efficient maximally stable extremal region (MSER) section method is presented in [14], in which MSER is used to efficiently generate region candidates of possible cell regions, a structured SVM [15] is then utilized to score each region, and finally the configurations of the selected regions are achieved using dynamic programming to explore the tree structure of MSER. However, in images that exhibit strong inhomogeneous background and foreground differences, the MSER detector cannot generate feasible region candidates, and thus its usage is limited. Recently, in computer vision community, codebook based hough transformation, such as implicit shape model [16], has also been widely studied and shown to be able to produce promising performance in general object detection tasks. Random forest method [17] or singular vector decomposition [18] is applied to learn a discriminative codebook. The so-called class-specific hough forest is able to directly map the image patches to the probabilistic votes and to the potential locations of the object center. However, it only associates one voting offset vector to an image patch during the training process, and in the testing stage, one patch can only vote along the directions that have already appeared in the training data. In addition, this method is not designed to detect objects that appear as a crowd, which is often the case when tackling nuclei localization in microscopy images.

Despite their success, the performance of aforementioned works heavily rely on various handcrafted features such as shapes, gradients, colors, etc. Fusion strategy is often used to combine those features for better results [19]. Such methods may work well on some specific types of microscopic images but will generally fail on others. To alleviate this problem and liberate the user from choosing proper handcrafted

features, the deep learning technique is adopted in the proposed method to free the user from laborious feature selections and parameter tuning [20–23]. Deep learning is a revived topic in recent years and achieved outstanding performance in both natural image analysis and biomedical image analysis [24–27]. Local features are typically extracted automatically from image patches via a cascade of convolutional neuron networks (CNNs). These CNNs are usually connected with fully connected networks, and the training label information is fed back to update the weights of CNNs via backpropagation [28].

Numerous works based on CNN structure can be found in the literature, with their applications ranging from fundamental low-level image processing to more sophisticated vision tasks. Farabet et al. [29] exploit a multi-scale strategy to extract hierarchical features for scene labeling task. Krizhevsky et al. [30] provide a GPU implementation of CNN and achieve record-breaking performance in ImageNet classification tasks. Besides classification, CNN has also been used as a regressor, and it is shown to be able to capture geometric information exhibited in the image. Toshev and Szegedy [31] adopt a similar architecture as that in [30] and achieve state-of-the-art performance of human pose estimation by directly doing regression on the human joints positions. A mask regression method has been proposed to do object detection in [32]. Other successful applications of deep learning in nature image analysis are reported in [32,33].

The CNN structure based learning framework has been proven successful in various biomedical image analysis tasks. Ciresan et al. [34] exploit a convolutional neural network framework to detect mitosis in breast cancer histology images. Another similar method is reported in [35] for membrane neuronal segmentation in a microscopy image. Other successful applications of deep learning in medical image analysis include Multi-Modality Isointense Infant Brain Image Segmentation [36] and predicting Alzheimer’s disease using 3D convolutional neural networks [37]. Further successful applications of CNN are reported in [38–40]. It is also worth noting that image representation of biomedical images learned by convolutional neural network performs better than the handcrafted features [41–44,26]. Variations of 2D recurrent neural network [45,46] and the recently proposed fully convolutional neural network [47] have also achieved promising results in medical image analysis [48,49].

In this chapter, we present a novel deep neural network aiming at robust nucleus localization in microscopy images. Unlike traditional methods that either take advantage of CNN as a pixel-wise classifier or as a regressor on the entire image feature scope, we improve the CNN structure by introducing a hybrid nonlinear transformation capable of learning both the voting offset vectors and corresponding voting confidence jointly for robust center localization. Unlike [17,33], this method assigns more than one pair of predictions for each image patch, which, together with the specific loss function, render this method more robust and capable of handling touching cases. This model, which is named a deep voting model, can be viewed as an implicit hough-voting codebook [50], using which every testing patch votes toward several directions with specific confidence. Given a test image, we collect all the votes with quantized confidence cumulatively and estimate the voting density map in

a way similar to Parzen-window estimation [51]. Although this method is inherently suited for general object detection tasks, we focus on nuclei center localization in this work. Comparative experimental results show superiority of the proposed deep voting method.

### 7.1.2 METHODOLOGY

Before going into details of our proposed model, we start with a brief introduction to the widely used CNN model. A CNN framework typically contains a stack of alternating convolutional layers (C) and pooling layers (P), followed by one or more fully connected layers (FC). Convolutional layer serves the purpose to extract patterns by convolving the filter banks on the input images (or feature maps). The linear combination of the convolutional outputs are passed to a nonlinear activation function (logistic, ReLu, etc.) to generate the output feature maps. Define  $L$  as the total number of layers, with the input layer as the first layer. Define  $\mathbf{f}_i^l$  as the  $i$ th feature map in  $l$ th layer. In general, we have

$$\mathbf{f}_i^l = h\left(\sum_j \mathbf{f}_j^{l-1} * \mathbf{k}_{ji}^l + \mathbf{b}_i^l\right), \quad (7.1)$$

where  $\mathbf{b}_i^l$  is the bias,  $\mathbf{k}_{ji}^l$  is the convolution kernel that works on the input feature maps in layer  $l - 1$ , and  $h$  is a nonlinear activation function.

Pooling layer (P) performs down-sampling of the input feature maps by a predefined factor, the number of output feature maps is exactly the same as that of the input maps. One of the most widely used pooling layers is called a max-pooling layer, it is used to reduce the size of the feature maps and introduce local shift and translation invariance properties to the network at the same time.

Fully connected layer (FC) takes all the outputs from the previous layer and connects them to every unit in this layer. If the previous layer is a convolutional layer and consists of 2D feature maps, the conventional way is to reshape those feature maps into one feature vector. This layer is used to perform high-level inference by multiplying the input vector with the weight matrix and following by a nonlinear transformation. Actually, this type of layer can also be viewed as a special case of convolutional layer with filter size  $1 \times 1$ . More formally, denote by  $\mathbf{W}^l$  the weights for  $l$ th layer, the output of this layer can be given by

$$\mathbf{f}_i^l = h(\mathbf{W}^l \mathbf{f}_i^{l-1} + \mathbf{b}_i^l). \quad (7.2)$$

When CNN is applied as a classifier, softmax is used at the top layer to encode the probability distribution of a certain class denoted by the corresponding unit. Let  $\mathbf{h}$  denote the activation function of the nodes in penultimate layer, and denote by  $\mathbf{W}$  the weight connecting the last layer and the penultimate layer. If the total input to one node of the softmax layer is  $a_j = \sum_k h_k W_{kj}$ , then we can have  $p_j = \frac{\exp(a_j)}{\sum \exp(a_j)}$ , and the class label can be obtained by  $\hat{i} = \arg \max_i p_j$ .

Denote by  $E$  the final loss function defined on a set of training data. Let  $\Theta$  represent the model's parameters composed of the kernel weights and biases. The model can be learned iteratively using a stochastic gradient descent algorithm of the following form:

$$\Theta(t) = \Theta(t-1) - \mu \frac{\partial E}{\partial \Theta}, \quad (7.3)$$

where  $\Theta(t)$  represents the model's parameter in the  $t$ th iteration and  $\mu$  denotes the learning rate. In practice, some optimization and regularization tricks, including normalization and shuffling of the training data, adaptive learning rate, momentum, and dropout, can be used to improve the performance.

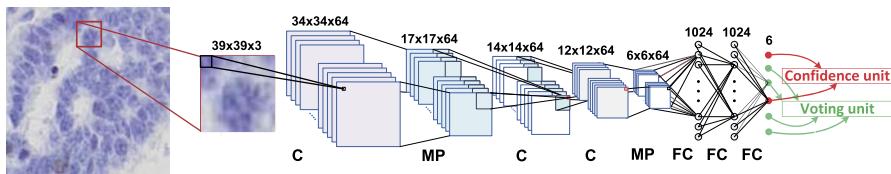
### 7.1.2.1 Learning the Deep Voting Model

**Preprocessing.** In the setting of this method, each image patch votes for several possible nucleus positions (the coordinates for the nucleus center are specified by voting offsets) with several specific voting confidence, a high voting confidence indicates that this image patch is highly informative to the corresponding vote. Please note that the number of votes is predefined. In this work, we propose to learn an implicit codebook using CNN for these voting offsets and corresponding voting confidence for the testing image patches.

Denote by  $\mathcal{P}$  the input image patch space composed of a set of multi-channel squared image patches of size  $d \times d \times c$ , where  $d$  and  $c$  represent the patch height/width and the number of image channels, respectively; and by  $\mathcal{T}$  the target space, defining the space of the proposed *target information*. We define the *target information* as the coalescence of voting offsets and voting confidence. For each training image patch, we first obtain the coordinates of  $k$  nucleus centers from a group of human annotation (ground-truth) that are closest to the center of the image patch. Let  $\{\mathcal{D}^i = (\mathcal{P}^i, \mathcal{T}^i)\}$  represent the  $i$ th training data where  $\mathcal{P}^i \in \mathcal{P}$  is the  $i$ th input image patch and  $\mathcal{T}^i \in \mathcal{T}$  is the corresponding *target information*.  $\mathcal{T}^i$  can be further defined as  $\{\alpha_j^i, \mathbf{v}_j^i\}_{j=1}^k$ , where  $k$  denotes the number of total votes from each image patch and  $\mathbf{v}_j^i$  is the 2D offset vector that is equal to the displacement from the coordinate of the  $j$ th nearest ground-truth nucleus aligned to the center of patch  $\mathcal{P}^i$ .  $\alpha_j^i$  represents the voting confidence corresponding to  $\mathbf{v}_j^i$ . In this work, we define it based on the length of the voting offset  $|\mathbf{v}_j^i|$  as

$$\alpha_j^i = \begin{cases} 1, & \text{if } |\mathbf{v}_j^i| \leq r_1, \\ \frac{1}{1+\beta|\mathbf{v}_j^i|}, & \text{if } r_1 < |\mathbf{v}_j^i| \leq r_2, \\ 0, & \text{otherwise,} \end{cases} \quad (7.4)$$

where  $\beta$  represents the confidence decay ratio.  $r_1, r_2$  are used to tune the voting range and are chosen as  $d$  and  $1.5d$ , respectively. Please note that (7.4) is only one of the possible definitions of  $\alpha_j^i$ , instead, it can be defined based on many interesting

**FIGURE 7.1**

The architecture of the proposed deep voting model. The C, MP, and FC represent the convolutional layer, max-pooling layer, and fully connected layer, respectively. The two different types of units (voting units and weight units) in the last layer are marked with different color. Please note that in this model the number of votes for each patch ( $k$ ) is 2.

properties of the training patches such as foreground area ratio, probability of being a specific class membership, distance from the patch center to the voting position, etc.

**Hybrid Nonlinear Transformation.** In order to jointly learn the voting offset vectors and voting confidence, we categorize the units of the output layer in the deep voting model as *voting units* and *confidence units*. As is shown in Fig. 7.1, we use two different colors to differentiate two types of units in the last layer: red nodes represent the confidence units and green units the voting units.

Since the voting offset is 2-dimensional, every voting confidence unit corresponds to 2 voting offset units. Observe that voting units can take any values and are used to specify the positions that each patch will vote to. The values for confidence units are confined to  $[0, 1]$  and are used as weights for the votes from every testing image patch. Existing activation functions (sigmoid, linear, or ReLu) associated to the output layer treat all the units in the same way and do not satisfy our needs. In order to jointly encode the geometric voting offset and the voting confidence information in the last layer of the model, we therefore introduce a **hybrid nonlinear transformation**  $Hy$  as the new activation function. Units of the output layer are treated differently based on their specific categories. For voting units, we use a simple linear activation function, for confidence units, we use the sigmoid activation function. Therefore,  $Hy$  is given by

$$Hy(x) = \begin{cases} \text{sigm}(x), & \text{if } x \text{ is the input to the } \textit{confidence units}, \\ x, & \text{otherwise,} \end{cases} \quad (7.5)$$

where  $\text{sigm}$  denotes the sigmoid function. Note that, similar to sigmoid activation in neural networks,  $Hy$  here is a point-wise operator. Alternatively,  $Hy$  can be constructed using other type of activation functions based on specific needs, e.g., ReLu, tanh, etc.

**Inference in Deep Voting Model.** In this method, we adopt CNN architecture during the codebook learning as shown in Fig. 7.1. Denote by  $L$  the number of layers, let functions  $f_1, \dots, f_L$  represent each layer's computations (e.g., linear transformations

and convolution, etc.), and denote by  $\theta_1, \dots, \theta_L$  the parameters of those layers. Note that pooling layers have no parameters. This model can be viewed as a mapping function,  $\psi$ , which maps input image patches to *target information*. Denote by  $\circ$  the composition of functions, then  $\psi$  can be further defined as  $f_L \circ f_{L-1} \circ \dots \circ f_1$  with parameters  $\Theta = \{\theta_1, \dots, \theta_L\}$ .

Assume we are given the training data  $\{\mathcal{D}^i = (\mathcal{P}^i, \mathcal{T}^i)\}_{i=1}^N$ , where  $N$  is the number of training samples. The parameters of the proposed model can be evaluated by solving the following optimization problem:

$$\arg \min_{\Theta} \frac{1}{N} \sum_{i=1}^N l(\psi(\mathcal{P}^i; \Theta), \mathcal{T}^i), \quad (7.6)$$

where  $l$  is the loss function defined on one training sample. The detailed definition of  $l$  is given in the following paragraphs. Let  $\psi(\mathcal{P}^i; \Theta) = \{w_j^i, d_j^i\}_{j=1}^k$  denote the model's output corresponding to input  $\mathcal{P}^i$ , where  $w_j^i$  and  $d_j^i$  are predicted voting confidence and offsets, respectively. Recall that  $\mathcal{T}^i = \{\alpha_j^i, v_j^i\}_{j=1}^k$  is the *target information* of  $\mathcal{P}^i$ , and  $k$  represents the number of votes from each image patch. The loss function  $l$  defined on  $(\mathcal{P}^i, \mathcal{T}^i)$  can be given by

$$l(\psi(\mathcal{P}^i; \Theta), \mathcal{T}^i) = \sum_{j=1}^k \frac{1}{2} (\alpha_j^i w_j^i \|d_j^i - v_j^i\|^2 + \lambda(w_j^i - \alpha_j^i)^2), \quad (7.7)$$

where  $\lambda$  is a regularization factor used to tune the weights of loss coming from voting confidence and offsets. There are two benefits of the proposed loss function: (i) The  $\alpha_j^i w_j^i$  in the first term penalizes uninformative votes that have either low predicted voting confidence or low voting confidence in the *target information*. This property alleviates the issue of fixed number of votes for each image patches, for uninformative votes, the loss coming from voting offsets vanishes, and this loss function degenerates into a squared error loss defined on the voting confidence. (ii) The second term not only punishes the error coming from voting confidence, it also acts as a regularization term to prevent the network from producing trivial solutions (setting all voting confidence to zero).

We use stochastic gradient descent and backpropagation algorithm [52] to solve (7.6). In order to calculate the gradients of (7.6) with respect to the model's parameters  $\Theta$ , we need to calculate the partial derivatives of the loss function defined on one single example with respect to the inputs of the nodes in the last layer. As shown in Fig. 7.1, the outputs of the proposed model are organized as  $k$  pairs of {confidence units, offset units}. Let  $\{Iw_j^i, Iv_j^i\}_{j=1}^k$  represent inputs to the units in the last layer for  $(\mathcal{P}^i, \mathcal{T}^i)$  in the forward pass, we can easily get the model's output,  $w_j^i = Hy(Iw_j^i)$  and  $v_j^i = Hy(Iv_j^i)$ . The partial derivatives of (7.7) with respect to

$Iw_j^i$  and  $\mathbf{Iv}_j^i$  are then given as

$$\begin{aligned} \frac{\partial l(\psi(\mathcal{P}^i; \Theta), \mathcal{T}^i)}{\partial Iw_j^i} &= \frac{\partial l(\psi(\mathcal{P}^i; \Theta), \mathcal{T}^i)}{\partial w_j^i} \frac{\partial w_j^i}{\partial Iw_j^i} \\ &= (\frac{1}{2}\alpha_j^i \|\mathbf{d}_j^i - \mathbf{v}_j^i\|^2 + \lambda(w_j^i - \alpha_j^i))Iw_j^i(1 - Iw_j^i), \end{aligned} \quad (7.8)$$

$$\begin{aligned} \frac{\partial l(\psi(\mathcal{P}^i; \Theta), \mathcal{T}^i)}{\partial \mathbf{Id}_j^i} &= \frac{\partial \mathbf{d}_j^i}{\partial \mathbf{Id}_j^i} \frac{\partial l(\psi(\mathcal{P}^i; \Theta), \mathcal{T}^i)}{\partial \mathbf{d}_j^i} \\ &= \alpha_j^i w_j^i (\mathbf{d}_j^i - \mathbf{v}_j^i), \end{aligned} \quad (7.9)$$

where  $\mathbf{Id}_j^i$ ,  $\mathbf{d}_j^i$  and  $\frac{\partial l(\psi(\mathcal{P}^i; \Theta), \mathcal{T}^i)}{\partial \mathbf{Id}_j^i}$  are 2D vectors. After getting the above partial derivatives, the following inferences used to obtain the gradients of (7.6) with respect to the model's parameters are exactly the same to the classical backpropagation algorithm used in conventional CNN [52].

### 7.1.3 WEIGHTED VOTING DENSITY ESTIMATION

Given a testing image, denote by  $\mathcal{P}(y)$  an image patch centered at position  $y$  on the testing image,  $y$  is a 2D vector representing the coordinate of the center of testing image patch  $\mathcal{P}(y)$ , and let  $\{w_j(y), \mathbf{d}_j(y)\}_{j=1}^k = \psi(\mathcal{P}(y); \Theta)$  represent the model's output for  $\mathcal{P}(y)$ , where  $w_j(y)$  and  $\mathbf{d}_j(y)$  represent the  $j$ th voting confidence and offset vector for patch  $\mathcal{P}(y)$ , respectively. Furthermore, denote by  $V(x|\mathcal{P}(y))$  the accumulated voting score of position  $x$  coming from patch  $\mathcal{P}(y)$ . Similar to the kernel density estimation using Parzen-window, we can write  $V(x|\mathcal{P}(y))$  as

$$V(x|\mathcal{P}(y)) = \sum_{j=1}^k w_j(y) \frac{1}{2\pi\sigma^2} \exp\left(-\frac{\|(y - \mathbf{d}_j(y)) - x\|^2}{2\sigma^2}\right). \quad (7.10)$$

In this work, we restrict the votes contributing to the location  $x$  to be calculated from a limited area represented by  $B(x)$ , which can be a circle or a bounding box centered at position  $x$ . To compute the final weighted voting density map  $V(x)$ , we accumulate the weighted votes from every testing image patch centered at  $y \in B(x)$ ,

$$V(x) = \sum_{y \in B(x)} V(x|\mathcal{P}(y)). \quad (7.11)$$

After obtaining the voting density map  $V(x)$ , a small threshold  $\xi \in [0, 1]$  is applied to remove the values smaller than  $\xi \cdot \max(V)$ . The following procedure for nucleus localization is to find all the local maximum locations in  $V$ .

In order to reduce the complexity for computing the voting density map in (7.11), we provide a fast implementation: Given a testing image, instead of following the

computation order specified by (7.10) and (7.11), we go through image patch  $\mathcal{P}(\mathbf{y})$  centered at every possible position  $\mathbf{y}$  in the testing image, and add weighted votes  $\{w_j(\mathbf{y})\}_{j=1}^k$  to pixel  $\{\mathbf{y} - \mathbf{d}_j(\mathbf{y})\}_{j=1}^k$  in a cumulative way. The accumulated vote map is then filtered by a Gaussian kernel to get the final voting density map  $V(\mathbf{x})$ .

### 7.1.4 EXPERIMENTS

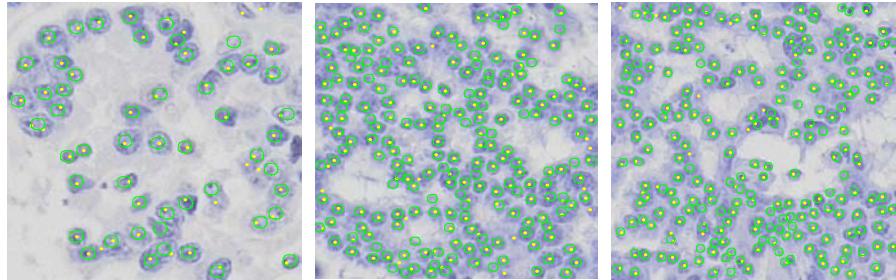
**Data and Implementation Details.** The proposed method has been extensively evaluated using 44 Ki67-stained NET microscopy images. Testing images are carefully chosen to cover challenging cases like touching cells, inhomogeneous intensity, blurred cell boundaries, and weak staining.

The architecture of the model is summarized in Fig. 7.1. The patch size is set to be  $39 \times 39$ . The convolutional kernel sizes are set to be  $6 \times 6$ ,  $4 \times 4$ , and  $3 \times 3$  for the three convolutional layers, respectively. All the max pooling layers have a window size of  $2 \times 2$ , and a stride of 2. A dropout ratio 0.25 is used at the training stage to prevent over fitting and the learning rate is set to be 0.0001.  $\beta$  and  $\lambda$  are set to be 0.5 and 384 in (7.4) and (7.7), respectively. The threshold  $\xi$  is set to be 0.2. The Parzen-window size is  $5 \times 5$ , and  $\sigma$  is 1 in (7.10). These parameters are set either by following conventions from the existing works or by empirical selection. In fact, our method is robust to hyper-parameter selection. The implementation is based on the fast CUDA kernel provided by Krizhevsky [30]. The proposed model is trained and tested using a PC with NVIDIA Tesla K40C GPU and an Intel Xeon E5 CPU.

**Evaluate the Deep Voting Model.** For quantitative evaluation, we define the ground-truth region as the circular regions of radius  $r$  centered at every human annotated nucleus center. In this work,  $r$  is roughly chosen to be half of the average radius of the nucleus. True positive ( $TP$ ) detections are defined as detected results that lie in the ground-truth region. False positive ( $FP$ ) detections refer to detection results that lie outside of the ground-truth region. The human annotated nucleus centers that do not match with any detection result are considered to be false negative ( $FN$ ). Based on  $TP$ ,  $FP$ , and  $FN$ , we can calculate the recall ( $R$ ), precision ( $P$ ), and  $F_1$  scores, which are given by  $R = \frac{TP}{TP+FN}$ ,  $P = \frac{TP}{TP+FP}$ , and  $F_1 = \frac{2PR}{P+R}$ . In addition to  $P$ ,  $R$ , and  $F_1$  score, we also compute the mean  $\mu$  and standard deviation  $\sigma$  of the Euclidean distance (defined as  $\mathbf{E}_d$ ) between the human annotated nucleus centers and the true positive detections, and also the absolute difference (defined as  $\mathbf{E}_n$ ) between the number of human annotation and the true positive nucleus centers.

We compare deep voting with no stride (referred to as **DV-1**), deep voting with stride 3 (referred to as **DV-3**), and standard patch wise classification using CNN (referred to as **CNN+PC**). **CNN+PC** has the same network architecture to the proposed model, except that its last layer is a softmax classifier and results in probability map for every testing image. Please note that the fast scanning strategy produces the same detection result as **DV-1**.

We also compare this algorithm with three other state of the art procedures, including kernel based Iterative Radial Voting method (**IRV**) [5], structured SVM based Non-overlapping Extremal Regions Selection method (**NERS**) [14], and Image based

**FIGURE 7.2**

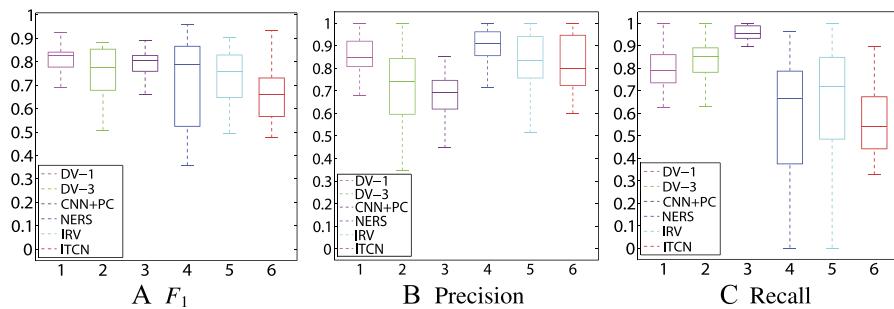
Nucleus detection results using deep voting on several randomly selected example images from Ki-67 stained NET dataset. The detected cell centers are marked with yellow dots. The ground-truth is represented with small green circles.

**Table 7.1** Nucleus localization evaluation on the three data sets.  $\mu_d$ ,  $\mu_n$ , and  $\sigma_d$ ,  $\sigma_n$  represent the mean and standard deviation of the two criteria  $\mathbf{E}_d$  and  $\mathbf{E}_n$ , respectively

Methods	P	R	F <sub>1</sub>	$\mu_d \pm \sigma_d$	$\mu_n \pm \sigma_n$
DV-1	0.852	0.794	<b>0.8152</b>	$2.98 \pm 1.939$	<b><math>9.667 \pm 10.302</math></b>
DV-3	0.727	0.837	0.763	$3.217 \pm 2.069$	$17.238 \pm 18.294$
CNN+PC	0.673	0.9573	0.784	$2.51 \pm 1.715$	$37.714 \pm 39.397$
NERS [14]	0.859	0.602	0.692	$2.936 \pm 2.447$	$41.857 \pm 57.088$
IRV [5]	0.806	0.682	0.718	$3.207 \pm 2.173$	$17.714 \pm 16.399$
ITCN [53]	0.778	0.565	0.641	$3.429 \pm 2.019$	$33.047 \pm 46.425$

Tool for Counting Nuclei method (ITCN) [53]. For fair comparison, we carefully preprocess the testing image for ITCN and IRV. We do not compare the proposed deep voting model with other methods, such as SVM and boosting using handcrafted features, because CNN is widely proven in recent literatures to produce superior performance in classification tasks.

**Experimental Results.** Fig. 7.2 shows some qualitative results using three randomly selected image patches. Hundreds of nuclei are correctly detected using our method. Deep voting is able to handle the touching objects, weak staining, and inhomogeneous intensity variations. Detailed quantitative results are summarized in Table 7.1. Fast scanning produces exactly the same result as DV-1 and thus is omitted from the table. It can be observed that deep voting consistently outperforms other methods, especially in terms of  $F_1$  score. Although NERS [14] achieves comparable  $P$  score, it does not produce good  $F_1$  score because Ki-67 stained NET images contain a lot of nuclei exhibiting weak staining and similar intensity values between nucleus and background, leading to large  $FP$ . Due to the relatively noisy probability map, both DV-3 and CNN-PC provide higher  $R$  score but with a large loss in precision. On

**FIGURE 7.3**

Boxplot of  $F_1$  score, precision, and recall on the NET data set.

the other hand, our method, which encodes the geometric information of the nucleus centroids and patch centers, produces much better overall performance than others. The **DV-3** also gives reasonable results with faster computational time. The boxplot in Fig. 7.3 provides detailed comparisons in terms of precision, recall, and  $F_1$  score.

The computation time for testing a  $400 \times 400$  RGB image is 22, 5.5, and 31 s for our three deep voting implementations, **DV-1**, **DV-3**, and fast scanning, respectively. In our work, both **DV-1** and **DV-3** are implemented using GPU, and deep voting using fast scanning is implemented with MATLAB.

### 7.1.5 CONCLUSION

In this section, we proposed a novel deep voting model for accurate and robust nucleus localization. We extended the conventional CNN model to jointly learn the voting confidence and voting offset by introducing a **hybrid nonlinear activation function**. We formulated our problem as a minimization problem with a novel loss function. In addition, we also demonstrated that our method can be accelerated significantly using simple strategies without sacrificing the overall detection accuracy. Both qualitative and quantitative experimental results demonstrate the superior performance of the proposed deep voting algorithm compared with several state of the art procedures. We will carry out more experiments and test our method on other image modalities in the future work.

---

## 7.2 STRUCTURED REGRESSION FOR ROBUST CELL DETECTION USING CONVOLUTIONAL NEURAL NETWORK

### 7.2.1 INTRODUCTION

In microscopic image analysis, robust cell detection is a crucial prerequisite for biomedical image analysis tasks, such as cell segmentation and morphological mea-

surements. Unfortunately, the success of cell detection is hindered by the nature of microscopic images with noise coming from touching cells, background clutters, large variations in the shape and the size of cells, and the use of different image acquisition techniques.

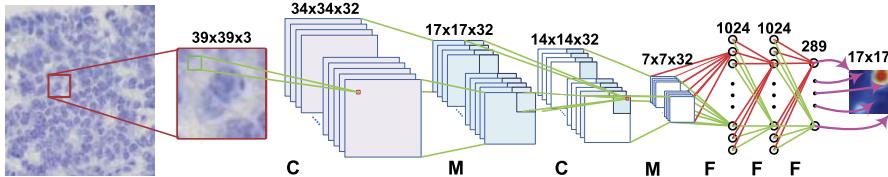
To alleviate these problems, a non-overlapping extremal regions selection method is presented in [14] and achieves state-of-the-art performance on their data sets. However, this work heavily relies on a robust region detector and thus the application is limited. Recently, deep learning based methods, which exploit the deep architecture to learn the hierarchical discriminative features, have shown great developments and achieved significant success in biomedical image analysis [26,27]. Convolutional neural network (CNN) attracts particular attentions among those works because of its outstanding performance. Ciresan et al. adopt CNN for mitosis detection [34] in breast cancer histology images and membrane neuronal segmentation [35] in microscopy images. Typically, CNN is used as a pixel-wise classifier. In the training stage, local image patches are fed into the CNN with their labels determined by the membership of the central pixel. However, this type of widely used approach ignores the fact the labeled regions are coherent and often exhibit certain topological structures. Failing to take this topological information into consideration will lead to implausible class label transition problem [54].

In this section, we propose a novel CNN based structured regression model for cell detection. The contributions are summarized as two parts: (i) We modify the conventional CNN by replacing the last layer (classifier) with a structured regression layer to encode the topological information. (ii) Instead of working on the label space, regression on the proposed structured proximity space for patches is performed so that centers of image patches are explicitly forced to get higher value than their neighbors. The proximity map produced with our novel fusion scheme contains much more robust local maxima for cell centers. To the best of our knowledge, this is the first study to report the application of structured regression model using CNN for cell detection.

### 7.2.2 METHODOLOGY

We formulate the cell detection task as a structured learning problem. We replace the last (classifier) layer that is typically used in conventional CNN with a structured regression layer. The proposed model encodes the topological information in the training data. In the testing stage, instead of assigning hard class labels to pixels, our model generates a proximity patch which provides much more precise cues to locate cell centers. To obtain the final proximity map for an entire testing image, we propose to fuse all the generated proximity patches together.

**CNN-Based Structured Regression.** Let  $\mathcal{X}$  denote the patch space, which consists of  $d \times d \times c$  local image patches extracted from  $c$ -channel color images. An image patch  $x \in \mathcal{X}$  centered at the location  $(u, v)$  of image  $I$  is represented by a quintuple  $\{u, v, d, c, I\}$ . We define  $\mathcal{M}$  as the proximity mask corresponding to image  $I$ , and

**FIGURE 7.4**

The CNN architecture used in the proposed structured regression model. C, M, and F represent the convolutional layer, max pooling layer, and fully connected layer, respectively. The purple arrows from the last layer illustrate the mapping between the final layer's outputs to the final proximity patch.

compute the value of the  $(i, j)$ th entry of  $\mathcal{M}$  as

$$\mathcal{M}_{ij} = \begin{cases} \frac{1}{1+\alpha D(i,j)}, & \text{if } D(i, j) \leq r, \\ 0, & \text{otherwise,} \end{cases} \quad (7.12)$$

where  $D(i, j)$  represents the Euclidean distance from pixel  $(i, j)$  to the nearest human annotated cell center.  $r$  is a distance threshold and is set to be 5 pixels.  $\alpha$  is the decay ratio and is set to be 0.8.

The  $\mathcal{M}_{ij}$  can have values in the interval  $\mathcal{V} = [0, 1]$ . An image patch  $x$  has a corresponding proximity patch on the proximity mask (shown in Fig. 7.4). We define  $s \in \mathcal{V}^{d' \times d'}$  as the corresponding proximity patch for patch  $x$ , where  $d' \times d'$  denotes the proximity patch size. Note that  $d'$  is not necessarily equal to  $d$ . We further define the proximity patch  $s$  of patch  $x$  as  $s = \{u, v, d', \mathcal{M}\}$ . It can be viewed as the *structured label* of patch  $x = \{u, v, d, c, I\}$ .

We define the training data as  $\{(x^i, y^i) \in (\mathcal{X}, \mathcal{Y})\}_{i=1}^N$ , whose elements are pairs of inputs and outputs:  $x^i \in \mathcal{X}$ ,  $y^i = \Gamma(s^i)$ ,  $N$  is the number of training samples, and  $\Gamma : \mathcal{V}^{d' \times d'} \rightarrow \mathcal{Y}$  is a mapping function to represent the vectorization operation in column-wise order for a proximity patch.  $\mathcal{Y} \subset \mathcal{V}^{p \times 1}$  represents the output space of the structured regression model, where  $p = d' \times d'$  denotes the number of units in the last layer. Define functions  $\{f_l\}_{l=1}^L$  and  $\{\theta_l\}_{l=1}^L$  as the operations and parameters corresponding to each of the  $L$  layers, the training process of the structured regression model can be formulated as learning a mapping function  $\psi$  composed with  $\{f_1, \dots, f_L\}$ , which will map the image space  $\mathcal{X}$  to the output space  $\mathcal{Y}$ .

Given a set of training data  $\{(x^i, y^i) \in (\mathcal{X}, \mathcal{Y})\}_{i=1}^N$ ,  $\{\theta_l\}_{l=1}^L$  will be learned by solving the following optimization problem

$$\arg \min_{\theta_1, \dots, \theta_L} \frac{1}{N} \sum_{i=1}^N \mathcal{L}(\psi(x^i; \theta_1, \dots, \theta_L), y^i), \quad (7.13)$$

where  $\mathcal{L}$  is the loss function which will be defined shortly.

Eq. (7.13) can be solved using the classical backpropagation algorithm. In order to back propagate the gradients from the last layer (structured regression layer) to the lower layers, we need to differentiate the loss function defined on one training sample with respect to the inputs to the last layer. Let  $\mathbf{a}^i$  and  $\mathbf{o}^i$  represent the inputs and the outputs of the last layer. For one training example  $(\mathbf{x}^i, \mathbf{y}^i)$ , we can have  $\mathbf{o}^i = \psi(\mathbf{x}^i; \theta_1, \dots, \theta_L)$ . Denote by  $y_j^i$ ,  $a_j^i$ , and  $o_j^i$  the  $j$ th element of  $\mathbf{y}^i$ ,  $\mathbf{a}^i$ , and  $\mathbf{o}^i$ , respectively. The loss function  $\mathcal{L}$  for  $(\mathbf{x}^i, \mathbf{y}^i)$  is given by

$$\begin{aligned}\mathcal{L}(\psi(\mathbf{x}^i; \theta_1, \dots, \theta_L), \mathbf{y}^i) &= \mathcal{L}(\mathbf{o}^i, \mathbf{y}^i) = \frac{1}{2} \sum_{j=1}^p (y_j^i + \lambda)(y_j^i - o_j^i)^2 \\ &= \frac{1}{2} \|(\text{Diag}(\mathbf{y}^i) + \lambda \mathbf{I})^{1/2} (\mathbf{y}^i - \mathbf{o}^i)\|_2^2,\end{aligned}\quad (7.14)$$

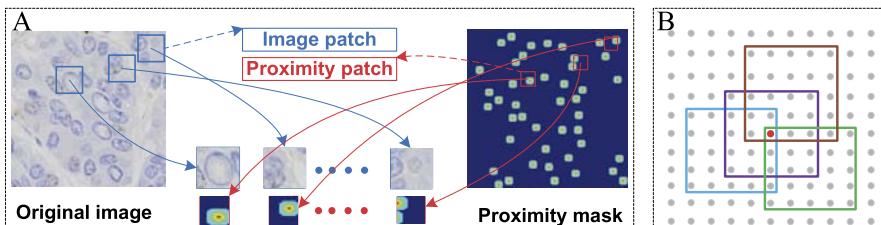
where  $\mathbf{I}$  is an identity matrix of size  $p \times p$ , and  $\text{Diag}(\mathbf{y}^i)$  is a diagonal matrix with the  $j$ th diagonal element equal to  $y_j^i$ . Since the nonzero region in the proximity patch is relatively small, our model might return a trivial solution. To avoid this problem, we adopt a weighting strategy [55] to give the loss coming from the network's outputs corresponding to the nonzero area in the proximity patch more weights. A small  $\lambda$  indicates strong penalization of errors coming from the outputs with low proximity values in the training data. This method is different from [55] which applies a bounding box mask regression approach on the entire image.

We choose the sigmoid activation function in the last layer, i.e.,  $o_j^i = \text{sigm}(a_j^i)$ . The partial derivative of (7.14) with respect to the input of the  $j$ th unit in the last layer is given by

$$\frac{\partial \mathcal{L}(\mathbf{o}^i, \mathbf{y}^i)}{\partial a_j^i} = \frac{\partial \mathcal{L}(\mathbf{o}^i, \mathbf{y}^i)}{\partial o_j^i} \frac{\partial o_j^i}{\partial a_j^i} = (y_j^i + \lambda)(o_j^i - y_j^i)a_j^i(1 - a_j^i).\quad (7.15)$$

Based on (7.15), we can evaluate the gradients of (7.13) with respect to the model's parameters in the same way as [52]. The optimization procedure is based on mini-batch stochastic gradient descent.

**CNN Architecture.** The proposed structured regression model contains several convolutional (C), max-pooling (M), and fully-connected layers (F). Fig. 7.4 illustrates one of the architectures and mapped proximity patches in the proposed model. The detailed model configuration is: Input( $49 \times 49 \times 3$ ) – C( $44 \times 44 \times 32$ ) – M( $22 \times 22 \times 32$ ) – C( $20 \times 20 \times 32$ ) – M( $10 \times 10 \times 32$ ) – C( $8 \times 8 \times 32$ ) – F(1024) – F(1024) – F(289). The activation function of last F (regression) layer is chosen to be the sigmoid, and ReLu is used for all the other F and C layers. Here, the sizes of C and M layers are defined as  $\text{width} \times \text{height} \times \text{depth}$ , where  $\text{width} \times \text{height}$  determines the dimensionality of each feature map and  $\text{depth}$  represents the number of feature maps. The filter size is chosen to be  $6 \times 6$  for the first convolutional layer and  $3 \times 3$  for the remaining two. The max pooling layer uses a window of size  $2 \times 2$  with a stride of 2.

**FIGURE 7.5**

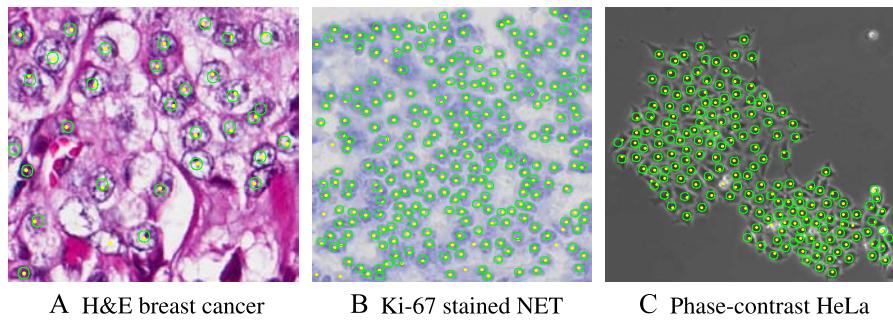
(A) The training data generation process. Each original image has a proximity mask of the same size and each local image patch has a proximity patch used as the structured *label*. (B) The fusion process. Each pixel receives predictions from its neighborhoods. For example, the red dot collects all the predictions from its 25 neighboring pixels and an average value will be assigned as the final result. In this figure, we only display 4 out of 25 proximity patches.

**Structured Prediction Fusion and Cell Localization.** Given a testing image patch  $x = (u, v, d, c, I)$ , it is easy to get the corresponding proximity mask  $s = \Gamma^{-1}(y)$ , where  $y \in \mathcal{Y}$  represents the model's output corresponding to  $x$ . In the fusion process,  $s$  will cast a proximity value for every pixel that lies in the  $d' \times d'$  neighborhood area of  $(u, v)$ , for example, pixel  $(u + i, v + j)$  in image  $I$  will get a prediction  $s_{ij}$  from pixel  $(u, v)$ . In other words, as we show in Fig. 7.5B, each pixel actually receives  $d' \times d'$  predictions from its neighboring pixels. To get the fused proximity map, we average all the predictions for each pixel from its neighbors and calculate its final proximity prediction. After this step, the cell localization can be easily obtained by finding the local maximum positions in the average proximity map.

**Speed Up.** Traditional sliding window method is time consuming. However, we have implemented two strategies to speed up. The first one comes from the property that our model generates a  $d' \times d'$  proximity patch for each testing patch. This makes it feasible to skip a lot of pixels and only test the image patches at a certain stride  $ss$  ( $1 \leq ss \leq d'$ ) without significantly sacrificing the accuracy. The second strategy, called *fast scanning* [56], is based on the fact that there exists a lot of redundant convolution operations among adjacent patches when computing the sliding-windows.

### 7.2.3 EXPERIMENTAL RESULTS

**Data Set and Implementation Details.** This model is implemented in C++ and CUDA based on the fast CNN kernels [30], and *fast scanning* [56] is implemented in MATLAB. The proposed algorithm is trained and tested on a PC with an Intel Xeon E5 CPU and an NVIDIA Tesla k40C GPU. The learning rate is set to be 0.0005 and a dropout rate of 0.2 is used for the fully connected layers. The  $\lambda$  is set to be 0.3 in (7.14).

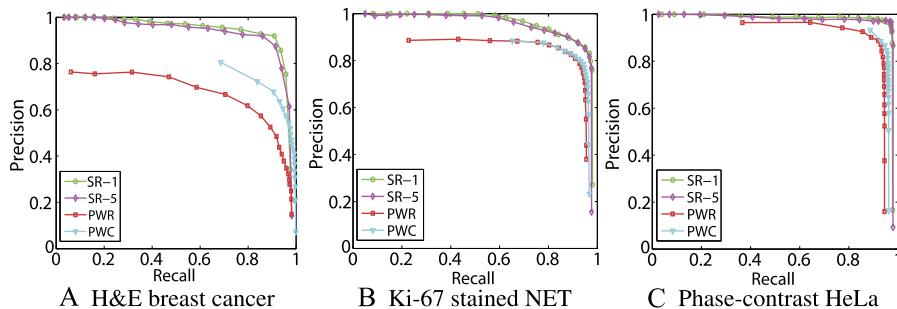
**FIGURE 7.6**

Cell detection results on three sample images from the three data sets. Yellow dots represent the detected cell centers. The ground truth annotations are represented by green circles for better illustrations.

Three data sets are used to evaluate the proposed method: First, The Cancer Genome Atlas (TCGA) dataset, from which we cropped and annotated 32 H&E-stained microscopy images of breast cancer cells, has a magnification of  $40\times$ . The detection task in this data set is challenging due to highly inhomogeneous background noise, a large variability of the size of cells, and background similarities. The second dataset is obtained from [14] and contains 22 phase contrast images of HeLa cervical cancer cell. These images exhibit large variations in sizes and shapes. The third dataset contains 60 Ki67-stained neuroendocrine tumor (NET), the magnification is  $40\times$ . Many touching cells, weak staining, and fuzzy cell boundaries are present in this data. Due to the limited resource of computation, we do not use cross-validation, instead all these datasets are equally split to construct training and testing sets.

**Model Evaluation.** Fig. 7.6 shows the qualitative detection results on three datasets. For quantitative analysis, we define the ground-truth areas as circular regions within 5 pixels of every annotated cell center. A detected cell centroid is considered to be a true positive ( $TP$ ) only if it lies within the ground-truth areas; otherwise, it is considered as a false positive ( $FP$ ). Each  $TP$  is matched with the nearest ground-truth annotated cell center. The ground-truth cell centers that are not matched by any detected results are considered to be false negatives ( $FN$ ). Based on the above definitions, we can compute the precision ( $P$ ), recall ( $R$ ), and  $F_1$ -score as  $P = \frac{TP}{TP+FP}$ ,  $R = \frac{TP}{TP+FN}$ , and  $F_1 = \frac{2PR}{P+R}$ , respectively.

We evaluated four variations of the proposed methods: (1, 2) *Structured Regression + testing with a stride ss* (SR-ss), where  $ss$  is chosen to be 1 for (1) and 5 for (2); (3) *CNN based Pixel-Wise Classification* (PWC), which shares a similar architecture with the proposed method except that it utilizes the softmax classifier in the last layer; and (4) *CNN based Pixel-Wise Regression* (PWR), which is similar to SR-1 but only predicts the proximity value for the central pixel of each patch.

**FIGURE 7.7**

Precision–recall curves of the four variations of the proposed algorithm on three data sets. SR-5 achieves almost the same results as SR-1. The proposed SR-1 significantly outperforms the other two pixel-wise methods using CNN.

**Fig. 7.7** shows the precision–recall curves of the four variations of the proposed method on each data set. These curves are generated by changing the threshold  $\zeta$  on the final proximity maps before finding the local maximum. We can see that SR-5 achieves almost the same performance as SR-1, and both PWR and PWC don't work as well as the proposed structured regression model, especially for the H& E breast cancer data set that exhibits high background similarity and large variations in cell size. This demonstrates that introducing structured regression increases the overall performance. The computational costs for SR-1, SR-5, and *fast scanning* are 14.5, 5, and 19 s for testing a  $400 \times 400$  RGB image. In the training stage, our model takes about 5 h to converge.

**Comparison with Other Works.** We also compare our structured regression model (SR) with four state-of-the-art methods, including Non-overlapping Extremal Regions Selection (NERS) [14], Iterative Radial Voting (IRV) [5], Laplacian-of-Gaussian filtering (LoG) [2], and Image-based Tool for Counting Nuclei (ITCN) [53]. In addition to Precision, Recall, and  $F_1$  score, we also compute the mean and standard deviation of two terms: (i) the absolute difference  $E_n$  between the number of true positive and the ground-truth annotations, and (ii) the Euclidean distance  $E_d$  between the true positive and the corresponding annotations. The quantitative experiment results are reported in **Table 7.2**. It is obvious that our method has better performance than the above listed alternate models on all the datasets considered, especially in terms of the  $F_1$ -score. This method also exhibits strong reliability with the lowest mean and standard deviations in  $E_n$  and  $E_d$  on NET and phase contrast data sets.

#### 7.2.4 CONCLUSION

In this section, we proposed a structured regression model for robust cell detection. The proposed method differs from the conventional CNN classifiers by introducing

**Table 7.2** The comparative cell detection results on three data sets.  $\mu_d$ ,  $\sigma_d$  represent the mean and standard deviation of  $\mathbf{E}_d$ , and  $\mu_n$ ,  $\sigma_n$  represent the mean and standard deviation of  $\mathbf{E}_n$

Data set	Methods	P	R	F <sub>1</sub>	$\mu_d \pm \sigma_d$	$\mu_n \pm \sigma_n$
H&E breast cancer	SR-1	<b>0.919</b>	0.909	<b>0.913</b>	<b>3.151 ± 2.049</b>	4.8750 ± 2.553
	NERS [14]	–	–	–	–	–
	IRV [5]	0.488	0.827	0.591	5.817 ± 3.509	9.625 ± 4.47
	LoG [2]	0.264	<b>0.95</b>	0.398	7.288 ± 3.428	<b>2.75 ± 2.236</b>
	ITCN [53]	0.519	0.528	0.505	7.569 ± 4.277	26.188 ± 8.256
NET	SR-1	0.864	<b>0.958</b>	<b>0.906</b>	<b>1.885 ± 1.275</b>	<b>8.033 ± 10.956</b>
	NERS [14]	<b>0.927</b>	0.648	0.748	2.689 ± 2.329	32.367 ± 49.697
	IRV [5]	0.872	0.704	0.759	2.108 ± 3.071	15.4 ± 14.483
	LoG [2]	0.83	0.866	0.842	3.165 ± 2.029	11.533 ± 21.782
	ITCN [53]	0.797	0.649	0.701	3.643 ± 2.084	24.433 ± 40.82
Phase contrast	SR-1	<b>0.942</b>	<b>0.972</b>	<b>0.957</b>	<b>2.069 ± 1.222</b>	<b>3.455 ± 4.547</b>
	NERS [14]	0.934	0.901	0.916	2.174 ± 1.299	11.273 ± 11.706
	IRV [5]	0.753	0.438	0.541	2.705 ± 1.416	58.818 ± 40.865
	LoG [2]	0.615	0.689	0.649	3.257 ± 1.436	29.818 ± 16.497
	ITCN [53]	0.625	0.277	0.371	2.565 ± 1.428	73.727 ± 41.867

a new structured regressor to capture the topological information of the data. Spatial coherence is maintained across the image at the same time. In addition, our proposed algorithm can be implemented with several fast implementation options. We experimentally demonstrated the superior performance of the proposed method compared with several state-of-the-art methods. We also showed that the proposed method can handle different types of microscopy images with outstanding performance. In future work, we will validate the generality of the proposed model on other image modalities.

## ACKNOWLEDGEMENTS

This research is supported by NIH grant R01 AR065479-02.

## REFERENCES

1. P. Quelhas, M. Marcuzzo, A.M. Mendonça, A. Campilho, Cell nuclei and cytoplasm joint segmentation using the sliding band filter, *IEEE Trans. Med. Imaging* 29 (8) (2010) 1463–1473.
2. Y. Al-Kofahi, W. Lassoued, W. Lee, B. Roysam, Improved automatic detection and segmentation of cell nuclei in histopathology images, *IEEE Trans. Biomed. Eng.* 57 (4) (2010) 841–852.

3. E. Bernardis, S.X. Yu, Finding dots: segmentation as popping out regions from boundaries, in: CVPR, 2010, pp. 199–206.
4. C. Zhang, J. Yarkony, F.A. Hamprecht, Cell detection and segmentation using correlation clustering, in: MICCAI, vol. 8673, 2014, pp. 9–16.
5. B. Parvin, Q. Yang, J. Han, H. Chang, B. Rydberg, M.H. Barcellos-Hoff, Iterative voting for inference of structural saliency and characterization of subcellular events, *IEEE Trans. Image Process.* 16 (2007) 615–623.
6. X. Qi, F. Xing, D.J. Foran, L. Yang, Robust segmentation of overlapping cells in histopathology specimens using parallel seed detection and repulsive level set, *IEEE Trans. Biomed. Eng.* 59 (3) (2012) 754–765.
7. F. Xing, H. Su, L. Yang, An integrated framework for automatic Ki-67 scoring in pancreatic neuroendocrine tumor, in: MICCAI, 2013, pp. 436–443.
8. B. Parvin, Qing Yang, Ju Han, Hang Chang, B. Rydberg, M.H. Barcellos-Hoff, Iterative voting for inference of structural saliency and characterization of subcellular events, *IEEE Trans. Image Process.* 16 (3) (2007) 615–623.
9. Adel Hafiane, Filiz Bunyak, Kannappan Palaniappan, Fuzzy clustering and active contours for histopathology image segmentation and nuclei detection, in: Advanced Concepts for Intelligent Vision Systems, vol. 5259, Springer, Berlin, Heidelberg, ISBN 978-3-540-88457-6, 2008, pp. 903–914.
10. Gang Li, Tianming Liu, Jingxin Nie, Lei Guo, Jarema Malicki, Andrew Mara, Scott A. Holley, Weiming Xia, Stephen T.C. Wong, Detection of blob objects in microscopic zebrafish images based on gradient vector diffusion, *Cytometry Part A* 71 (10) (2007) 835–845.
11. H. Cinar Akakin, H. Kong, C. Elkins, J. Hemminger, B. Miller, J. Ming, E. Plocharczyk, R. Roth, M. Weinberg, R. Ziegler, G. Lozanski, M.N. Gurcan, Automated detection of cells from immunohistochemically-stained tissues: application to Ki-67 nuclei staining, in: Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series, vol. 8315, 2012, p. 3.
12. Lin Yang, Oncel Tuzel, Peter Meer, David J. Foran, Automatic image analysis of histopathology specimens using concave vertex graph, in: Medical Image Computing and Computer-Assisted Intervention – MICCAI 2008, in: Lect. Notes Comput. Sci., vol. 5241, ISBN 978-3-540-85987-1, 2008, pp. 833–841.
13. Hui Kong, M. Gurcan, K. Belkacem-Boussaid, Partitioning histopathological images: an integrated framework for supervised color-texture segmentation and cell splitting, *IEEE Trans. Med. Imaging* 30 (9) (2011) 1661–1677.
14. C. Arteta, V. Lempitsky, J.A. Noble, A. Zisserman, Learning to detect cells using non-overlapping extremal regions, in: MICCAI, vol. 7510, 2012, pp. 348–356.
15. Luca Bertelli, Tianli Yu, Diem Vu, Burak Gokturk, Kernelized structural SVM learning for supervised object segmentation, in: 2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2011, pp. 2153–2160.
16. B. Leibe, A. Leonardis, B. Schiele, Combined object categorization and segmentation with an implicit shape model, in: Proceedings of the Workshop on Statistical Learning in Computer Vision, Prague, Czech Republic, 2004.
17. Juergen Gall, Victor Lempitsky, Class-Specific Hough Forests for Object Detection, 2009, pp. 255–258.
18. Xiangfei Kong, Kuan Li, Jingjing Cao, Qingxiong Yang, Liu Wenyan, Hep-2 cell pattern classification with discriminative dictionary learning, *Pattern Recognit.* 47 (7) (2014) 2379–2388.

19. Hai Su, Fuyong Xing, Jonah D. Lee, Charlotte A. Peterson, Lin Yang, Automatic myonuclear detection in isolated single muscle fibers using robust ellipse fitting and sparse optimization, *IEEE/ACM Trans. Comput. Biol. Bioinform.* 11 (4) (2014) 714–726.
20. Kunihiro Fukushima, Neocognitron: a self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position, *Biol. Cybern.* 36 (4) (1980) 193–202.
21. Sven Behnke, *Hierarchical Neural Networks for Image Interpretation*, vol. 2766, Springer Science & Business Media, 2003.
22. Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, Gradient-based learning applied to document recognition, *Proc. IEEE* 86 (11) (1998) 2278–2324.
23. Patrice Y. Simard, Dave Steinkraus, John C. Platt, Best practices for convolutional neural networks applied to visual document analysis, in: 2013 12th International Conference on Document Analysis and Recognition, vol. 2, IEEE Computer Society, 2003, p. 958.
24. Clement Farabet, Camille Couprie, Laurent Najman, Yann LeCun, Learning hierarchical features for scene labeling, *IEEE Trans. Pattern Anal. Mach. Intell.* 35 (8) (2013) 1915–1929.
25. A. Cruz-Roa, J. Arevalo-Ovalle, A. Madabhushi, F. Gonzalez-Osorio, A deep learning architecture for image representation, visual interpretability and automated basal-cell carcinoma cancer detection, in: MICCAI, vol. 8150, 2013, pp. 403–410.
26. S. Liao, Y. Gao, A. Oto, D. Shen, Representation learning: a unified deep learning framework for automatic prostate MR segmentation, in: MICCAI, vol. 8150, 2013, pp. 254–261.
27. R. Li, W. Zhang, H. Suk, L. Wang, J. Li, D. Shen, S. Ji, Deep learning based imaging data completion for improved brain disease diagnosis, in: MICCAI, 2014, pp. 305–312.
28. Y.A. LeCun, L. Bottou, G.B. Orr, K. Müller, Efficient BackProp, in: *Neural Networks: Tricks of the Trade*, vol. 1524, 1998, pp. 9–50.
29. Clément Farabet, Camille Couprie, Laurent Najman, Yann LeCun, Learning hierarchical features for scene labeling, *IEEE Trans. Pattern Anal. Mach. Intell.* 35 (8) (2013) 1915–1929.
30. Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton, ImageNet classification with deep convolutional neural networks, in: NIPS, 2012, pp. 1106–1114.
31. Alexander Toshev, Christian Szegedy, DeepPose: human pose estimation via deep neural networks, in: IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 1653–1660.
32. Christian Szegedy, Alexander Toshev, Dumitru Erhan, Deep neural networks for object detection, in: NIPS, 2013, pp. 2553–2561.
33. Gernot Riegler, David Ferstl, Matthias Rüther, Horst Bischof, Hough networks for head pose estimation and facial feature localization, in: Proceedings of the British Machine Vision Conference, 2014.
34. D. Ciresan, A. Giusti, L.M. Gambardella, J. Schmidhuber, Mitosis detection in breast cancer histology images with deep neural networks, in: MICCAI, vol. 8150, 2013, pp. 411–418.
35. D.C. Ciresan, A. Giusti, L.M. Gambardella, J. Schmidhuber, Deep neural networks segment neuronal membranes in electron microscopy images, in: NIPS, 2012, pp. 2852–2860.
36. W. Zhang, R. Li, H. Deng, L. Wang, W. Lin, S. Ji, D. Shen, Deep convolutional neural networks for multi-modality isointense infant brain image segmentation, in: Neuroimage, 2014.
37. A. Payan, G. Montana, Predicting Alzheimer’s disease: a neuroimaging study with 3D convolutional neural networks, 2015.

38. Yuanpu Xie, Fuyong Xing, Xiangfei Kong, Hai Su, Lin Yang, Beyond classification: structured regression for robust cell detection using convolutional neural network, in: MICCAI, 2015, pp. 358–365.
39. Yuanpu Xie, Xiangfei Kong, Fuyong Xing, Fujun Liu, Hai Su, Lin Yang, Deep voting: a robust approach toward nucleus localization in microscopy images, in: MICCAI, 2015, pp. 374–382.
40. Fuyong Xing, Yuanpu Xie, Lin Yang, An automatic learning-based framework for robust nucleus segmentation, *IEEE Trans. Med. Imaging* 35 (2) (2016) 550–566.
41. G. Wu, M. Kim, Q. Wang, Y. Gao, S. Liao, D. Shen, Unsupervised deep feature learning for deformable registration of MR brain images, in: MICCAI, vol. 8150, 2013, pp. 649–656.
42. A. Prasoon, K. Petersen, C. Igel, F. Lauze, E. Dam, M. Nielsen, Deep feature learning for knee cartilage segmentation using a triplanar convolutional neural network, in: Int. Conf. Med. Image Comput. Comput. Assist. Intervent. (MICCAI), vol. 8150, 2013, pp. 246–253.
43. H.R. Roth, L. Lu, A. Seff, K.M. Cherry, J. Hoffman, S. Wang, J. Liu, E. Turkbey, R.M. Summers, A new 2.5 D representation for lymph node detection using random sets of deep convolutional neural network observations, in: MICCAI, 2014, pp. 520–527.
44. A.A. Cruz-Roa, J.E.A. Ovalle, A. Madabhushi, F.A.G. Osorio, A deep learning architecture for image representation, visual interpretability and automated basal-cell carcinoma cancer detection, in: MICCAI, 2013, pp. 403–410.
45. Alex Graves, Santiago Fernández, Jürgen Schmidhuber, Multi-dimensional recurrent neural networks, in: ICANN, 2007, pp. 549–558.
46. Wonmin Byeon, Thomas M. Breuel, Federico Raue, Marcus Liwicki, Scene labeling with LSTM recurrent neural networks, in: CVPR, 2015, pp. 3547–3555.
47. Jonathan Long, Evan Shelhamer, Trevor Darrell, Fully convolutional networks for semantic segmentation, in: CVPR, 2015, pp. 3431–3440.
48. Yuanpu Xie, Zizhao Zhang, Manish Sapkota, Lin Yang, Spatial clockwork recurrent neural network for muscle perimysium segmentation, in: MICCAI, 2016.
49. Olaf Ronneberger, Philipp Fischer, Thomas Brox, U-Net: convolutional networks for biomedical image segmentation, in: MICCAI, 2015, pp. 234–241.
50. Angela Yao, Juergen Gall, Luc Van Gool, A hough transform-based voting framework for action recognition, in: 2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2010, pp. 2061–2068.
51. E. Parzen, On the estimation of a probability density function and the mode, *Ann. Math. Stat.* (1962) 1065–1076.
52. Y. Lecun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, 86 (1998) 2278–2324.
53. J. Byun, M.R. Verardo, B. Sumengen, G.P. Lewis, B.S. Manjunath, S.K. Fisher, Automated tool for the detection of cell nuclei in digital microscopic images: application to retinal images, *Mol. Vis.* 12 (2006) 949–960.
54. P. Kontschieder, S.R. Bulò, H. Bischof, M. Pelillo, Structured class-labels in random forests for semantic image labelling, in: ICCV, 2012, pp. 2190–2197.
55. C. Szegedy, A. Toshev, D. Erhan, Deep neural networks for object detection, in: NIPS, 2013, pp. 2553–2561.
56. A. Giusti, D.C. Ciresan, J. Masci, L.M. Gambardella, J. Schmidhuber, Fast image scanning with deep max-pooling convolutional neural networks, in: ICIP, 2013, pp. 4034–4038.

This page intentionally left blank

PART

# Medical Image Segmentation

3

This page intentionally left blank

# Deep Learning Tissue Segmentation in Cardiac Histopathology Images

# 8

Jeffrey J. Nirschl\*, Andrew Janowczyk<sup>†</sup>, Eliot G. Peyster\*, Renee Frank\*,  
Kenneth B. Margulies\*, Michael D. Feldman\*, Anant Madabhushi<sup>†</sup>

University of Pennsylvania, Philadelphia, PA, United States\* Case Western Reserve University,  
Cleveland, OH, United States<sup>†</sup>

## CHAPTER OUTLINE

<b>8.1</b>	<b>Introduction .....</b>	180
<b>8.2</b>	<b>Experimental Design and Implementation .....</b>	183
8.2.1	<i>Data Set Description .....</i>	183
8.2.2	<i>Manual Ground Truth Annotations.....</i>	183
8.2.3	<i>Implementation .....</i>	183
8.2.3.1	<i>Network Architecture .....</i>	183
8.2.3.2	<i>Building a Training Database .....</i>	183
8.2.3.3	<i>Training a Deep Learning Model.....</i>	185
8.2.4	<i>Training a Model Using Engineered Features.....</i>	185
8.2.5	<i>Experiments .....</i>	186
8.2.5.1	<i>Experiment 1: Comparison of Deep Learning and Random Forest Segmentation .....</i>	186
8.2.5.2	<i>Experiment 2: Evaluating the Sensitivity of Deep Learning to Training Data .....</i>	187
8.2.6	<i>Testing and Performance Evaluation .....</i>	188
<b>8.3</b>	<b>Results and Discussion.....</b>	188
8.3.1	<i>Experiment 1: Comparison of Deep Learning and Random Forest Segmentation.....</i>	188
8.3.2	<i>Experiment 2: Evaluating the Sensitivity of Deep Learning to Training Data .....</i>	188
<b>8.4</b>	<b>Concluding Remarks .....</b>	191
<b>Notes.....</b>		191
<b>Disclosure Statement .....</b>		191
<b>Funding .....</b>		192
<b>References.....</b>		192

## CHAPTER POINTS

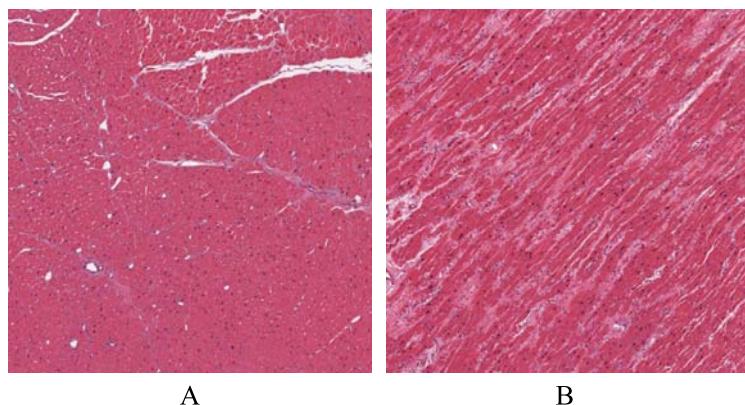
- Deep learning is a representation learning approach that learns informative features directly from data. It is domain independent and a generic workflow can be applied for multiple uses in digital pathology
- Deep learning is an efficient and robust method for myocyte and stroma segmentation in cardiac histopathology images with significant improvement over traditional handcrafted feature approaches

## 8.1 INTRODUCTION

The advent of whole-slide imaging (WSI) has increased the digitization of pathology slides, which is generating large, information-rich data sets where modern computer vision and machine learning methods thrive [1–3]. Traditional approaches for image analysis in digital pathology have involved manually engineering image features to use in a machine learning classifier [3]. These features generally involve pixel intensity statistics, texture descriptors, and image decompositions, as well as task-specific features. However, representation learning methods, such as deep convolutional neural networks or “deep learning”, have made significant advances in computer vision and are becoming increasingly common in medical image analysis [3,4]. Deep learning has many potential applications to digital pathology because it excels at tasks with large and complex training data sets, such as WSI. In addition, since deep learning models do not carry domain or tissue-specific information, a general workflow can be rapidly applied to problems across different domains. In this chapter, we provide a brief introduction to digital pathology, deep learning, and cardiac histopathology. We then present a workflow for segmentation of myocytes versus stroma in cardiac histopathology images. The long-term motivation for developing this segmentation workflow is to allow the exploration of whether computer-extracted histomorphometric features of tissue architecture from cardiac biopsy images are predictive of cardiac failure and/or rejection in the context of heart transplant patients.

Manual pathological examination takes advantage of the human visual system, which excels at automatically and efficiently partitioning an image into distinct components. The pathologist uses domain knowledge to provide biological context to the image and interpret structures into their appropriate tissue or cellular subtypes. This is followed by the identification of image features that can be used to label tissue as normal or abnormal. The final diagnosis is made through a combination of diagnostic criteria, histopathological image features, and prior experience. Computational analysis of digital pathology slides takes a similar process; image segmentation generally precedes the extraction of tissue-specific and disease-relevant features, which are ultimately used for diagnosis.

Image segmentation in digital pathology involves the partitioning the image into histologic primitives, or the biologically-relevant tissue, cellular, or sub-cellular structures (e.g. stromal tissue, nuclei, etc.) [5–8]. This allows the extraction and quantification of tissue or cell-specific features that can be used for diagnosis or prognosis

**FIGURE 8.1**

Example normal and abnormal cardiac tissue. (A) Cardiac biopsy from a patient without heart failure shows stroma limited to perivascular regions as well as regular, dense arrays of cardiomyocytes. (B) A biopsy from a patient with clinical heart failure shows an expansion of stromal tissue that disrupts the cardiomyocyte arrays. In addition, myocytes and their nuclei can be enlarged relative to normal tissue. Segmentation of myocytes and stroma is a first step toward quantifying these tissue abnormalities. Images are  $2000\text{ }\mu\text{m} \times 2000\text{ }\mu\text{m}$  at  $5\times$  magnification.

[9–14]. The process of applying computerized image analysis to quantify histologic features is known as quantitative histomorphometry. One important reason to segment histologic primitives prior to quantitative histomorphometry is to account for features that may have different predictive power depending on their context within the tissue. For example, normal cardiac tissue has stroma but it is limited to perivascular regions. However, when stroma is embedded between myocytes it is abnormal and can represent disease or fibrosis, as shown in Fig. 8.1. Although segmentation is an important step in digital pathology, manual segmentation is neither efficient nor routinely incorporated into clinical practice. Thus, developing automated algorithms for robust tissue segmentation is an important prerequisite for quantitative histomorphometry and predictive modeling.

Representation learning methods, such as deep convolutional neural networks or “deep learning”, have made significant advances in computer vision and are becoming increasingly common in medical image analysis [3,4,15]. Deep learning has many potential applications to digital pathology because it excels at tasks with large and complex training data sets, such as WSI. In addition, since deep learning models do not carry domain or tissue specific information, a general workflow can be applied to multiple cases. Deep learning differs from conventional approaches in that important features are not specified and designed *a priori*, but are rather learned directly from the data. A representation learning approach to image segmentation in digital

pathology reduces the time spent engineering features and ensures that the biological diversity and technical variance of the data set is captured.

In general, deep neural networks work by arranging nodes or artificial “neurons” in successive, convolutional, max-pooling, and fully connected layers as described elsewhere [16,17]. Parameters that specify neuron weighting are iteratively adjusted during training to approximate functions from data in a way that minimizes a loss function, which is typically correlated with the task at hand (e.g. class separation). The net result is the extraction of hierarchical features that map the input image into abstract feature vectors that capture complex and nonlinear relationships in the data.

Deep learning has recently been successfully applied to a number of digital pathology problems including localizing invasive ductal carcinoma [18], mitosis detection [5,6], colon gland segmentation [19], epithelial and stromal segmentation [8], and histopathological diagnostic support [14]. In this work, we explore the use of a deep learning framework for tissue segmentation in cardiac histopathology.

The two most prominent tissue classes in cardiac tissue are the myocytes, responsible for heart contraction, and the stroma, consisting of cellular and acellular support tissue. The stromal tissue is generally minimal in normal heart tissue. However, in the context of heart failure or cardiac rejection there is an expansion of the stromal tissue (see Fig. 8.1).

Cardiac biopsy is performed in the setting of new-onset heart failure, or for post-transplant rejection surveillance in adult heart transplant recipients [20,21]. Histopathological examination of these cases shows an expansion of the cellular and acellular stromal tissue, fibrosis, and morphological changes within myocytes, among other features. Manual review of biopsies is time consuming and intra- and inter-observer variability is high, even using revised diagnostic guidelines [22].

In cardiac biopsies, some features based on diagnostic criteria and domain knowledge include the percent tissue area that is stroma (fibrosis), lymphocyte presence and number in each tissue compartment, as well as morphological features of myocytes and their nuclei. Together, these features may allow automatic and quantitative prediction of the risk of heart failure or cardiac rejection, which would significantly improve efficiency, reliability, and quality of care for the treatment of heart failure and heart transplant patients.

In the remainder of this chapter, we focus on developing a deep learning framework for myocyte and stroma segmentation in cardiac histopathology images. Section 8.2 describes the design and implementation of two models for tissue segmentation, a deep learning approach or a random forest classifier with handcrafted features. We present our results for these experiments in Section 8.3. The first set of experiments compares the segmentation performance of a deep learning model to a traditional feature-engineered approach and supervised machine learning classifier. The second highlights the sensitivity of deep learning to the fidelity of the training annotations as well as patch representation within the training set. We conclude in Section 8.4 with potential downstream applications of myocyte and stroma segmentation in heart failure and rejection.

---

## 8.2 EXPERIMENTAL DESIGN AND IMPLEMENTATION

### 8.2.1 DATA SET DESCRIPTION

The data set includes 103 whole-slide images of H&E stained, full-thickness cardiac biopsies obtained at the time of death for non-heart-failure controls (52 patients) or heart transplant for patients with end-stage heart failure (51 patients). Images were scanned at  $20\times$  magnification using an Aperio ScanScope slide scanner and down-sampled to  $5\times$  magnification for image processing and analysis. An equal number of patients from each disease class (10 failing or non-failing) were randomly allocated to a held-out validation set and the remaining patients were used for training, such that 83 patient WSI were used for training models and 20 patients were kept as a held-out validation cohort.

### 8.2.2 MANUAL GROUND TRUTH ANNOTATIONS

For each patient in the training data, at least  $1 \times 10^4 \mu\text{m}^2$  tissue area of the WSI, per class, was manually annotated as myocyte or stroma. Training patches were selected from these annotated regions with hypersampling of regions at the interface between two tissue classes (myocyte or stroma), which we refer to as edge-hypersampling. See [Fig. 8.2](#) for example training annotations used for patch selection.

Manual ground truth annotations from the 20 patients in the held-out validation set were used to assess segmentation performance. These validation annotations were performed on one ROI of  $500 \mu\text{m} \times 500 \mu\text{m}$   $5\times$  magnification image for each patient. In each ROI, the entire field was annotated, in detail, to allow a pixel-level comparison of the manual and automated segmentation methods. Example ground truth validation annotations are shown in [Section 8.4](#).

### 8.2.3 IMPLEMENTATION

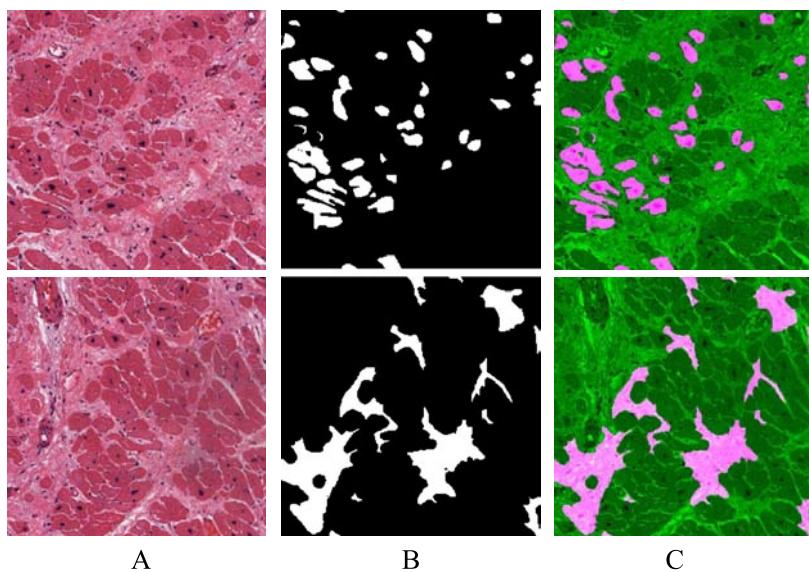
There are a number of excellent and well-maintained platforms for implementing deep learning. Apart from language, speed, and ease of implementation, an actively maintained package with a strong development community is important. We provide an abbreviated list of platforms in [Table 8.1](#). In this tutorial, we use the open-source framework Caffe [\[23\]](#). We direct the reader to [\[23\]](#) and the Caffe online documentation for additional information.

#### 8.2.3.1 *Network Architecture*

Many software platforms have existing networks that can be modified for new specific uses cases. In this example, we use a modified version of the AlexNet architecture [\[24,25\]](#), which was used in the  $32 \times 32$  CIFAR-10 challenge [\[26\]](#).

#### 8.2.3.2 *Building a Training Database*

The CIFAR-10 network requires training patches to be  $32 \times 32$  pixels. The first step, then, is to determine the appropriate magnification for training patches. In general, we

**FIGURE 8.2**

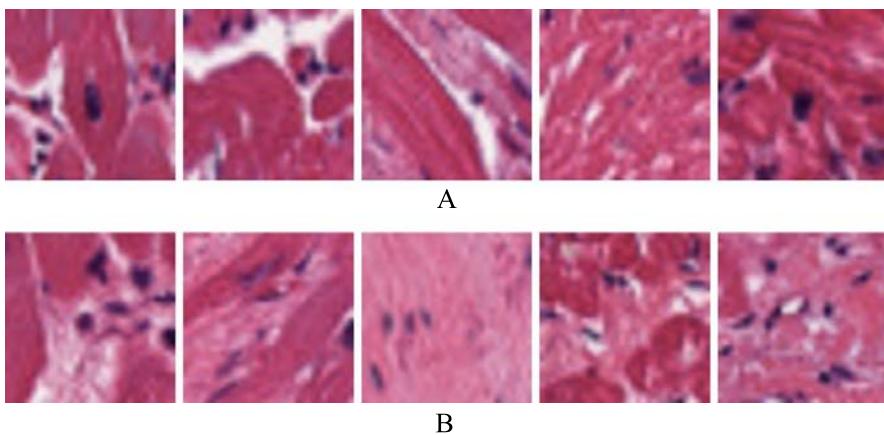
Ground truth for myocytes or stroma. (A) Original images. (B) Binary ground truth mask for myocytes (top) or stroma (bottom) for pixel-level training annotations. White pixels indicate class membership. (C) Overlay image of binary mask in B on original image in A. Images are  $500 \mu\text{m} \times 500 \mu\text{m}$  at  $5\times$  magnification.

**Table 8.1** Abbreviated list of deep learning software packages

Software	Language	URL
Caffe	C++	<a href="http://caffe.berkeleyvision.org/">http://caffe.berkeleyvision.org/</a>
Torch	LUA, C	<a href="http://torch.ch/">http://torch.ch/</a>
Theano	Python, C	<a href="http://deeplearning.net/software/theano/">http://deeplearning.net/software/theano/</a>
TensorFlow	Python, C++	<a href="https://www.tensorflow.org/">https://www.tensorflow.org/</a>

recommend the highest magnification that allows the histologic primitive of interest to fit within the patch window, while still containing contextual information outside of the histologic primitive. The patch should also have enough resolution and context for an expert to correctly classify the center pixel of image. For cardiac histopathology, we downsample the  $20\times$  magnification WSI to  $5\times$  apparent magnification, which allows a typical myocyte in cross-section to occupy approximately 50% of the patch window. Example training patches are shown in Fig. 8.3.

As mentioned previously, deep learning identifies discriminative features directly from training data. Thus, the next important consideration is the choice and representation of training patches. We find that training is more efficient when the transition zones, or the interfaces between one or more classes, are over-represented in the data

**FIGURE 8.3**

Example training patches. Example  $32 \times 32$  training patches used for both the deep learning and RF approach, where the center pixel of the patch determines the class label. (A) Myocyte training patches. (B) Stroma training patches. Each patch is  $64 \mu\text{m} \times 64 \mu\text{m}$  at  $5\times$  magnification.

set, as we later show in Experiment 2. To this end, we hypersample patches from the edges of the annotated training masks such that edge patches are over-represented in the training data set based on their distance to the mask edge.

#### 8.2.3.3 Training a Deep Learning Model

We train the deep learning classifier using  $5 \times 10^3$  patches per patient, and augment the training set by rotating each patch by  $90^\circ$ ,  $180^\circ$ , or  $270^\circ$  for  $1.66 \times 10^6$  patches per class and a total of  $3.32 \times 10^6$  training patches. These patches were split into three training and testing-folds at the patient level with balanced myocyte and stroma patches. Each fold was trained for  $6 \times 10^5$  iterations on a Tesla K20c GPU with CUDA 7.0 using the cuDNN setting, the AdaGrad solver [27] built into Caffe, and a fixed batch size. The network parameters for the AlexNet architecture are shown in Table 8.2.

#### 8.2.4 TRAINING A MODEL USING ENGINEERED FEATURES

We compare our deep learning approach to a traditional segmentation workflow using engineered features and a Random Forest classifier. Random forests were chosen as a classifier because they are among the top performing supervised machine learning algorithms [28,29], they perform well on biomedical image data sets [30], and require fewer adjustments to obtain stable results.

The Random Forest classifier for myocyte and stroma segmentation was trained using 1000 trees and 333 engineered-features based on intensity and texture. Ta-

**Table 8.2** Deep learning hyperparameters

Variable	Setting
Batch size	128
Initial learning rate	0.001
Learning rate schedule	AdaGrad
Rotations	[0, 90, 180, 270]
Number of iterations	600,000
Number of epochs	23
Weight decay	0.004
Random mirror	Enabled
Transformations	Mean-centered

**Table 8.3** Description of features used for the Random Forest stroma classifier

Category	Length	Features
Intensity	13 × 3 images	Energy, entropy, min, max, standard deviation, variance, mean, median, median absolute deviation, range, root mean squared, skewness, and kurtosis of patch intensity
Texture	98 × 3 images	13 Haralick texture features [33], 25 Laws texture features [34], 1 Local Binary Pattern [36], 21 XY gradient features, 6 Gabor features [32], 26 CoLlAGe features [31], 6 CORF features [35]

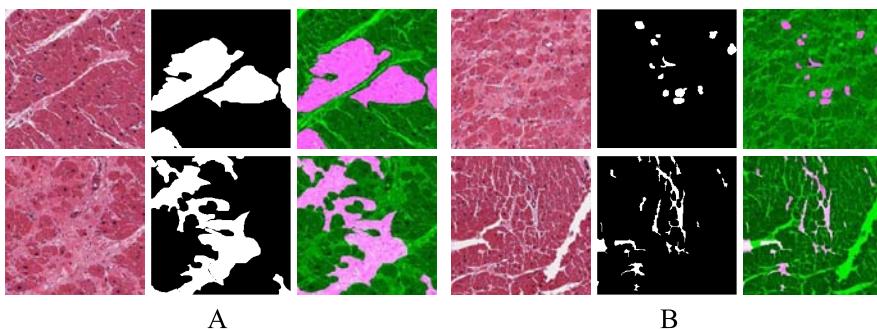
Intensity features were calculated on a  $32 \times 32$  sliding window with the center pixel assigned the computed value for each patch. The three grayscale images of each patch that were used include a grayscale conversion of the RGB image and a color-deconvolved hematoxylin or eosin grayscale image.

Table 8.3 lists the features used here. The intensity features capture the differences in H&E staining between myocytes and stroma by considering intensity magnitude, variation, and higher-order moments. Texture features such as Co-occurrence of Local Anisotropic Gradient Orientations (CoLlAGe) features [31], Gabor features [32], and Haralick features [33], among other texture features [34–36], were designed to detect the edges of class boundaries as well as the heterogeneity and disorder of the stroma relative to myocytes. These features were computed for each  $32 \times 32$  patch of the grayscale converted RGB image as well as the color-deconvolved hematoxylin or eosin grayscale image [37]. The feature vector and label for the center pixel patch was used for training. All of the training patches were used to construct a single RF model.

## 8.2.5 EXPERIMENTS

### 8.2.5.1 Experiment 1: Comparison of Deep Learning and Random Forest Segmentation

Experiment 1 compares the segmentation performance of deep learning model to a random forest model using engineered features in cardiac histopathology images. Each approach was trained using the same set of image patches and the output from

**FIGURE 8.4**

Coarse versus fine training annotations. This figure shows two different approaches for generating training annotations for deep learning. (A) Coarse annotations maximize the tissue area that is annotated, but may lack precise class boundaries. (B) Fine training annotations capture less tissue area and may require more effort, but generate delineate class boundaries with pixel-level precision. Experiment 2 shows the effect of training annotation and patch representation on the deep learning output. Myocytes are annotated in the top row and stromal tissue in the bottom row. Images are  $500\text{ }\mu\text{m} \times 500\text{ }\mu\text{m}$ .

each model is an image where each pixel is assigned a probability of class membership, between 0 and 1. Thresholding the probability output at 0.5 should maximally separate the classes, which is why a fixed threshold of 0.5 was used to create binary masks.

### **8.2.5.2 Experiment 2: Evaluating the Sensitivity of Deep Learning to Training Data**

Routine tissue annotation in digital pathology is generally performed at intermediate magnifications and produces masks that lack delineation of precise, pixel-level tissue boundaries. While this approach can rapidly annotate large regions of the WSI, it may have consequences for the data that ultimately go into the network. Since deep learning techniques learn models from the data presented, it is important to consider the training annotation method as well as the best procedure for sampling patches from within these annotations. Fig. 8.4 shows an example of coarse annotations (A) versus fine annotations with pixel-level boundaries (B).

Uniform random sampling of patches from training annotations assumes that all patches provide equally valuable information for classification. This is not true in general, though, as WSI often contain large regions of tissue that are qualitatively very different and easily classified using simple approaches. However, the transition zones between tissue classes are more challenging, and we can direct the deep learning network to learn these boundaries better by modifying the patch representation within the training data. We modify patch representation by hypersampling patches near the edge of the training annotation, or edge-hypersampling, an approach also

used in [38]. In this experiment we compare uniform and edge-hypersampling approaches for both coarse and fine, pixel-level training annotations.

### 8.2.6 TESTING AND PERFORMANCE EVALUATION

The fully annotated ROIs from the held-out patients were used to assess segmentation performance for both sets of experiments. We evaluate the segmentation performance for each experiment using multiple criteria including: the Area Under the Curve (AUC), F1 score, True Positive Rate (TPR), True Negative Rate (TNR), Positive Predictive Value (PPV), Modified Hausdorff Distance (MHD) [39,40].

---

## 8.3 RESULTS AND DISCUSSION

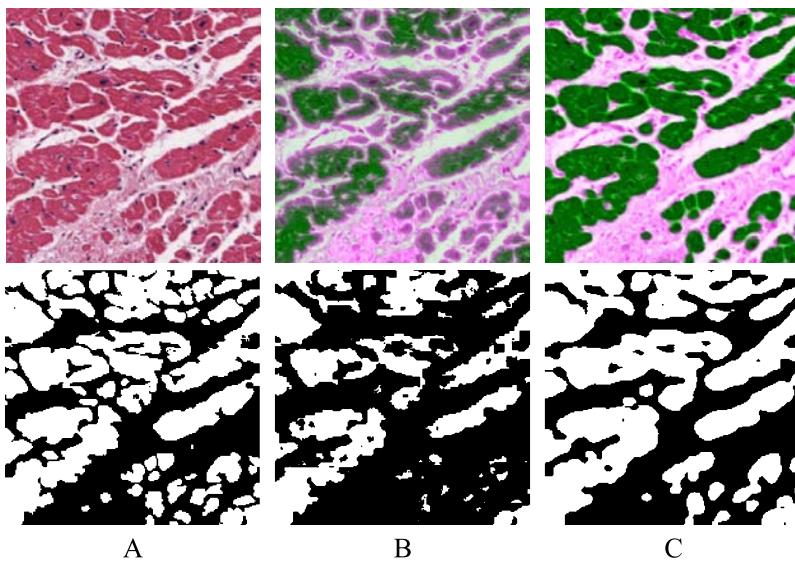
### 8.3.1 EXPERIMENT 1: COMPARISON OF DEEP LEARNING AND RANDOM FOREST SEGMENTATION

An overlay of the probability output on the original image for each classifier is shown in Fig. 8.5. Table 8.4 shows the quantitative evaluation of each classifier on 20 held-out patients using the performance metrics listed in the previous section (AUC, F1 score, TPR, TNR, PPV, MHD). Qualitatively, the deep learning model produces much more confident predictions of class membership, as noted by the green or magenta intensity of the probability outputs. It is also clear that both the deep learning and random forest models perform well on regions far from the interface with the opposing class, as expected. Since large regions of the image are easier to classify both models achieve a high F1 score and TPR. However, only the deep learning model is able to achieve both a high TPR (sensitivity) and TNR (specificity).

What is also evident from Fig. 8.5 is that deep learning more accurately delineates the class boundaries. This is quantified in Table 8.4 using the MHD, which is a robust measurement for matching objects based on their edge points [40]. The MHD shows that deep learning more accurately delineates tissue boundaries compared to the random forest model. Finally, related to the previous point, deep learning excels at detecting fine features such as small myocytes embedded in stromal tissue. These are often assigned low probabilities in the random forest model, which are lost with thresholding. Together, these data suggest that deep learning should be used when precise delineation of tissue boundaries are desired. However, in the next experiment we examine the sensitivity of deep learning and highlight an important caveat with deep learning models.

### 8.3.2 EXPERIMENT 2: EVALUATING THE SENSITIVITY OF DEEP LEARNING TO TRAINING DATA

Fig. 8.6 shows the effect of coarse versus fine training annotations and edge hypersampling on deep learning segmentation. The first result is that coarse annotations

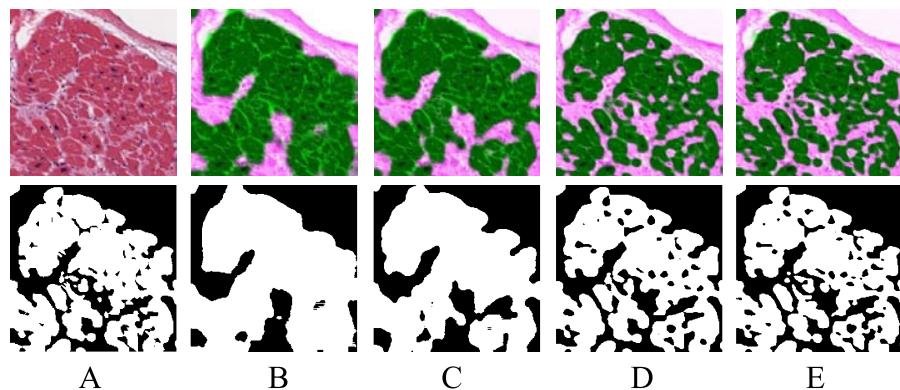
**FIGURE 8.5**

Qualitative comparison of segmentation performance. (A) The original H&E image and ground truth annotation. (B) Random forest probability output overlaid on the original image and binarized segmentation mask. (C) Deep learning probability output overlaid on the original image. The green or magenta channel intensity in the probability output probability of the pixel belonging to the myocyte or stroma class, respectively. White pixels in the binary masks represent the myocyte class. Images are 500  $\mu\text{m} \times 500 \mu\text{m}$ .

**Table 8.4** Mean  $\pm$  SD segmentation performance on validation set for Experiment 1

Metric	RF	DL Fold 1	DL Fold 2	DL Fold 3	DL Mean
AUC	$0.80 \pm 0.12$	$0.95 \pm 0.02$	$0.94 \pm 0.03$	$0.95 \pm 0.02$	$0.95 \pm 0.02$
F1 score	$0.91 \pm 0.05$	$0.96 \pm 0.03$	$0.96 \pm 0.04$	$0.96 \pm 0.03$	$0.96 \pm 0.03$
TPR	$0.92 \pm 0.09$	$0.96 \pm 0.03$	$0.96 \pm 0.05$	$0.96 \pm 0.04$	$0.96 \pm 0.04$
TNR	$0.67 \pm 0.29$	$0.94 \pm 0.03$	$0.93 \pm 0.05$	$0.94 \pm 0.03$	$0.94 \pm 0.04$
PPV	$0.90 \pm 0.05$	$0.96 \pm 0.03$	$0.97 \pm 0.02$	$0.97 \pm 0.03$	$0.97 \pm 0.03$
MHD	$122 \pm 68$	$50 \pm 11$	$54 \pm 12$	$49 \pm 10$	$51 \pm 11$

produce coarse results. While it is not entirely surprising, the visual results are striking. Edge hypersampling can partially compensate for coarse annotations, but the best results are obtained after re-annotation with precise pixel-level boundaries. The effect of edge-hypersampling is less clear in the specific case of fine myocyte and stroma annotations. This is likely due to the fact that a majority of the annotations are already at the tissue interface and already considered “edges”. The addition of edge-

**FIGURE 8.6**

Effect of training annotations fidelity on DL performance. (A) Original image (top) and ground truth annotation. (B) DL network trained with uniform random patch sampling from coarse masks. (C) Network trained with edge-hypersampling of patches from coarse masks. (D) Network with uniform random patch sampling of fine training masks. (E) Network trained with edge-hypersampling of fine training masks. Probability outputs overlaid on the original images (top row) and binary masks (bottom). Images are 500  $\mu\text{m} \times 500 \mu\text{m}$ .

**Table 8.5** Mean  $\pm$  SD segmentation performance on validation set for Experiment 2

Metric	Coarse-Unif	Coarse-Edge	Fine-Unif	Fine-Edge
AUC	$0.80 \pm 0.10$	$0.84 \pm 0.08$	$0.95 \pm 0.02$	$0.95 \pm 0.02$
F1 score	$0.90 \pm 0.06$	$0.92 \pm 0.06$	$0.96 \pm 0.05$	$0.96 \pm 0.03$
TPR	$0.94 \pm 0.08$	$0.96 \pm 0.05$	$0.96 \pm 0.03$	$0.96 \pm 0.03$
TNR	$0.67 \pm 0.24$	$0.71 \pm 0.19$	$0.95 \pm 0.03$	$0.94 \pm 0.03$
PPV	$0.88 \pm 0.08$	$0.89 \pm 0.07$	$0.96 \pm 0.03$	$0.97 \pm 0.03$
MHD	$250 \pm 148$	$226 \pm 146$	$50 \pm 11$	$49 \pm 10$

hypersampling to fine annotations has little effect, but importantly does not harm performance (Table 8.5).

Although, we manually annotate myocytes and stroma, due to their irregular boundaries, different histologic primitives may benefit from less time-consuming approaches. For example, reliable nuclei annotations can be obtained using an approach similar to Janowczyk et al. [38]. In this case, high fidelity “edge” annotations for the non-nuclei class were generated by dilating the nuclei mask and then subtracting the original mask. In any case, it is important for digital pathology users to match the granularity of their training annotations with the quality of the desired output.

---

## 8.4 CONCLUDING REMARKS

In this chapter we described the use of deep learning for the specific case of segmenting myocytes and stroma in cardiac histopathology images. We developed two models to segment myocytes and stroma, a deep learning approach and the use of a random forest classifier in conjunction with handcrafted features. A deep learning approach was used because this method has shown excellent performance in computer vision tasks and it excels with large data sets.

We evaluated the performance of our two models using a held-out data set of 20 patient images with expert annotated tissue boundaries, where deep learning was superior in all metrics. Most notably, deep learning segmentation had both a high sensitivity and specificity as well as more accurate tissue boundaries, compared to a random forest model. In addition, our results suggested that deep learning is highly sensitive to the fidelity of the training annotations and that edge-hypersampling can improve segmentation performance.

In summary, our deep learning framework required no feature engineering and showed improved segmentation performance compared to a classifier trained using handcrafted features. This will enable the identification and quantification of tissue-specific features predictive of heart failure or cardiac transplant rejection. Future work using these segmentation results will allow the development of features predictive of cardiac disease, and may eventually lead to pipelines for image-based predictors of outcome based off cardiac histopathology. This has the potential to improve cardiac biopsy screening, provide real-time feedback for clinicians, and serve as an objective second reader for pathologists.

---

## NOTES

For additional examples of deep learning in digital pathology, please see: <http://www.andrewjanowczyk.com/category/deep-learning>.

---

## DISCLOSURE STATEMENT

Dr. Madabhushi is the co-founder and stake holder in Ibris Inc., a cancer diagnostics company. Drs. Madabhushi and Feldman are equity holders and have technology licensed to both Elucid Bioimaging and Inspirata Inc. Drs. Madabhushi and Feldman are scientific advisory consultants for Inspirata Inc. and sit on its scientific advisory board. Dr. Feldman is also a consultant for Phillips Healthcare, XFIN, and Virbio. Dr. Margulies hold research grants from Thoratec Corporation and Merck and serves as a scientific consultant/ advisory board member for Janssen, Merck, Pfizer, Ridgetop Research, Glaxo-Smith-Kline, NovoNordisk.

---

## FUNDING

Research reported in this publication was supported by the National Cancer Institute of the National Institutes of Health under award numbers R21CA179327-01, R21CA195152-01, U24CA199374-01, the National Institute of Diabetes and Digestive and Kidney Diseases under award number R01DK098503-02, the DOD Prostate Cancer Synergistic Idea Development Award (PC120857), the DOD Lung Cancer Idea Development New Investigator Award (LC130463), the DOD Prostate Cancer Idea Development Award, the Case Comprehensive Cancer Center Pilot Grant, the VelaSano Grant from the Cleveland Clinic, the Wallace H. Coulter Foundation Program in the Department of Biomedical Engineering at Case Western Reserve University, the I-Corps@Ohio Program. JJN was supported by NINDS F30NS092227.

The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health.

---

## REFERENCES

1. Anant Madabhushi, Digital pathology image analysis: opportunities and challenges, *Imaging Med.* 1 (1) (2009) 7–10, <http://dx.doi.org/10.2217/iim.09.9>, URL <http://www.futuremedicine.com/doi/abs/10.2217/iim.09.9>.
2. Farzad Ghaznavi, Andrew Evans, Anant Madabhushi, Michael Feldman, Digital imaging in pathology: whole-slide imaging and beyond, *Annu. Rev. Phytopathol.* 8 (2013) 331–359, <http://dx.doi.org/10.1146/annurev-pathol-011811-120902>, URL <http://www.ncbi.nlm.nih.gov/pubmed/23157334>.
3. Rohit Bhargava, Anant Madabhushi, Emerging themes in image informatics and molecular analysis for digital pathology, *Annu. Rev. Biomed. Eng.* 18 (March) (2016) 387–412, <http://dx.doi.org/10.1146/annurev-bioeng-112415-114722>.
4. Yann LeCun, Yoshua Bengio, Geoffrey Hinton, Deep learning, *Nature* 521 (7553) (2015) 436–444, <http://dx.doi.org/10.1038/nature14539>, URL <https://www.ncbi.nlm.nih.gov/pubmed/26017442>.
5. Dan Cireşan, Ueli Meier, Juergen Schmidhuber, Multi-column deep neural networks for image classification, *Proc. Int. Conf. Pattern Recognit.* 25 (February) (2012) 3642–3649, <http://dx.doi.org/10.1109/CVPR.2012.6248110>.
6. Haibo Wang, Angel Cruz-Roa, Ajay Basavanhally, Hannah Gilmore, Natalie Shih, Mike Feldman, John Tomaszewski, Fabio Gonzalez, Anant Madabhushi, Mitosis detection in breast cancer pathology images by combining handcrafted and convolutional neural network features, *J. Med. Imaging* 1 (3) (2014) 034003, <http://dx.doi.org/10.1117/1.JMI.1.3.034003>.
7. J. Xu, L. Xiang, Q. Liu, H. Gilmore, J. Wu, J. Tang, A. Madabhushi, Stacked sparse autoencoder (SSAE) for nuclei detection on breast cancer histopathology images, *IEEE Trans. Med. Imaging* (2015), <http://dx.doi.org/10.1109/TMI.2015.2458702>.
8. Jun Xu, Xiaofei Luo, Guanhao Wang, Hannah Gilmore, Anant Madabhushi, A deep convolutional neural network for segmenting and classifying epithelial and stromal regions in histopathological images, *Neurocomputing* (2016) 1–10, <http://dx.doi.org/10.1016/j.neucom.2016.01.034>, URL <http://linkinghub.elsevier.com/retrieve/pii/S0925231216001004>.

9. A.H. Beck, A.R. Sangoi, S. Leung, R.J. Marinelli, T.O. Nielsen, M.J. van de Vijver, R.B. West, M. van de Rijn, D. Koller, Systematic analysis of breast cancer morphology uncovers stromal features associated with survival, *Sci. Transl. Med.* 3 (108) (2011) 108ra113, <http://dx.doi.org/10.1126/scitranslmed.3002564>, URL <http://stm.scienmag.org/content/scitransmed/3/108/108ra113.full.html>.
10. Ajay Basavanhally, Michael Feldman, Natalie Shih, Carolyn Mies, John Tomaszewski, Shridar Ganeshan, Anant Madabhushi, Multi-field-of-view strategy for image-based outcome prediction of multi-parametric estrogen receptor-positive breast cancer histopathology: comparison to oncotype DX, *J. Pathol. Inform.* 2 (2) (2011) S1, <http://dx.doi.org/10.4103/2153-3539.92027>, URL <http://www.ncbi.nlm.nih.gov/pubmed/22811953>.
11. Anant Madabhushi, Shannon Agner, Ajay Basavanhally, Scott Doyle, George Lee, Computer-aided prognosis: predicting patient and disease outcome via quantitative fusion of multi-scale, multi-modal data, *Comput. Med. Imaging Graph.* 35 (7–8) (2011) 506–514, <http://dx.doi.org/10.1016/j.compmedimag.2011.01.008>.
12. Scott Doyle, Michael Feldman, John Tomaszewski, Anant Madabhushi, A boosted Bayesian multiresolution classifier for prostate cancer detection from digitized needle biopsies, *IEEE Trans. Biomed. Eng.* 59 (5) (2012) 1205–1218, <http://dx.doi.org/10.1109/TBME.2010.2053540>.
13. James S. Lewis, Sahirzeeshan Ali, Jingqin Luo, Wade L. Thorstad, Anant Madabhushi, A quantitative histomorphometric classifier (QuHbIC) oropharyngeal squamous cell carcinoma, *Am. J. Surg. Pathol.* 38 (1) (2014) 128–137, <http://dx.doi.org/10.1097/PAS.0000000000000086>.
14. Geert Litjens, Clara I. Sánchez, Nadya Timofeeva, Meyke Hermsen, Iris Nagtegaal, Iringo Kovacs, Christina Hulsbergen-van de Kaa, Peter Bult, Bram van Ginneken, Jeroen van der Laak, Deep learning as a tool for increased accuracy and efficiency of histopathological diagnosis, *Sci. Rep.* 6 (April) (2016) 2628, <http://dx.doi.org/10.1038/srep26286>, URL <http://www.nature.com/articles/srep26286>.
15. Y. Bengio, A. Courville, P. Vincent, Representation learning: a review and new perspectives, *IEEE Trans. Pattern Anal. Mach. Intell.* 35 (8) (2013) 1798–1828, <http://dx.doi.org/10.1109/TPAMI.2013.50>.
16. Yoshua Bengio, Learning deep architectures for AI, *Found. Trends Mach. Learn.* 2 (1) (2009) 1–127, <http://dx.doi.org/10.1561/2200000006>.
17. Jürgen Schmidhuber, Deep learning in neural networks: an overview, *Neural Netw.* 61 (2015) 85–117, <http://dx.doi.org/10.1016/j.neunet.2014.09.003>.
18. Angel Cruz-Roa, Ajay Basavanhally, Fabio González, Hannah Gilmore, Michael Feldman, Shridar Ganeshan, Natalie Shih, John Tomaszewski, Anant Madabhushi, Automatic detection of invasive ductal carcinoma in whole slide images with convolutional neural networks, *Proc. SPIE* 9041 (216) (2014) 904103, <http://dx.doi.org/10.1117/12.2043872>.
19. Philipp Kainz, Michael Pfeiffer, Martin Urschler, Semantic segmentation of colon glands with deep convolutional neural networks and total variation segmentation, *arXiv:1511.06919*, 2015.
20. L.T. Cooper, K.L. Baughman, a.M. Feldman, a. Frustaci, M. Jessup, U. Kuhl, G.N. Levine, J. Narula, R.C. Starling, J. Towbin, R. Virmani, The role of endomyocardial biopsy in the management of cardiovascular disease: a scientific statement from the American Heart Association, the American College of Cardiology, and the European Society of Cardiology endorsed by the Heart Failure Society of America and the Heart Failure Association of the European Society of Cardiology, *Eur. Heart J.* 28 (24)

- (2007) 3076–3093, <http://dx.doi.org/10.1093/eurheartj/ehm456>, URL <http://eurheartj.oxfordjournals.org/cgi/doi/10.1093/eurheartj/ehm456>.
21. Maria Rosa Costanzo, Anne Dipchand, Randall Starling, Allen Anderson, Michael Chan, Shashank Desai, Savitri Fedson, Patrick Fisher, Gonzalo Gonzales-Stawinski, Luigi Martinelli, David McGiffin, Francesco Parisi, Jon Smith, David Taylor, Bruno Meiser, Steven Webber, David Baran, Michael Carboni, Thomas Dengler, David Feldman, Maria Frigerio, Abdallah Kfoury, Daniel Kim, Jon Kobashigawa, Michael Shullo, Josef Stehlík, Jeffrey Teuteberg, Patricia Uber, Andreas Zuckermann, Sharon Hunt, Michael Burch, Geetha Bhat, Charles Canter, Richard Chinnock, Marisa Crespo-Leiro, Reynolds Delgado, Fabienne Dobbels, Kathleen Grady, Walter Kao, Jacqueline Lamour, Gareth Parry, Jignesh Patel, Daniela Pini, Sean Pinney, Jeffrey Towbin, Gene Wolfel, Diego Delgado, Howard Eisen, Lee Goldberg, Jeff Hosenpud, Maryl Johnson, Anne Keogh, Clive Lewis, John O’Connell, Joseph Rogers, Heather Ross, Stuart Russell, Johan Vanhaecke, The International Society of Heart and Lung Transplantation guidelines for the care of heart transplant recipients, *J. Heart Lung Transplant.* 29 (8) (2010) 914–956, <http://dx.doi.org/10.1016/j.healun.2010.05.034>.
  22. Annalisa Angelini, Claus Boegelund Andersen, Giovanni Bartoloni, Fiona Black, Paul Bishop, Helen Doran, Marny Fedrigo, Jochen W.U. Fries, Martin Goddard, Heike Goebel, Desley Neil, Ornella Leone, Andrea Marzullo, Monika Ortmann, Francois Paraf, Samuel Rotman, Nesrin Turhan, Patrick Bruneval, Anna Chiara Frigo, Francesco Grigoletto, Alessio Gasparetto, Roberto Mencarelli, Gaetano Thiene, Margaret Burke, A web-based pilot study of inter-pathologist reproducibility using the ISHLT 2004 working formulation for biopsy diagnosis of cardiac allograft rejection: the European experience, *J. Heart Lung Transplant.* 30 (11) (2011) 1214–1220, <http://dx.doi.org/10.1016/j.healun.2011.05.011>.
  23. Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, Trevor Darrell, Caffe: convolutional architecture for fast feature embedding, in: Proceedings of the ACM International Conference on Multimedia, 2014, pp. 675–678, arXiv:1408.5093, 2015.
  24. Alex Krizhevsky, G. Hinton, Convolutional deep belief networks on CIFAR-10, Unpublished manuscript, pp. 1–9, URL <http://www.cs.utoronto.ca/~kriz/conv-cifar10-aug2010.pdf>, 2010.
  25. Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton, ImageNet classification with deep convolutional neural networks, *Adv. Neural Inf. Process. Syst.* (2012) 1–9, <http://dx.doi.org/10.1016/j.protcy.2014.09.007>.
  26. Alex Krizhevsky, Learning Multiple Layers of Features from Tiny Images, Science Department, University of Toronto, 2009, pp. 1–60, URL <https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf>.
  27. John Duchi, Elad Hazan, Yoram Singer, Adaptive subgradient methods for online learning and stochastic optimization, *J. Mach. Learn. Res.* 12 (2011) 2121–2159, <http://dx.doi.org/10.1109/CDC.2012.6426698>, URL <http://jmlr.org/papers/v12/duchi11a.html>.
  28. Rich Caruana, Alexandru Niculescu-Mizil, An empirical comparison of supervised learning algorithms, in: Proceedings of the 23rd International Conference on Machine Learning, vol. C(1), 2006, pp. 161–168, URL <http://portal.acm.org/citation.cfm?doid=1143844.1143865>.
  29. Rich Caruana, Nikos Karampatziakis, Ainur Yessenalina, An empirical evaluation of supervised learning in high dimensions, in: Proceedings of the 25th International Conference

- on Machine Learning, 2008, pp. 96–103, URL <http://portal.acm.org/citation.cfm?doid=1390156.1390169>.
- 30. Chintan Parmar, Patrick Grossmann, Johan Bussink, Philippe Lambin, Hugo J.W.L. Aerts, Machine learning methods for quantitative radiomic biomarkers (Supplement), *Sci. Rep.* 5 (2015) 13087, <http://dx.doi.org/10.1038/srep13087>.
  - 31. Prateek Prasanna, Pallavi Tiwari, Anant Madabhushi, Co-occurrence of local anisotropic gradient orientations (CoLiAGE): distinguishing tumor confounders and molecular subtypes on MRI, *Lect. Notes Comput. Sci.* 8675 (Part 3) (2014) 73–80, [http://dx.doi.org/10.1007/978-3-319-10443-0\\_10](http://dx.doi.org/10.1007/978-3-319-10443-0_10).
  - 32. Dennis Gabor, Theory of communication. Part 1: The analysis of information, *J. Inst. Electr. Eng., Part 3, Radio Commun. Eng.* 93 (26) (1946) 429–441, <http://dx.doi.org/10.1049/ji-3-2.1946.0074>, URL [http://ieeexplore.ieee.org/xpls/abs\\_all.jsp?arnumber=5298517](http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5298517).
  - 33. Robert Haralick, K. Shanmugan, I. Dinstein, Textural features for image classification, URL <http://dceanalysis.bigr.nl/Haralick73-Texturalfeaturesforimageclassification.pdf>, 1973.
  - 34. K.I. Laws, Textured Image Segmentation, PhD thesis, Univ. of Southern California, 1980.
  - 35. George Azzopardi, Nicolai Petkov, A CORF computational model of a simple cell that relies on LGN input outperforms the Gabor function model, *Biol. Cybern.* 106 (3) (2012) 177–189, <http://dx.doi.org/10.1007/s00422-012-0486-6>.
  - 36. L. Wang, Texture unit, textural spectrum and texture analysis, *IEEE Trans. Geosci. Remote Sens.* 28 (4) (1990) 509, <http://dx.doi.org/10.1109/TGRS.1990.572934>.
  - 37. Arnout C. Ruifrok, Ruth L. Katz, Dennis A. Johnston, Comparison of quantification of histochemical staining by hue–saturation–intensity (HSI) transformation and color-deconvolution, *Appl. Immunohistochem. Mol. Morphol.* 11 (1) (2003) 85–91, <http://dx.doi.org/10.1097/00129039-200303000-00014>.
  - 38. Andrew Janowczyk, Scott Doyle, Hannah Gilmore, Anant Madabhushi, A resolution adaptive deep hierarchical (RADHiCaL) learning scheme applied to nuclear segmentation of digital pathology images, *Comput. Methods Biomed. Eng.* 1163 (April) (2016) 1–7, <http://dx.doi.org/10.1080/21681163.2016.1141063>, URL <http://www.tandfonline.com/doi/full/10.1080/21681163.2016.1141063>.
  - 39. D.M.W. Powers, Evaluation: from precision, recall and F-measure to ROC, informedness, markedness & correlation, *J. Mach. Learn. Technol.* 2 (1) (2011) 37–63.
  - 40. Marie-Pierre Dubuisson, Anil K. Jain, A modified Hausdorff distance for object matching, *Proc. Int. Conf. Pattern Recognit.* (1994) 566–568, <http://dx.doi.org/10.1109/ICPR.1994.576361>.

This page intentionally left blank

# Deformable MR Prostate Segmentation via Deep Feature Learning and Sparse Patch Matching

9

**Yanrong Guo, Yaozong Gao, Dinggang Shen***University of North Carolina at Chapel Hill, Chapel Hill, NC, United States*

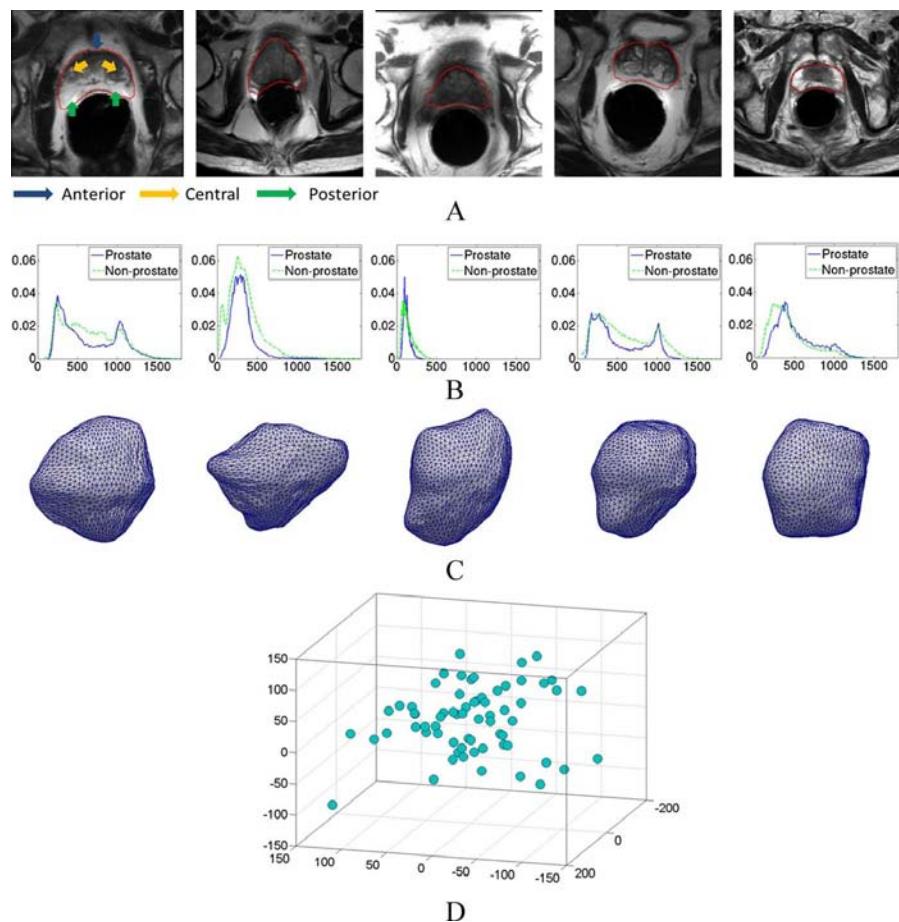
## CHAPTER OUTLINE

<b>9.1</b>	<b>Background</b>	197
<b>9.2</b>	<b>Proposed Method</b>	199
9.2.1	Related Work	199
9.2.2	Learning Deep Feature Representation	201
9.2.3	Segmentation Using Learned Feature Representation	206
<b>9.3</b>	<b>Experiments</b>	211
9.3.1	Evaluation of the Performance of Deep-Learned Features	212
9.3.2	Evaluation of the Performance of Deformable Model	216
<b>9.4</b>	<b>Conclusion</b>	219
<b>References</b>		219

## 9.1 BACKGROUND

Prostate cancer is the second leading cause of cancer death in American men, behind only lung cancer [1]. As the main imaging modality for clinical inspection of the prostate, Magnetic Resonance (MR) imaging provides better soft tissue contrast than ultrasound imaging in a non-invasive way and has an emerging role in prostate cancer diagnosis and treatment [2,3]. The accurate localization of the prostate is an important step for assisting diagnosis and treatment, such as for guiding biopsy procedure [2] and radiation therapy [3]. However, manual segmentation of the prostate is tedious and time-consuming, and it also suffers from intra- and inter-observer variability. Therefore, developing automatic and reliable segmentation methods for MR prostate is clinically desirable and an important task.

However, accurate prostate localization in MR images is difficult due to the following two main challenges. First, the appearance patterns vary a lot around the prostate boundary across patients. As we can see from Fig. 9.1A, the image contrasts

**FIGURE 9.1**

(A) Typical T2-weighted prostate MR images. Red contours indicate the prostate glands manually delineated by an expert. (B) Intensity distributions of the prostate and background voxels around the prostate boundary of (A). (C) The 3D illustrations of prostate surfaces corresponding to each image in (A). (D) The prostate shape distribution obtained from PCA analysis.

at different prostate regions, i.e., the anterior, central, and posterior regions, change across different subjects and also within each subject. Fig. 9.1B gives the intensity distributions of the prostate and background voxels around the prostate boundary, respectively. As shown in the figure, the intensity distributions vary highly across different patients and often do not follow the Gaussian distribution.

To evaluate the shape difference in our dataset, we adopt the PCA analysis by mapping each high-dimensional shape vector onto a space spanned by the first three

principal components. Note that the shape vector is formed by the concatenation of all vertex coordinates, and then linearly aligned to the mean shape before PCA analysis. Fig. 9.1D shows the distribution of 66 prostate shapes, from which we can see inter-patient shape variation in the shape repository.

---

## 9.2 PROPOSED METHOD

### 9.2.1 RELATED WORK

Most of recent studies in T2-weighted MR prostate segmentation focus on two types of methods: multi-atlas-based [4–7] and deformable-model-based [8,9] segmentation methods. Multi-atlas-based methods are widely used in medical imaging [10–12]. Most of research focuses on the design of sophisticated atlas-selection or label-fusion methods. For example, Yan et al. [5] proposed a label image constrained atlas selection and label fusion method for prostate MR segmentation. During the atlas selection procedure, label images are used to constrain the manifold projection of intensity images; by doing so, the misleading projection can be potentially avoided due to the use of other anatomical structures. Ou et al. [7] proposed an iterative multi-atlas label fusion method by gradually improving the registration based on the prostate vicinity between the target and atlas images. For deformable-model-based methods, Toth [8] proposed to incorporate different features in the context of AAMs (Active Appearance Models). Besides, with the adoption of a level set, the issue of landmark correspondence can be avoided.

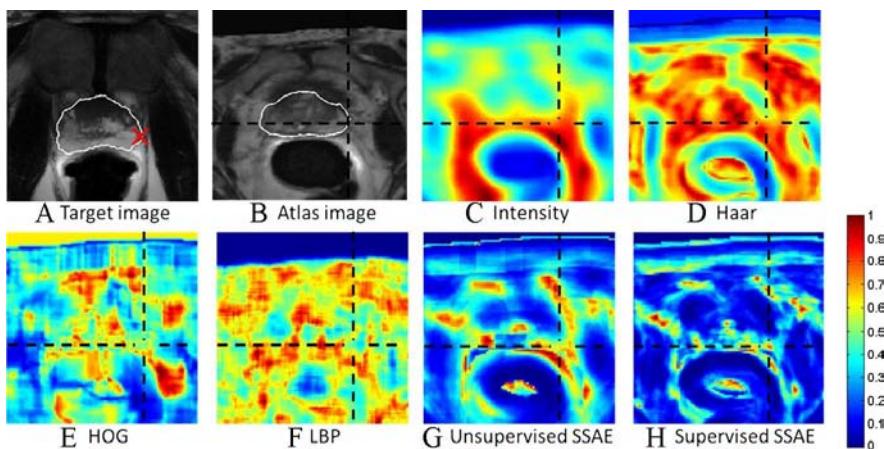
**Handcrafted vs. Deep-Learned Features.** Both types of above-mentioned methods require careful feature engineering to achieve good performance. The multi-atlas based methods require good features for identifying correspondences between a new testing image and each atlas image [13], while the deformable model relies on discriminative features for separating the target object (e.g., the prostate) from the background [14]. Traditionally, intensity patch is often used as features for the above two methods [15,16]. However, due to the inhomogeneity of MR images, the simple intensity features often fail in the segmentation of MR images with different contrasts and illuminations. To overcome this problem, recent MR prostate segmentation methods started to use features that are specifically designed for vision tasks, such as gradient [17], Haar-like wavelets [18], Histogram of Oriented Gradients (HOG) [19], SIFT [20], Local Binary Patterns (LBP) [21], and variance adaptive SIFT [14]. Compared to simple intensity features, these vision-based features show better invariance to illumination, and they also provide certain invariance to small rotations. In [22], the authors showed that better prostate segmentation could be obtained by using the combination of these features.

One major limitation of the aforementioned handcrafted features is the inability of adapting themselves to the data at hand. This means that the representation power and effectiveness of these features could vary across different kinds of image data. To deal with this limitation, the learning based feature representation methods [23,24] have been developed to extract latent information that can be adapted to the data at

hand. As one important type of feature learning methods, deep learning has recently become a hot topic in machine learning [23], computer vision [25], and many other research fields, including medical image analysis [26]. Compared with handcrafted features, which need expert knowledge for careful design and also lack sufficient generalization power to different domains, deep learning is able to automatically learn effective feature hierarchies from the data. Therefore, it draws an increasing interest in the research communities. For example, Vincent et al. [27] showed that the features learned by deep belief network and the stacked denoising auto-encoder beat the state-of-the-art handcrafted features for the digit classification problem in the MINST dataset. Farabet et al. [28] proposed to use the convolutional neural network to learn useful feature representations, which are more powerful in the application of scene labeling than the engineered features, and their method also achieved the state-of-the-art performance. In the field of medical image analysis, Shin et al. [29] applied the stacked auto-encoders to organ identification in MR images, which shows the potential of deep learning methods in analysis of medical images. In summary, compared with handcrafted features, deep learning has the following advantages: (i) Instead of designing effective features for a new task by trial and error, deep learning largely saves researchers' time by automating this process. Also, it is capable of exploiting the complex feature patterns, for which the manual feature engineering is not good. (ii) Unlike the handcrafted features, which are usually shallow in feature representation due to the difficulty of designing the high-level abstract features, deep learning is able to learn the feature hierarchy in a layer-by-layer manner, by first learning the low-level features and then recursively building more comprehensive high-level features based on the previously learned low-level features. (iii) When unsupervised pre-training is combined with supervised fine-tuning, the deep-learned features can be optimized for a certain task, such as segmentation, thus boosting the final performance. Motivated by the above factors, we propose to learn a hierarchical feature representation based on deep feature learning [30,31] from MR prostate images.

Before presenting the proposed deep feature learning, we first demonstrate the limitations of handcrafted features in MR prostate images. In computer vision, the common handcrafted features include Haar features [18], HOG features [20], and Local Binary Patterns [21]. They have been proposed in different applications, with promising results such as in object detection of natural images. However, these features are not suitable for MR prostate images, as they are not invariant to both the inhomogeneity of MR images and the appearance variations of the prostate gland. To illustrate limitations of these handcrafted features, we compare the effectiveness of different features in identifying correspondences in two images, in the context of multi-atlas based segmentation methods.

[Fig. 9.2](#) shows a typical example by computing the similarity maps between one point (shown as a red cross in [Fig. 9.2A](#)) in the target image ([Fig. 9.2A](#)) and all points in an aligned atlas image ([Fig. 9.2B](#)). The white contours in (A) and (B) show the prostate boundaries, and the black dashed cross in [Fig. 9.2B](#) indicates correct correspondence of the red-cross target point in the atlas image. The effectiveness of features can be reflected by the similarity map. If features are distinctive for corre-



**FIGURE 9.2**

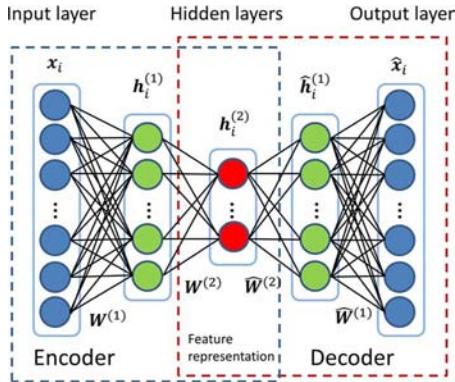
The similarity maps computed between a reference voxel (red cross) in the target image (A) and all voxels in the atlas image (B) by the four handcrafted feature representations, i.e., intensity (C), Haar (D), HOG (E), and LBP (F), as well as the two deep-learned feature representations, namely the unsupervised SSAE (G) and the supervised SSAE (H). White contours indicate the prostate boundaries, and the black dashed crosses indicate the ground-truth correspondence in (B), which is corresponding to the red cross in (A).

spondence detection, the similarity computed by using these features would be high for correct correspondences while low for incorrect correspondences. Figs. 9.2C–F show the similarity maps computed using different handcrafted features, such as intensity patch features, Haar features, HOG features, and LBP features, respectively. It is clear that none of these features can capture correct correspondence, as the similarity between the corresponding voxels indicated by red crosses is low, compared to that of nearby voxels. This shows that the existing handcrafted features are insufficient in multi-atlas based segmentation for the MR prostate.

Thus, to relieve the limitation of handcrafted features, it is necessary to learn discriminant features adaptive to MR prostate images. To demonstrate the effectiveness of deep learning features, Figs. 9.2G and 9.2H provide the similarity maps computed using the two kinds of deep-learned features obtained by our proposed *unsupervised* and *supervised* stacked sparse auto-encoder (SSAE), respectively. Compared to the similarity maps of handcrafted features, it is clear that the correct correspondence can be better identified with the deep-learned features, especially for the supervised SSAE.

### 9.2.2 LEARNING DEEP FEATURE REPRESENTATION

**Auto-Encoder.** Serving as the fundamental component for SSAE, the basic auto-encoder (AE) trains a feed-forward nonlinear neural network, which contains three

**FIGURE 9.3**

Construction of a basic AE.

layers, i.e., input layer, hidden layer, and output layer, as illustrated in Fig. 9.3. Each layer is represented by a number of nodes. Blue nodes on the left and right sides of Fig. 9.3 indicate the input and output layers, respectively, and green nodes indicate the hidden layer. Nodes in the two neighboring layers are fully connected, which means that each node in the previous layer can contribute to any node in the next layer. Basically, AE consists of two steps, namely encoding and decoding. In the encoding step, AE encodes the input vector into a concise representation through connections between input and hidden layers. In the decoding step, AE tries to reconstruct the input vector from the encoded feature representation in the hidden layer. The goal of AE is to find a concise representation of the input data, which could be used for the purpose of best reconstruction. Since we are interested in the representation of image patches, in this application the input to AE is an image patch, which is concatenated as a vector. In the training stage, given a set of training patches  $\mathbf{X} = \{\mathbf{x}_i \in \mathbb{R}^L, i = 1, \dots, N\}$ , where  $N$  and  $L$  are the number and the dimension of training patches, respectively, AE automatically learns the weights of all connections in the network by minimizing the reconstruction error in Eq. (9.1),

$$\underset{\mathbf{W}, \mathbf{b}, \hat{\mathbf{W}}, \hat{\mathbf{b}}}{\operatorname{argmin}} \sum_{i=1}^N \|\mathbf{x}_i - (\hat{\mathbf{W}}(\sigma(\mathbf{W}\mathbf{x}_i + \mathbf{b})) + \hat{\mathbf{b}})\|_2^2 \quad (9.1)$$

where  $\mathbf{W}, \mathbf{b}, \hat{\mathbf{W}}, \hat{\mathbf{b}}$  are the parameters in the AE network, and  $\sigma(\mathbf{a}) = (1 + \exp(-\mathbf{a}))^{-1}$ . Given an input vector  $\mathbf{x}_i$ , AE first encodes it into the concise representation  $\mathbf{h}_i = \sigma(\mathbf{W}\mathbf{x}_i + \mathbf{b})$ , where  $\mathbf{h}_i$  is the responses of  $\mathbf{x}_i$  at the hidden nodes, and the dimension of  $\mathbf{h}$  equals to the number of nodes in the hidden layer. In the next step, AE tries to decode the original input from the encoded representation, i.e., with  $\hat{\mathbf{W}}\mathbf{h}_i + \hat{\mathbf{b}}$ . To learn effective features for the input training patches, AE requires the dimension of the hidden layer to be less than that of the input layer. Otherwise, the

minimization of Eq. (9.1) would lead to trivial solutions, e.g., identity transformation. Studies [32] have also shown that the basic AE learns very similar features as PCA.

Once the weights  $\{\mathbf{W}, \mathbf{b}, \hat{\mathbf{W}}, \hat{\mathbf{b}}\}$  have been learned through the training patches, in the testing stage the AE could efficiently obtain a concise feature representation for a new image patch  $\mathbf{x}_{\text{new}}$  by a forward passing step, i.e.,  $\mathbf{h}_{\text{new}} = \sigma(\mathbf{W}\mathbf{x}_{\text{new}} + \mathbf{b})$ .

**Sparse Auto-Encoder.** Rather than limiting the dimension of hidden layer (i.e., feature representation), an alternative could be imposing regularization on the hidden layer. Sparse auto-encoder (SAE) falls into this category. Instead of requiring the dimension of hidden layer to be less than that of the input layer, SAE imposes a sparsity regularization on the responses of hidden nodes (i.e.,  $\mathbf{h}$ ) to avoid the problem of trivial solutions suffered by the basic AE. Specifically, SAE enforces the average response of each hidden node over the training set to be infinitesimal, i.e.,  $\rho^j = \sum_{i=1}^N \mathbf{h}_i^j \approx \rho$ , where  $\mathbf{h}_i^j$  is the response of the  $i$ th training input at hidden node  $j$ , and  $\rho$  is a very small constant. In this way, to balance both the reconstruction power and the sparsity of the hidden layer, only a few useful hidden nodes could have responses for each input, thus forcing the SAE network to learn sparse feature representation of the training data. Mathematically, we can extend Eq. (9.1) to derive the objective function of SAE by adding a sparsity constraint term shown below:

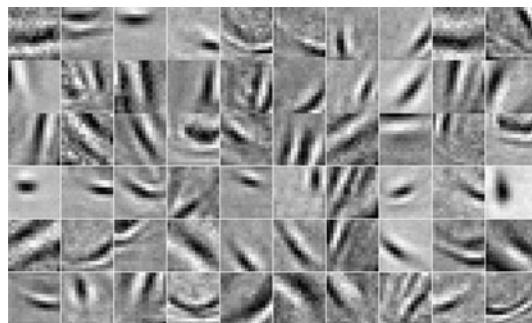
$$\underset{\mathbf{W}, \mathbf{b}, \hat{\mathbf{W}}, \hat{\mathbf{b}}}{\operatorname{argmin}} \sum_{i=1}^N \|\mathbf{x}_i - (\hat{\mathbf{W}}(\sigma(\mathbf{W}\mathbf{x}_i + \mathbf{b}) + \hat{\mathbf{b}})\|_2^2 + \delta \sum_{j=1}^M KL(\rho \mid \rho^j) \quad (9.2)$$

$$KL(\rho \mid \rho^j) = \rho \log \frac{\rho}{\rho^j} + (1 - \rho) \log \frac{1 - \rho}{1 - \rho^j}$$

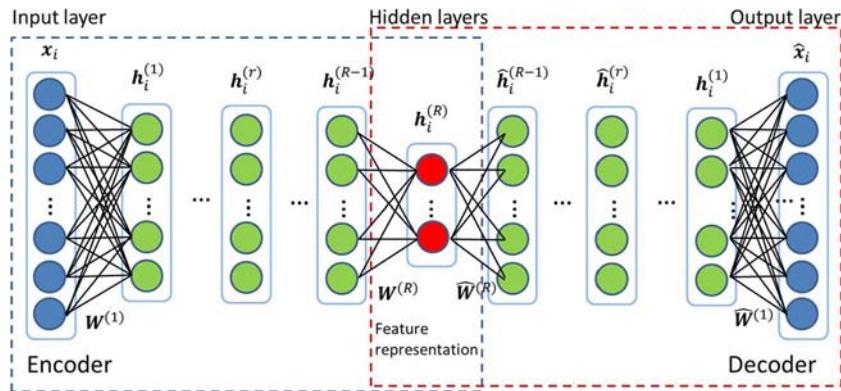
where  $\delta$  is a parameter to balance between reconstruction and sparsity terms, and  $M$  is the number of hidden nodes.  $KL(\rho \mid \rho^j)$  is the Kullback–Leibler divergence between two Bernoulli distributions with probability  $\rho$  and  $\rho^j$ . As we can see, the sparsity term is minimized only when  $\rho^j$  is close to  $\rho$  for every hidden node  $j$ . Since  $\rho$  is set to be a small constant, minimizing Eq. (9.2) could lead to the sparse responses of hidden nodes, thus the sparsity of learned feature representation.

**Stacked Sparse Auto-Encoder.** By using SAE, we can learn the low-level features (such as Gabor-like features as shown in Fig. 9.4) from the original data (MR image patches). However, low-level features are not good enough due to large appearance variations of the MR prostate. It is necessary to learn abstract high-level features, which could be invariant to the inhomogeneity of MR images. Motivated by the human perception, which constitutes a deep network to describe concepts in a hierarchical way using multiple levels of abstraction, we recursively apply SAE to learn more abstract/high-level features based on the features learned from the low-level. This multi-layer SAE model is referred to as a stacked sparse auto-encoder (SSAE), which stacks multiple SAEs on top of each other for building deep hierarchies.

Fig. 9.5 shows a typical SSAE with  $R$  stacked SAEs. Let  $\mathbf{W}^{(r)}$ ,  $\mathbf{b}^{(r)}$ ,  $\hat{\mathbf{W}}^{(r)}$ , and  $\hat{\mathbf{b}}^{(r)}$  denote the connection weights and intercepts between the input layer and

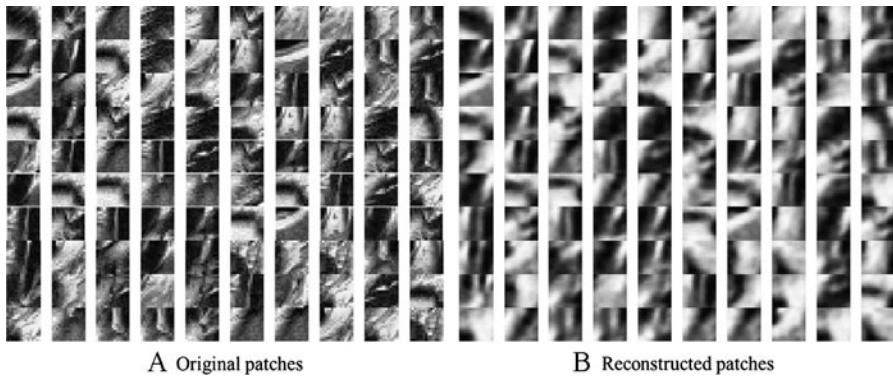
**FIGURE 9.4**

The low-level feature representation learned from the SAE. Here, we reshape each row in  $\mathbf{W}$  into the size of image patch, and visualize only its first slice as an image filter.

**FIGURE 9.5**

Construction of the *unsupervised* SSAE with  $R$  stacked SAEs.

hidden layer, and the weights and intercepts between the hidden layer and output layer in the  $r$ th SAE, respectively. In the encoding part of the SSAE, the input vector  $x_i$  is first encoded by the first SAE for obtaining the low-level representation  $\mathbf{h}_i^{(1)}$ , i.e.,  $\mathbf{h}_i^{(1)} = \sigma(\mathbf{W}^{(1)}\mathbf{x}_i + \mathbf{b}^{(1)})$ . Then, the low-level representation  $\mathbf{h}_i^{(1)}$  of the first SAE is considered as the input vector to the next SAE, which encodes it into higher level representation  $\mathbf{h}_i^{(2)}$ , i.e.,  $\mathbf{h}_i^{(2)} = \sigma(\mathbf{W}^{(2)}\mathbf{h}_i^{(1)} + \mathbf{b}^{(2)})$ . Generally, the  $r$ th level representation  $\mathbf{h}_i^{(r)}$  can be obtained by a recursive encoding procedure  $\mathbf{h}_i^{(r)} = \sigma(\mathbf{W}^{(r)}\mathbf{h}_i^{(r-1)} + \mathbf{b}^{(r)})$  with  $\mathbf{h}_i^{(0)} = \mathbf{x}_i$ . Similarly, the decoding step of SSAE recursively reconstructs the input of each SAE. In this example, SSAE first reconstructs the low-level representation  $\hat{\mathbf{h}}_i^{(r-1)}$  from the high-level representation  $\hat{\mathbf{h}}_i^{(r)}$ , i.e.,  $\hat{\mathbf{h}}_i^{(r-1)} = \hat{\mathbf{W}}^{(r)}\hat{\mathbf{h}}_i^{(r)} + \hat{\mathbf{b}}^{(r)}$  with  $\hat{\mathbf{h}}_i^{(R)} = \mathbf{h}_i^{(R)}$  for  $r = R, \dots, 2$ . Then, using the

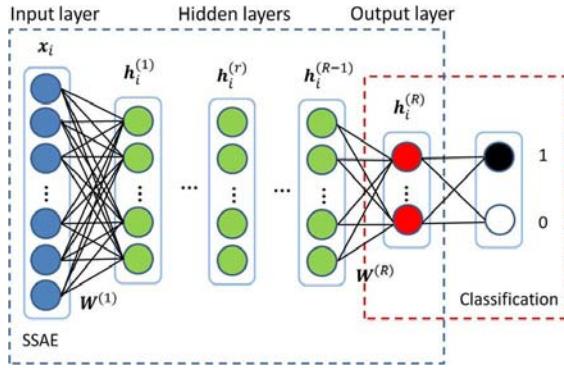
**FIGURE 9.6**

Typical prostate image patches (A) and their reconstructions (B) by using the unsupervised SSAE with four stacked SAEs.

reconstructed low-level representation  $\hat{\mathbf{h}}_i^{(1)}$ , the original input vector could be estimated, i.e.,  $\hat{\mathbf{x}}_i = \hat{\mathbf{W}}^{(1)}\hat{\mathbf{h}}_i^{(1)} + \hat{\mathbf{b}}^{(1)}$ .

After stacking multiple SAEs together by feeding the output layer from the low-level SAE as the input layer of a high-level SAE, SSAE is able to extract more useful and general high-level features. In the optimization of SSAE, this deep architecture is first pre-trained in an unsupervised layer-wise manner and then fine-tuned by the back propagation. Since the aforementioned SSAE network is trained based only on the original image patches, without using the supervised label information, it is denoted as the *unsupervised SSAE*. Fig. 9.6 shows some typical prostate image patches and their reconstructions by the unsupervised SSAE with  $R = 4$ .

However, since the unsupervised SSAE trains the whole network on the unlabeled data, the high-level features learned from unsupervised SSAE are only data-adaptive, that is, not necessarily discriminative to separate prostate and background voxels. To make the learned feature representation discriminative [33,34], the supervised fine-tuning is often adopted by stacking another classification output layer on the top of the encoding part of the SSAE, as shown in the red dashed box of Fig. 9.7. This top layer is used to predict the label likelihood of the input data  $\mathbf{x}_i$  by using the features learned from the most high-level representation  $\mathbf{h}_i^{(R)}$ . The number of nodes in the classification output layer equals to the number of labels (i.e., “1” denotes the prostate, and “0” denotes the background). Using the optimized parameters from the pre-training of SSAE as initialization, the entire neural network (Fig. 9.7) can be further fine-tuned by back-propagation to maximize the classification performance. This tuning step is referred to as the supervised fine-tuning, in contrast with the unsupervised fine-tuning mentioned before. Accordingly, the entire deep network is referred to as the *supervised SSAE*. Fig. 9.8 gives a visual illustration of typical feature representations of the first and second hidden layers learned by a four-layer supervised

**FIGURE 9.7**

Construction of the supervised SSAE with a classification layer, which fine-tunes the SSAE with respect to the task of voxel-wise classification between prostate (label = 1) and background (label = 0).

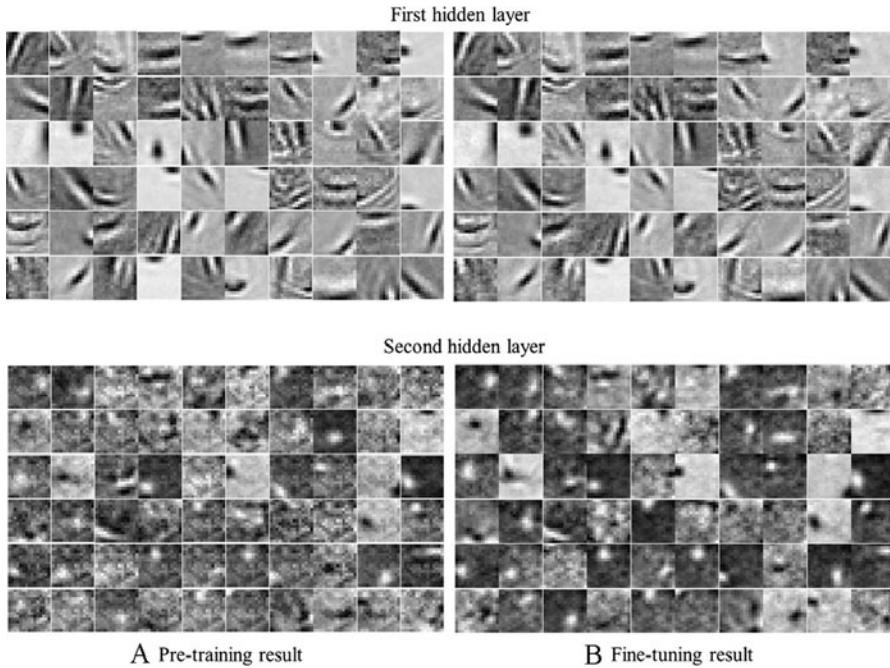
SSAE based on the visualization method in [35]. Here, Figs. 9.8A and 9.8B show the visualization of 60 units obtained from the first and second hidden layers under unsupervised pre-training (with unlabeled image patches) and supervised fine-tuning (with labeled image patches), respectively. It can be seen that higher hidden layer tends to be more affected by the classification layer we introduced.

After learning all the parameters  $\{\mathbf{W}^{(r)}, \hat{\mathbf{W}}^{(r)}, \mathbf{b}^{(r)}, \hat{\mathbf{b}}^{(r)}\}$  of SSAE ( $r = 1, \dots, R$ ), where  $R$  denotes the number of stacked SAEs, the high-level representations of a new image patch  $\mathbf{x}_{\text{new}}$  can be efficiently obtained by a forward pass, i.e.,  $\mathbf{h}_{\text{new}}^r = \sigma(\mathbf{W}^{(r)} \mathbf{h}_{\text{new}}^{(r-1)} + \mathbf{b}^{(r)})$  with  $\mathbf{h}_{\text{new}}^0 = \mathbf{x}_{\text{new}}$  for  $r = 1, \dots, R$ . The final high-level representation  $\mathbf{h}_{\text{new}}^R$  will be used as features to guide the sparse patch matching (Section 9.2.3), and propagate labels from atlas images to the target image for estimating the prostate likelihood map.

### 9.2.3 SEGMENTATION USING LEARNED FEATURE REPRESENTATION

**Sparse Patch Matching with the Deep-Learned Features.** Before sparse patch matching, all atlas images are registered to the target image. This registration includes two steps. First, linear registration is applied for initial alignment, with the guidance from the landmarks automatically detected around the prostate region [36]. Then, the free-form deformation (FFD) [37] is further adopted to the linearly-aligned images for deformable registration.

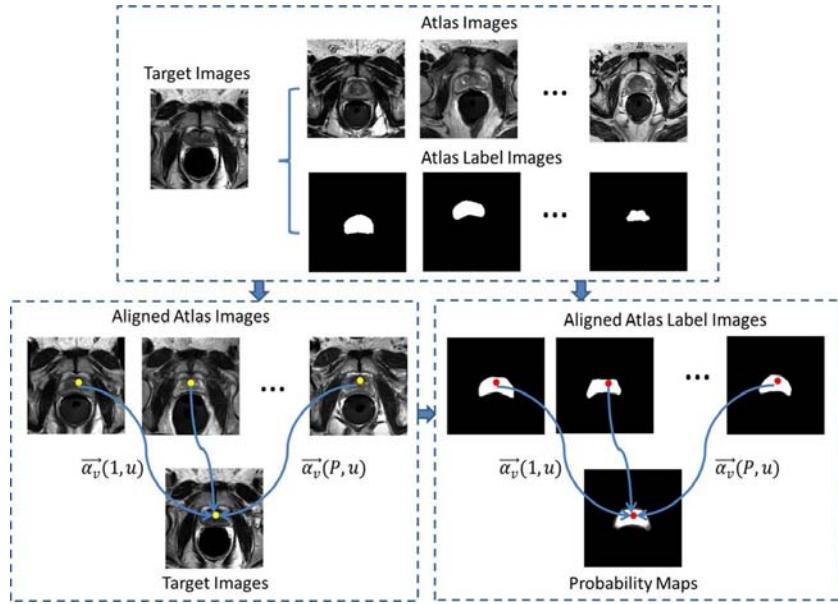
After learning the SSAE networks (either in unsupervised or supervised manner), each new image patch in the testing stage can be encoded as a high-level feature vector (i.e., the last hidden layer of the SSAE). These features can be fed into a segmentation framework for labeling voxels as either the prostate or the background. As one of the popular segmentation frameworks, multi-atlas based segmentation demon-

**FIGURE 9.8**

Visualization of typical feature representations of the first hidden layer (first row) and second hidden layer (second row) for the unsupervised pre-training (A) and supervised fine-tuning (B), respectively.

strates its effectiveness on dealing with image variations in different applications [38,39]. However, traditionally the multi-atlas based segmentation adopts only the intensity or handcrafted features for measuring the similarity between different local patches, or computing the weights of different patches during label propagation. Since MR prostate images exhibit large structural and appearance variations, we propose to incorporate the deep learning features, instead of the conventional handcrafted features, into the multi-atlas segmentation framework. As the features extracted by the deep learning methods are usually more robust and discriminative, the performance of multi-atlas segmentation can be improved at the same time. Fig. 9.9 gives the general description of our multi-atlas based method, called sparse patch matching. In this method, instead of computing pair-wise intensity similarity as a matching weight, we propose to select only a small number of similar atlas patches by sparse representation, which is more robust to outliers. In the following, we give the detailed description of our sparse patch matching method.

In order to estimate the prostate likelihood map of a target image  $I_s$ , we first align all the atlas images  $\{I_p, p = 1, \dots, P\}$  and their label maps  $\{G_p, p = 1, \dots, P\}$  onto the target image  $I_s$ . Then, to determine the prostate likelihood of a particular

**FIGURE 9.9**

The schematic description of sparse patch matching.

voxel  $v$  in the target image  $I_s$ , we first extract the image patch centered at voxel  $v$  from the target image, and then all image patches within a certain searching neighborhood  $\mathbb{N}(v)$  across all the aligned atlas images.

Next, the deep-learned feature representations for those extracted intensity patches are obtained through the encoding procedure of the learned SSAE as introduced in Section 9.2.2. Denote  $f_s(v)$  as the deep-learned features for the target image patch at point  $v$ , and denote  $A_v$  as the feature matrix resulted from column-wise combination of deep-learned features of atlas patches, i.e.,  $A_v = [f_p(u) \mid p = 1, \dots, P; u \in \mathbb{N}(v)]$ . To estimate the prostate likelihood  $Q_s(v)$  for voxel  $v$  in the target image  $I_s$ , we linearly combine the label of each voxel  $u \in \mathbb{N}(v)$  from each atlas image  $I_p$  with a weighting vector  $\alpha_v = [\alpha_v(p, u)]_{u=1, \dots, |\mathbb{N}(v)|; p=1, \dots, P}$  as follows:

$$Q_s(v) = \frac{\sum_{p=1}^P \sum_{u \in \mathbb{N}(v)} \alpha_v(p, u) \times G_p(u)}{\sum_{p=1}^P \sum_{u \in \mathbb{N}(v)} \alpha_v(p, u)}. \quad (9.3)$$

According to Eq. (9.3), it is easy to see that the robustness and accuracy of prostate likelihood estimation depend on how well the weighting vector  $\alpha_v$  is determined. In the literature, different weight estimation methods have been proposed [40,41]. Most multi-atlas based segmentation methods directly compute  $\alpha_v$  as the pair-wise similarity between intensity patches, such as using the Euclidean distance. In our method, we compute the weighting vector  $\alpha_v$  different from the previous methods in respect to

the following two aspects. *First*, instead of using the intensity or handcrafted features, the high-level features are learned from the deep learning architecture. *Second*, with the help of recently proposed sparse representation method [4], we enforce sparsity constraint upon the weighting vector  $\alpha_v$ . In this way, we seek for the best representation of the target patch using a limited set of similar atlas patches. Mathematically, the optimization of  $\alpha_v$  can be formulated as the sparse representation problem below:

$$\alpha_v = \arg \min_{\alpha_v} \frac{1}{2} \|f_s(v) - A_v \alpha_v\|_2^2 + \eta \|\alpha_v\|_1 \quad \text{s.t. } \alpha_v \geq 0 \quad (9.4)$$

The first term is the data fitting term, which measures difference between the target feature vector  $f_s(v)$  and the linearly combined feature representation  $A_v \alpha_v$  from all atlas image patches. The second term is the sparsity term, which attributes to the sparsity property of the weighting vector  $\alpha_v$ .  $\eta$  controls the strength of sparsity constraint on the weighting vector  $\alpha_v$ . If  $\eta$  is larger, the number of nonzero elements in  $\alpha_v$  will be smaller. In this way, only a few patches in patch dictionary  $A_v$  will be selected to reconstruct the target features  $f_s(v)$  in a nonparametric fashion, thus reducing the risk of including those misleading atlas patches in the likelihood estimation.

Based on the derived weighting vector  $\alpha_v$ , the prostate likelihood  $Q_s(v)$  for a target point  $v$  can be estimated by Eq. (9.3). Since the weighting vector  $\alpha_v$  is sparse, the prostate likelihood  $Q_s(v)$  is finally determined by the linear combination of labels corresponding to atlas patches with nonzero elements in vector  $\alpha_v$ . After estimating the prostate likelihood for all voxels in the target image  $I_s$ , a likelihood map  $Q_s$  is generated, which can be used to robustly locate the prostate region (as shown in Fig. 9.9). Usually, a simple thresholding or level set method [42,43] can be applied to binarize the likelihood map for segmentation. However, since each voxel in the target image is independently estimated in the multi-atlas segmentation method, the final segmentation could be weird as no shape prior is considered. To robustly and accurately estimate the final prostate region from the prostate likelihood map, it is necessary to take into account the prostate shape prior during the segmentation.

**Data and Shape Driven Deformable Model.** The main purpose of this section is to segment the prostate region based on the prostate likelihood map estimated in the previous section. The likelihood map can be used in two aspects of deformable model construction. First, the initialization of deformable model can be easily built by thresholding the likelihood map. In this way, the limitation of model initialization problem in the traditional deformable segmentation can be naturally relieved. Second, the likelihood map can be used as the appearance force to drive the evolution of the deformable model. Besides, in order to deal with large inter-patient shape variation, we propose to use sparse shape prior for deformable model regularization.

Here, our deformable model is represented by a 3D surface, which is composed of  $K$  vertices  $\{\mathbf{d}_k \mid k = 1, \dots, K\}$ . After concatenating these  $K$  vertices  $\{\mathbf{d}_k \mid k = 1, \dots, K\}$  into a vector  $\mathbf{d}$ , each deformable model can be represented as a shape vector with length of  $3 \cdot K$ . Let  $D$  denote a large shape dictionary that includes prostate shape vectors of all training subjects. Each column of shape dictionary  $D$

corresponds to the shape vector of one subject. The shape dictionary can be used as a shape prior to constrain the deformable model in a learned shape space. Instead of assuming the Gaussian distribution of shapes and then simply using PCA for shape modeling as done in the Active Shape Model [44], we adopt a recently proposed method, named sparse shape composition [45], for shape modeling. In the sparse shape composition, the shapes are sparsely represented by shape instances in the shape dictionary without the need of the Gaussian assumption. Specifically, given a new shape vector  $\mathbf{d}$  and shape dictionary  $\mathbf{D}$ , sparse shape composition method reconstructs shape vector  $\mathbf{d}$  as the sparse representation of shape dictionary  $\mathbf{D}$  by minimizing the following objective function:

$$(\boldsymbol{\varepsilon}, \psi) = \arg \min_{\boldsymbol{\varepsilon}, \psi} \|\psi(\mathbf{d}) - \mathbf{D}\boldsymbol{\varepsilon}\|_2^2 + \mu\|\boldsymbol{\varepsilon}\|_1 \quad (9.5)$$

where  $\psi(\mathbf{d})$  denotes the target shape  $\mathbf{d}$  that is affine aligned onto the mean shape space of shape dictionary  $\mathbf{D}$ .  $\boldsymbol{\varepsilon}$  indicates the sparse coefficient for the linear shape combination. Once  $(\boldsymbol{\varepsilon}, \psi)$  are estimated by Eq. (9.5), the regularized shape can be computed by  $\psi^{-1}(\mathbf{D}\boldsymbol{\varepsilon})$ , where  $\psi^{-1}$  is the inverse affine transform of  $\psi$ .

For each target image, the segmentation task is formulated as the deformable model optimization problem. During the optimization procedure, each vertex of deformable model  $d_k$  is driven iteratively by the information from both prostate likelihood map and shape model until converged at the prostate boundaries. Mathematically, the evolution of the deformable model can be formulated as the minimization of an energy function, which contains a data energy  $E_{\text{data}}$  and a shape energy  $E_{\text{shape}}$  as in Eq. (9.6):

$$E = E_{\text{data}} + \lambda E_{\text{shape}}. \quad (9.6)$$

The data term  $E_{\text{data}}$  is used to attract the 3D surface toward the object boundary based on the likelihood map. Specifically, each vertex  $d_k$  is driven by the force related to the gradient vector of prostate likelihood map. Denote  $\nabla \vec{Q}_s(d_k)$  as the gradient vector at vertex  $d_k$  in the prostate likelihood map, and  $\vec{n}_s(d_k)$  as the normal vector on the vertex  $d_k$  of surface. When vertex  $d_k$  deforms exactly to the prostate boundary and also its normal direction aligns with the gradient direction of prostate boundary, the local matching term  $\langle \nabla \vec{Q}_s(d_k), \vec{n}_s(d_k) \rangle$  is maximized. In this way, we formulate to minimize the data energy  $E_{\text{data}}$  as

$$E_{\text{data}} = - \sum_k \langle \nabla \vec{Q}_s(d_k), \vec{n}_s(d_k) \rangle. \quad (9.7)$$

Since all the vertices on the deformable model are jointly evolved during the deformation, the matching of the deformable model with the prostate boundary will be robust to possible incorrect likelihood on some vertices, as well as inconsistency between neighboring vertices.

The shape term  $E_{\text{shape}}$  is used to encode the geometric property of the prostate shape based on the estimated coefficient  $\boldsymbol{\varepsilon}$  and the transformation  $\varphi$  in Eq. (9.5).

Specifically, the shape term is formulated as follows:

$$E_{\text{shape}} = \|\mathbf{d} - \psi^{-1}(\mathbf{D}\boldsymbol{\varepsilon})\|_2^2 + \beta \sum_k \left\| \mathbf{d}_k - \frac{\sum_{d_j \in \mathbb{N}(\mathbf{d}_k)} \mathbf{d}_k}{\sum_{d_j \in \mathbb{N}(\mathbf{d}_k)} 1} \right\|_2^2 \quad (9.8)$$

where the first term constrains the deformed shape  $\mathbf{d}$  to be close to the regularized shape  $\psi^{-1}(\mathbf{D}\boldsymbol{\varepsilon})$  by the sparse shape composition, and the second term imposes the smoothness constraint on shape, which prevents large deviations between each vertex  $\mathbf{d}_k$  and the center of its neighboring vertices  $d_j \in \mathbb{N}(\mathbf{d}_k)$ .

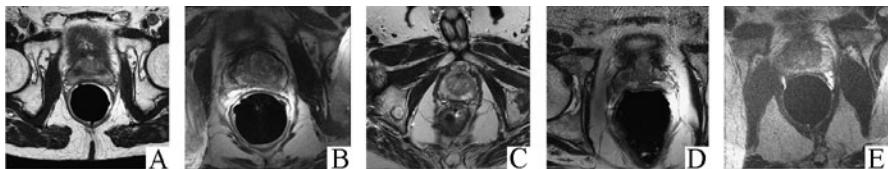
By combining Eq. (9.7) and Eq. (9.8) into Eq. (9.6), the vertices of the deformable model are iteratively driven toward the prostate boundary while constraining the shape in a nonparametric shape space.

### 9.3 EXPERIMENTS

We evaluate our method on the dataset, which includes 66 T2-weighted MR images from the University of Chicago Hospital. The images are acquired with 1.5 T magnetic field strength from different patients under different MR image scanners (34 images from Philips Medical Systems and 32 images from GE Medical Systems). Under this situation, the difficulty for the segmentation task increases since both shape and appearance differences are large. In Fig. 9.10, images B and E were acquired from a GE MRI scanner, while the other three were acquired from a Philips MRI scanner. As shown in Fig. 9.10, image C was obtained without the endorectal coil. It has a different prostate shape with other four images acquired with the endorectal coil. Besides, the prostate appearance suffers from the inhomogeneity (as in B and D) and noise (as in D and E), which further result in large variations. The image dimension and spacing are different from image to image. For example, the image dimension varies from  $256 \times 256 \times 28$  to  $512 \times 512 \times 30$ . The image spacing varies from  $0.49 \times 0.49 \times 3$  mm to  $0.56 \times 0.56 \times 5$  mm. The manual delineation of the prostate in each image is provided by a clinical expert as the ground truth for quantitative evaluation. As the preprocessing of the dataset, the bias field correction [46] and histogram matching [47] are applied to each image successively. We adopted a two-fold cross-validation. Specifically, in each case, the images of one fold are used for training the models, while the images of the other fold are used for testing the performance.

The parameters for deep feature learning are listed below. The patch size is  $15 \times 15 \times 9$ . The number of layers in SSAE framework is 4. The number of nodes in each layer of SSAE is 800, 400, 200, and 100, respectively. Thus, the deep-learned features have the dimensionality of 100. The target activation  $\rho$  for the hidden units is 0.15. The sparsity penalty  $\beta$  is 0.1. The Deep Learning Toolbox [48] is used for training our SSAE framework.

The searching neighborhood  $\mathbb{N}(v)$  is defined as the  $7 \times 7 \times 7$  neighborhood centered at voxel  $v$ . For sparse patch matching, the parameter  $\eta$  in Eq. (9.4), which con-

**FIGURE 9.10**

Five typical T2-weighted MR prostate images acquired from different scanners, showing large variations of both prostate appearance and shape, especially for the cases with or without using the endorectal coils.

**Table 9.1** Definitions of four evaluation measurements

Dice ratio	$\frac{2 \cdot V(S \cap F)}{V(S) + V(F)}$
Precision	$\frac{V(S \cap F)}{V(F)}$
Hausdorff distance	$\max(H(e_S, e_F), H(e_F, e_S)),$ $H(e_S, e_F) = \max \left\{ \min_{d_i \in e_S} \min_{d_j \in e_F} \text{dist}(d_i, d_j) \right\}$
Average surface distance	$\frac{1}{2} \left( \frac{\sum_{d_i \in e_S} \min_{d_j \in e_F} \text{dist}(d_i, d_j)}{\sum_{d_i \in e_S} 1} + \frac{\sum_{d_j \in e_F} \min_{d_i \in e_S} \text{dist}(d_j, d_i)}{\sum_{d_j \in e_F} 1} \right)$

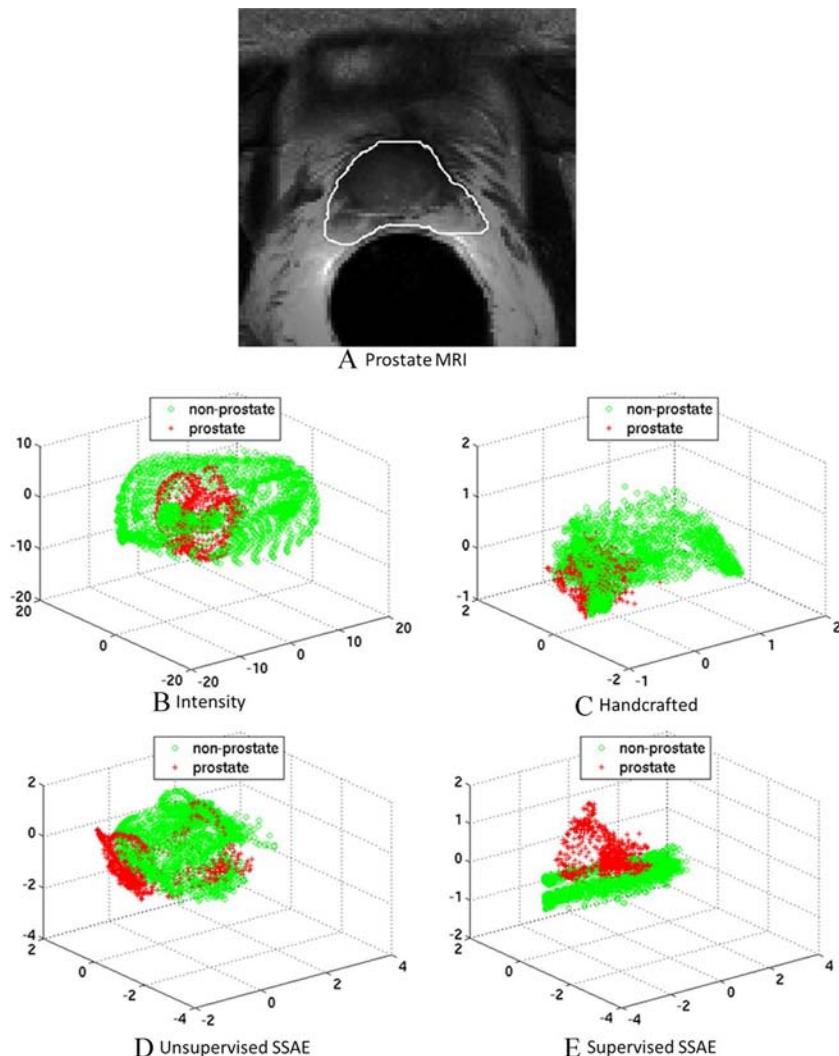
*S*, ground truth segmentation; *F*, automatic segmentation; *V*, volume size; *e<sub>S</sub>*, surfaces of ground-truth segmentation; *e<sub>F</sub>*, surface of automatic segmentation; *d<sub>i</sub>*, vertices on the surfaces *e<sub>S</sub>*; *d<sub>j</sub>*, vertices on the surfaces *e<sub>F</sub>*; *dist(d<sub>i</sub>, d<sub>j</sub>)*, Euclidean distance between vertices *d<sub>i</sub>* and *d<sub>j</sub>*.

trols the sparsity constraint, is 0.001. For the final deformable model segmentation, the parameter  $\lambda$  in Eq. (9.6), which weights the shape energy during deformation, is 0.5, and the parameter  $\mu$  in Eq. (9.5) is 0.5.

Given the ground-truth segmentation *S* and the automatic segmentation *F*, the segmentation performance is evaluated by four metrics: Dice ratio, precision, Hausdorff distance, and average surface distance. Dice ratio and precision measure the overlap between two segmentations. Hausdorff distance and average surface distance measure the boundary distance between two surfaces of segmentation. Detailed definitions are given as Table 9.1.

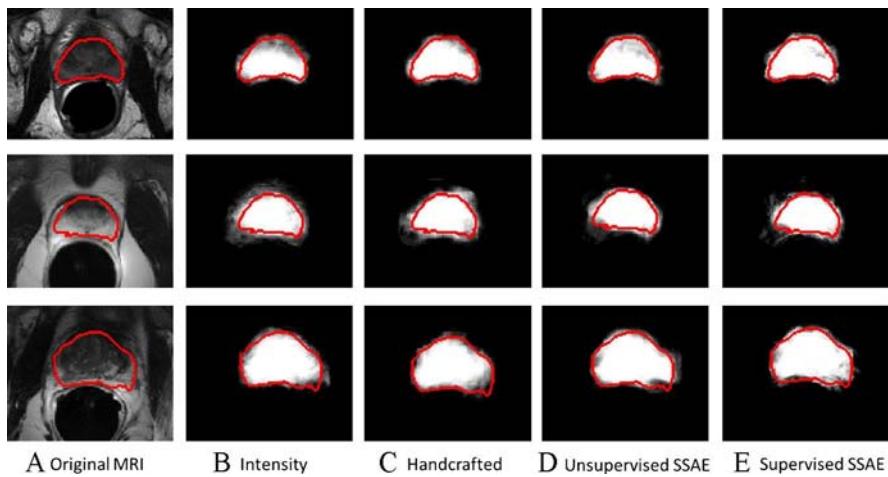
### 9.3.1 EVALUATION OF THE PERFORMANCE OF DEEP-LEARNED FEATURES

Inspired by [49], we plot the PCA-projected features to show the effectiveness of different features in separating voxels from different classes (e.g., the prostate class and the non-prostate class). After mapping each feature vector to the subspace spanned by the first three principal components, the effective features would (i) cluster the voxels with the same class label as close as possible and (ii) separate the voxels with different class labels as far as possible. First, we demonstrate the discrimination power of our deep learning features in Fig. 9.11, by visualizing the PCA-projected feature dis-

**FIGURE 9.11**

Distributions of voxel samples by using four types of features: (A) a typical slice of prostate MRI with white contour as ground truth, (B) intensity, (C) handcrafted, (D) unsupervised SSAE, and (E) supervised SSAE. Red crosses and green circles denote prostate and non-prostate voxel samples, respectively.

tributions of different feature representations of a typical prostate MRI (Fig. 9.11A), i.e., intensity patch (Fig. 9.11B), handcrafted (Fig. 9.11C), features learned by unsupervised SSAE (Fig. 9.11D), and features learned by supervised SSAE (Fig. 9.11E).

**FIGURE 9.12**

(A) Typical slices of T2 MR images with manual segmentations. The likelihood maps produced by sparse patch matching with four feature representations: (B) intensity patch, (C) handcrafted, (D) unsupervised SSAE, and (E) supervised SSAE. Red contours indicate the manual ground-truth segmentations.

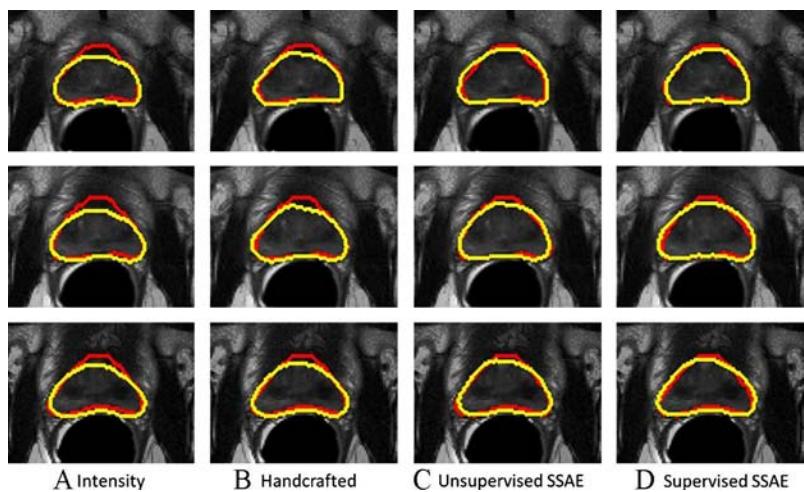
Specifically, for the case of handcrafted features, we include three commonly used features, i.e., Haar [50], HoG [47], and LBP [21]. The same patch size is used for computing all features under comparison. It can be seen that the deep-learned features from supervised SSAE have better clustering results in the projected feature space, and thus better discriminative power than other two predefined features (i.e., intensity and handcrafted), as well as deep-learned features by unsupervised SSAE. The superior performance of supervised SSAE over the unsupervised SSAE indicates the necessity of utilizing label information to improve the discrimination power of learned features.

Next, we evaluate the segmentation accuracy of different feature representations in the context of sparse patch matching. Table 9.2 lists the quantitative results (Dice ratio, precision, Hausdorff distance, and average surface distance) for all feature representations. The  $p$ -values (computed with paired t-test at 5% significance level), comparing the supervised SSAE with all other methods, are provided below each quantitative result. It can be observed that our supervised SSAE method significantly outperforms all the intensity and handcrafted feature methods. According to the paired t-test at 5% significance level, both our proposed methods (unsupervised and supervised SSAE) outperform the rest of competing method, but the supervised SSAE is not statistically superior to the unsupervised SSAE.

Fig. 9.12 further shows the typical likelihood maps estimated by four different feature representations for three different patients. It can be observed that the features learned from supervised SAE can better capture the prostate boundary, especially on

**Table 9.2** The mean and standard deviation of quantitative results for MR prostate segmentation with different feature representations. Best performance is indicated by bold font

<b>Method</b>	<b>Intensity</b>	<b>Haar</b>	<b>HOG</b>	<b>LBP</b>	<b>Handcraft</b>	<b>Unsupervised SSAE</b>	<b>Supervised SSAE</b>
Dice (%)	$85.3 \pm 6.2$ ( $1.1e-04$ )	$85.6 \pm 4.9$ ( $2.2e-05$ )	$85.7 \pm 4.9$ ( $6.1e-05$ )	$85.5 \pm 4.3$ ( $5.5e-05$ )	$85.9 \pm 4.5$ ( $7.0e-06$ )	$86.7 \pm 4.4$ ( $2.1e-01$ )	<b><math>87.1 \pm 4.2</math></b> (NA)
Precision (%)	$85.1 \pm 8.3$ ( $1.9e-03$ )	$85.9 \pm 8.5$ ( $3.63e-02$ )	$85.3 \pm 8.7$ ( $4.5e-03$ )	$83.7 \pm 7.7$ ( $1.3e-06$ )	$87.3 \pm 7.4$ ( $7.0e-01$ )	<b><math>87.3 \pm 7.3</math></b> ( $7.3e-01$ )	$87.1 \pm 7.3$ (NA)
Hausdorff	$8.68 \pm 4.24$ ( $1.8e-01$ )	$8.50 \pm 2.86$ ( $1.9e-01$ )	$8.51 \pm 2.69$ ( $1.6e-01$ )	$8.59 \pm 2.38$ ( $1.1e-01$ )	$8.55 \pm 2.91$ ( $1.5e-01$ )	$8.65 \pm 2.69$ ( $6.3e-02$ )	<b><math>8.12 \pm 2.89</math></b> (NA)
ASD	$1.87 \pm 0.93$ ( $6.0e-03$ )	$1.76 \pm 0.52$ ( $8.9e-03$ )	$1.74 \pm 0.50$ ( $3.1e-02$ )	$1.75 \pm 0.44$ ( $2.5e-02$ )	$1.77 \pm 0.54$ ( $5.0e-04$ )	$1.68 \pm 0.49$ ( $5.8e-01$ )	<b><math>1.66 \pm 0.49</math></b> (NA)

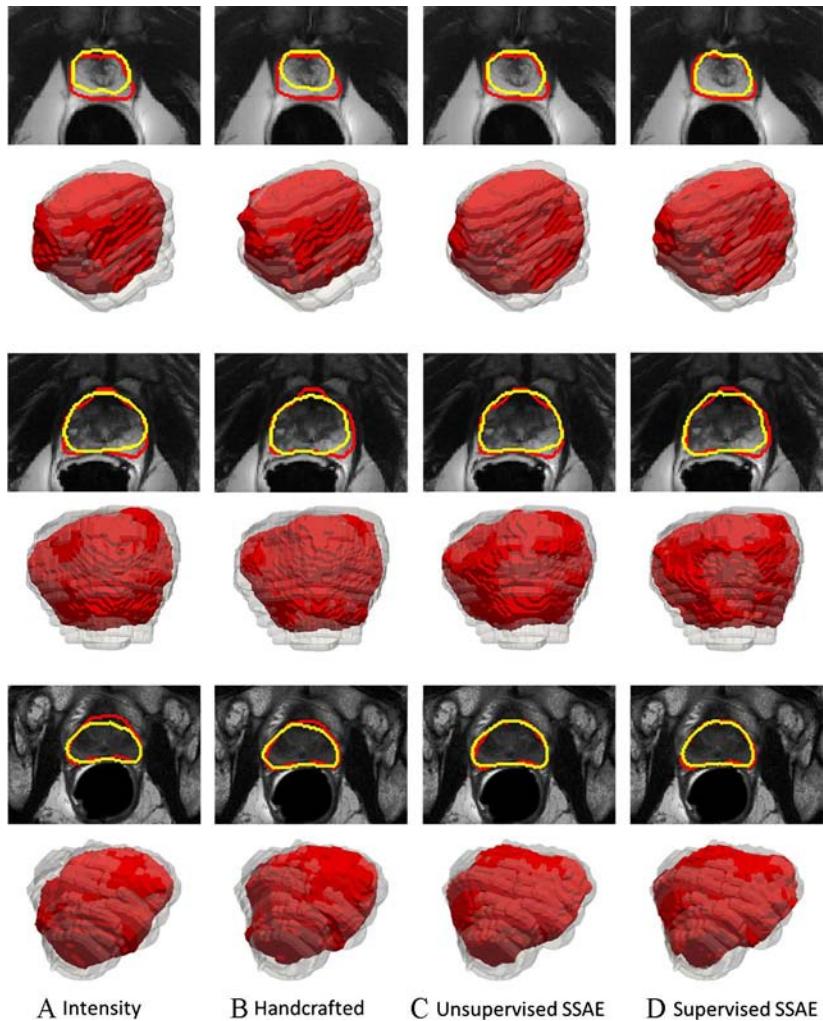
**FIGURE 9.13**

Typical prostate segmentation results of the same patients produced by four different feature representations: (A) intensity, (B) handcrafted, (C) unsupervised SSAE, and (D) supervised SSAE. Three rows show the results for three different slices of the same patient, respectively. Red contours indicate the manual ground-truth segmentations, and yellow contours indicate the automatic segmentations.

the anterior and right posterior parts of the prostate. Fig. 9.13 shows some typical segmentation results obtained by the sparse label matching method with four different feature representations, respectively. Similarly, the proposed method (i.e., supervised SSAE) achieves the best segmentation, especially on the anterior parts of the prostate, which demonstrates the effectiveness of our proposed method. Moreover, Fig. 9.14 gives the typical prostate segmentation results of different patients produced by four different feature representations, respectively. 3D visualization of the segmentation result has been provided below each segmentation result shown in 2D. For 3D visualization in each row, the red surface indicates automatic segmentation results with different features, such as intensity, handcrafted, unsupervised SSAE and supervised SSAE, respectively. The transparent gray surfaces indicate the ground-truth segmentations. Our proposed supervised SSAE method improves the segmentation accuracy on both the anterior and posterior parts of prostates.

### 9.3.2 EVALUATION OF THE PERFORMANCE OF DEFORMABLE MODEL

In this section, we further evaluate our deformable model to show its effectiveness. The comparison methods contain three different deformable model based methods. The first one is the conventional Active Shape Model (ASM). The second one uses intensity features for multi-atlas label fusion and then finalizes the segmentation by

**FIGURE 9.14**

Typical prostate segmentation results of three different patients produced by four different feature representations: (A) intensity, (B) Handcrafted, (C) unsupervised SSAE, and (D) supervised SSAE. The three odd rows show the results for three different patients, respectively. Red contours indicate the manual ground-truth segmentations, and yellow contours indicate the automatic segmentations. The three even rows show the 3D visualization of the segmentation results corresponding to the images above. For 3D visualizations in each row, the red surfaces indicate the automatic segmentation results using different features, such as intensity, handcrafted, unsupervised SSAE and supervised SSAE, respectively. The transparent gray surfaces indicate the ground-truth segmentations.

**Table 9.3** Mean and standard deviation of quantitative results for the segmentations obtained by supervised SSAE with/without using deformable model (DM). The best performance is indicated by bold font. \* denotes the statistically best performance among all the methods with/without deformable model (according to the paired t-test at 5% significant level)

<b>Method</b>	<b>ASM</b>		<b>Intensity</b>		<b>Handcrafted</b>		<b>Supervised SSAE</b>	
	<b>w/o DM</b>	<b>w/ DM</b>	<b>w/o DM</b>	<b>w/ DM</b>	<b>w/o DM</b>	<b>w/ DM*</b>		
Dice (%)	78.4 ± 9.7 (2.7e–11)	85.3 ± 6.2 (3.0e–06)	86.0 ± 4.3 (7.3e–10)	85.9 ± 4.5 (7.7e–08)	86.4 ± 4.4 (5.1e–06)	87.1 ± 4.2 (2.6e–03)	<b>87.8 ± 4.0*</b> (NA)	
Precision (%)	71.9 ± 13.8 (1.3e–21)	85.1 ± 8.3 (8.0e–17)	89.3 ± 7.4 (4.5e–07)	87.3 ± 7.4 (1.4e–11)	<b>92.3 ± 7.3</b> (7.7e–02)	87.1 ± 7.3 (5.7e–20)	91.6 ± 6.5 (NA)	
Hausdorff	11.50 ± 5.48 (8.5e–07)	8.68 ± 4.24 (3.6e–04)	7.72 ± 2.90 (2.5e–02)	8.55 ± 2.91 (3.8e–05)	7.97 ± 2.92 (1.2e–04)	8.12 ± 2.89 (7.4e–03)	<b>7.43 ± 2.82*</b> (NA)	
ASD	3.12 ± 1.71 (9.4e–10)	1.87 ± 0.93 (6.4e–04)	1.78 ± 0.55 (2.0e–07)	1.77 ± 0.54 (2.0e–05)	1.71 ± 0.50 (1.3e–03)	1.66 ± 0.49 (1.6e–02)	<b>1.59 ± 0.51*</b> (NA)	

adopting a deformable model on the produced likelihood map, similar to our proposed method. The second method follows the same procedure as the first one except using the handcrafted features, such as Haar, HOG, and LBP, instead of intensity patch for multi-atlas label fusion. Table 9.3 shows the segmentation results of intensity, handcrafted and supervised SSAE with/without deformable model and the  $p$ -value (with paired t-test at 5% significance level) between the supervised SSAE with deformable model and all other methods. According to the paired t-test at 5% significance level on Dice ratio, our proposed deformable model is statistically the best among all the competing methods. Specifically, our proposed supervised SAE outperforms the ASM, the intensity based deformable model, and the handcrafted based deformable model by DSC 10.7%, 2.1%, and 1.6%, respectively. Besides, it can be seen that, after adopting the second level of deformable segmentation, the segmentation accuracy can be further improved for all the comparing methods.

## 9.4 CONCLUSION

In this book chapter, we present an automatic segmentation algorithm for T2 MR prostate images. To address the challenges of making the feature representation robust to large appearance variations of the prostate, we propose to extract the deep-learned features by the SSAE framework. Then, the learned features are used under sparse patch matching framework to estimate the prostate likelihood map of the target image. To further relieve the impact of large shape variation in the prostate shape repository, a deformable model is driven toward the prostate boundary under the guidance from the estimated prostate likelihood map and sparse shape prior. The proposed method is extensively evaluated on the dataset containing 66 prostate MR images. By comparing with several state-of-the-art MR prostate segmentation methods, our method demonstrates superior performance in term of segmentation accuracy.

## REFERENCES

1. American Cancer Society, Prostate cancer, available from <http://www.cancer.org/acs/groups/cid/documents/webcontent/003134-pdf.pdf>.
2. K.M. Pondman, et al., MR-guided biopsy of the prostate: an overview of techniques and a systematic review, Eur. Urol. 54 (3) (2008) 517–527.
3. H. Hricak, et al., The role of preoperative endorectal magnetic resonance imaging in the decision regarding whether to preserve or resect neurovascular bundles during radical retropubic prostatectomy, Cancer 100 (12) (2004) 2655–2663.
4. S. Liao, et al., Automatic prostate MR image segmentation with sparse label propagation and domain-specific manifold regularization, in: J. Gee, et al. (Eds.), Information Processing in Medical Imaging, Springer, Berlin, Heidelberg, 2013, pp. 511–523.
5. P. Yan, et al., Label image constrained multiatlas selection, IEEE Trans. Cybern. 45 (6) (2015) 1158–1168.

6. Y. Cao, X. Li, P. Yan, Multi-atlas based image selection with label image constraint, in: 2012 11th International Conference on Machine Learning and Applications (ICMLA), 2012.
7. Y. Ou, et al., Multi-atlas segmentation of the prostate: a zooming process with robust registration and atlas selection, in: PROMISE12, 2012.
8. R. Toth, A. Madabhushi, Multifeature landmark-free active appearance models: application to prostate MRI segmentation, *IEEE Trans. Med. Imaging* 31 (8) (2012) 1638–1650.
9. R. Toth, et al., Accurate prostate volume estimation using multifeature active shape models on T2-weighted MRI, *Acad. Radiol.* 18 (6) (2011) 745–754.
10. M.R. Sabuncu, et al., A generative model for image segmentation based on label fusion, *IEEE Trans. Med. Imaging* 29 (10) (2010) 1714–1729.
11. Y. Jin, et al., Automatic clustering of white matter fibers in brain diffusion MRI with an application to genetics, *NeuroImage* 100 (2014) 75–90.
12. Y. Jin, et al., Automated multi-atlas labeling of the fornix and its integrity in Alzheimer’s disease, in: 2015 IEEE 12th International Symposium on Biomedical Imaging (ISBI), 2015.
13. Y. Jin, et al., Identification of infants at high-risk for autism spectrum disorder using multiparameter multiscale white matter connectivity networks, *Hum. Brain Mapp.* 36 (12) (2015) 4880–4896.
14. M. Yang, et al., Prostate segmentation in MR images using discriminant boundary features, *IEEE Trans. Biomed. Eng.* 60 (2) (2013) 479–488.
15. T.R. Langerak, et al., Label fusion in atlas-based segmentation using a selective and iterative method for performance level estimation (SIMPLE), *IEEE Trans. Med. Imaging* 29 (12) (2010) 2000–2008.
16. S.S. Chandra, et al., Patient specific prostate segmentation in 3-D magnetic resonance images, *IEEE Trans. Med. Imaging* 31 (10) (2012) 1955–1964.
17. K. Somkantha, N. Theera-Umpon, S. Auephanwiriyakul, Boundary detection in medical images using edge following algorithm based on intensity gradient and texture gradient features, *IEEE Trans. Biomed. Eng.* 58 (3) (2011) 567–573.
18. P. Viola, M. Jones, Rapid object detection using a boosted cascade of simple features, in: Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2001.
19. N. Dalal, B. Triggs, Histograms of oriented gradients for human detection, in: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005, 2005.
20. D. Lowe, Distinctive image features from scale-invariant keypoints, *Int. J. Comput. Vis.* 60 (2) (2004) 91–110.
21. T. Ojala, M. Pietikäinen, D. Harwood, A comparative study of texture measures with classification based on featured distributions, *Pattern Recognit.* 29 (1) (1996) 51–59.
22. S. Liao, et al., Sparse patch-based label propagation for accurate prostate localization in CT images, *IEEE Trans. Med. Imaging* 32 (2) (2013) 419–434.
23. Y. Bengio, Representation learning: a review and new perspectives, *IEEE Trans. Pattern Anal. Mach. Intell.* 35 (8) (2013) 1798–1828.
24. J. Mairal, et al., Supervised dictionary learning, in: Advances in Neural Information Processing Systems, 2009.
25. C. Farabet, et al., Learning hierarchical features for scene labeling, *IEEE Trans. Pattern Anal. Mach. Intell.* 35 (8) (2013) 1915–1929.

26. G. Carneiro, J.C. Nascimento, A. Freitas, The segmentation of the left ventricle of the heart from ultrasound data using deep learning architectures and derivative-based search methods, *IEEE Trans. Image Process.* 21 (3) (2012) 968–982.
27. P. Vincent, et al., Stacked denoising autoencoders: learning useful representations in a deep network with a local denoising criterion, *J. Mach. Learn. Res.* 11 (2010) 3371–3408.
28. C. Farabet, et al., Scene parsing with multiscale feature learning, purity trees, and optimal covers, in: *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, 2012.
29. S. Hoo-Chang, et al., Stacked autoencoders for unsupervised feature learning and multiple organ detection in a pilot study using 4D patient data, *IEEE Trans. Pattern Anal. Mach. Intell.* 35 (8) (2013) 1930–1943.
30. P. Vincent, et al., Extracting and composing robust features with denoising autoencoders, in: *Proceedings of the Twenty-Fifth International Conference on Machine Learning (ICML'08)*, ACM, 2008.
31. Y. Bengio, et al., Greedy layer-wise training of deep networks, in: *Advances in Neural Information Processing Systems*, vol. 19 (NIPS'06), MIT Press, 2006.
32. Y. Bengio, Deep learning of representations for unsupervised and transfer learning, in: *Workshop on Unsupervised and Transfer Learning*, 2012.
33. S.R. Bulò, P. Kotschieder, Neural decision forests for semantic image labelling, in: *IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, 2014.
34. H.-I. Suk, D. Shen, Deep learning-based feature representation for AD/MCI classification, in: K. Mori, et al. (Eds.), *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2013*, Springer, Berlin, Heidelberg, 2013, pp. 583–590.
35. H. Lee, et al., Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations, in: *Proceedings of the 26th Annual International Conference on Machine Learning*, ACM, Montreal, Quebec, Canada, 2009, pp. 609–616.
36. Y. Zhan, et al., Active scheduling of organ detection and segmentation in whole-body medical images, in: D. Metaxas, et al. (Eds.), *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2008*, Springer, Berlin, Heidelberg, 2008, pp. 313–321.
37. M. Modat, et al., Fast free-form deformation using graphics processing units, *Comput. Methods Programs Biomed.* 98 (3) (2010) 278–284.
38. M. Jorge Cardoso, et al., STEPS: similarity and truth estimation for propagated segmentations and its application to hippocampal segmentation and brain parcellation, *Med. Image Anal.* 17 (6) (2013) 671–684.
39. S.K. Warfield, K.H. Zou, W.M. Wells, Simultaneous truth and performance level estimation (STAPLE): an algorithm for the validation of image segmentation, *IEEE Trans. Med. Imaging* 23 (7) (2004) 903–921.
40. A.J. Asman, B.A. Landman, Non-local statistical label fusion for multi-atlas segmentation, *Med. Image Anal.* 17 (2) (2013) 194–208.
41. S. Nouranian, et al., An automatic multi-atlas segmentation of the prostate in transrectal ultrasound images using pairwise atlas shape similarity, in: K. Mori, et al. (Eds.), *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2013*, Springer, Berlin, Heidelberg, 2013, pp. 173–180.
42. T.F. Chan, L.A. Vese, Active contours without edges, *IEEE Trans. Image Process.* 10 (2) (2001) 266–277.
43. M. Kim, et al., Automatic hippocampus segmentation of 7.0 Tesla MR images by combining multiple atlases and auto-context models, *NeuroImage* 83 (2013) 335–345.

44. T.F. Cootes, et al., Active shape models – their training and application, *Comput. Vis. Image Underst.* 61 (1) (1995) 38–59.
45. S. Zhang, Y. Zhan, D.N. Metaxas, Deformable segmentation via sparse representation and dictionary learning, *Med. Image Anal.* 16 (7) (2012) 1385–1396.
46. J.G. Sled, A.P. Zijdenbos, A.C. Evans, A nonparametric method for automatic correction of intensity nonuniformity in MRI data, *IEEE Trans. Med. Imaging* 17 (1) (1998) 87–97.
47. Y. Gao, S. Liao, D. Shen, Prostate segmentation by sparse representation based classification, *Med. Phys.* 39 (10) (2012) 6372–6387.
48. <https://github.com/rasmusbergpalm/DeepLearnToolbox>.
49. G.E. Hinton, R.R. Salakhutdinov, Reducing the dimensionality of data with neural networks, *Science* 313 (5786) (2006) 504–507.
50. S.G. Mallat, A theory for multiresolution signal decomposition: the wavelet representation, *IEEE Trans. Pattern Anal. Mach. Intell.* 11 (7) (1989) 674–693.

# 10

## Characterization of Errors in Deep Learning-Based Brain MRI Segmentation

Akshay Pai\*,†, Yuan-Ching Teng†, Joseph Blair†, Michiel Kallenberg\*,  
Erik B. Dam\*, Stefan Sommer†, Christian Igel†, Mads Nielsen\*,†

Biomediq A/S, Copenhagen, Denmark\* University of Copenhagen, Copenhagen, Denmark†

### CHAPTER OUTLINE

10.1	Introduction .....	224
10.2	Deep Learning for Segmentation .....	225
10.3	Convolutional Neural Network Architecture .....	226
10.3.1	Basic CNN Architecture .....	226
10.3.2	Tri-Planar CNN for 3D Image Analysis .....	227
10.4	Experiments .....	228
10.4.1	Dataset .....	228
10.4.2	CNN Parameters .....	229
10.4.3	Training .....	230
10.4.4	Estimation of Centroid Distances .....	230
10.4.5	Registration-Based Segmentation .....	231
10.4.6	Characterization of Errors .....	231
10.5	Results .....	233
10.5.1	Overall Performance .....	233
10.5.2	Errors .....	233
10.6	Discussion .....	238
10.7	Conclusion .....	240
	References .....	240
	Notes .....	242

### CHAPTER POINTS

- Deep learning methods typically require large training datasets
- Geometric information of the anatomy can improve the performance of deep learning for medical image segmentation
- In our experiments, deep learning methods and registration-based methods produced complimentary types of errors. Thus, combining model-based and data-driven segmentation approaches is a promising future research direction

---

## 10.1 INTRODUCTION

Quantitative analysis of medical images often requires segmentation of anatomical structures. For instance, the volume of the hippocampus in the brain is associated with dementia of the Alzheimer's type [14]. In addition to volume quantification, segmentation aids in the analysis of regional statistics such as shape, structure, and texture. Segmentation also extends to detecting abnormal biological processes or even pathological anomalies, for instance microbleeds, and tumors. In this chapter, we will specifically deal with segmentation of normal brains from magnetic resonance images (MRI) using deep learning. The purpose is to evaluate and understand the characteristics of errors made by a deep learning approach as opposed to a model-based approach such as segmentation based on multi-atlas nonlinear registration. In contrast to the deep learning approach, registration-based methods rely heavily on topological assumptions about the objects in the image, i.e., that the anatomical structures are similar enough to be mapped onto each other.

Segmentation essentially involves dividing images into meaningful regions, which can be viewed as a voxel classification task. The most simplistic approach is to manually segment brains. However, this is a time-consuming process and has significant interoperable variations. Automating delineation provides a systematic way of obtaining regions of interest in a brain on the fly as soon as a medical image is acquired. The field of brain segmentation is dominated by multi-atlas based methods. They are driven by a combination of spatial normalization through registration followed by a classification step, which can be simple majority voting or a more sophisticated method such as Bayesian weighting.

With the advance in computational capabilities and refinement of algorithms, deep learning based methods have seen an increased usage [27,17]. They have proven to be extremely efficient, stable, and state-of-the-art in many computer vision applications. The architecture of a deep neural network is loosely inspired by the human cortex [16]. It consists of several layers, each performing a nonlinear transformation. In contrast to most traditional machine learning approaches where features typically need to be handcrafted to suit the problem at hand, deep learning methods aim at automatically learning features from raw or only slightly pre-processed input. The idea is that each layer adds a higher level of abstraction – a more abstract feature representation – on top of the previous one.

This chapter will focus on the application of deep neural networks, specifically convolutional neural networks (CNNs), to brain MRI segmentation. After a short introduction, we will evaluate the accuracy of the CNNs compared to a standard multi-atlas registration based method. We reproduce results from [7] on the MICCAI 2012 challenge dataset. We will focus on the characterization of the errors made by CNNs in comparison with the registration-based method. Finally, we will discuss some directions for future work such as ensemble learning of multiple classifiers including CNNs.

---

## 10.2 DEEP LEARNING FOR SEGMENTATION

Most segmentation methods either rely solely on intensity information (e.g., appearance models) or combine intensity and higher order structural information of objects. A popular example of the latter is multi-atlas registration, which is based on a specific set of templates that, in a medical application, typically contains labelings of anatomical structures. An image registration algorithm then maps these templates to a specific subject space through a nonlinear transformation. Voting schemes are then applied to choose the right label from a given set of labels obtained from the transformed templates [6].

In a typical model-based approach, images are first registered to a common anatomical space and then specific features are extracted from the aligned images. The features are often higher order statistics of a region such as histograms of gradients or curvatures. Learning algorithms such as support vector machines (SVMs) or neural networks are then used to classify regions in a medical image. Fischl et al. [10] have integrated both learning and model based methods for segmentation. Images are normalized to a common template, and then a Markov random field learns anatomically corresponding positions and intensity patterns for different anatomical structures. In the above-mentioned algorithms, domain knowledge about spatial locations of various anatomical structures is incorporated as a Bayesian prior distribution of the segmentation algorithm.

In contrast, data-driven deep learning approaches aim at learning a multi-level feature representation from image intensities, typically not using anatomical background knowledge about spatial relations. The feature representation at each level results from a nonlinear transformation of the representations at the previous level. This is in contrast to shallow architectures, which have at most one distinguished layer of nonlinear processing (i.e., one level of data abstraction). Popular shallow architectures include linear methods such as logistic regression and linear discriminant analysis as well as kernel methods such as SVMs and Gaussian processes. Kernel classifiers use kernel functions to map input from a space where linear separation is difficult to a space where the separation is easier. The class of kernel functions, and therefore the feature representation, has to be chosen *a priori*. Because deep learning methods learn the feature representation, it has been argued that they have the advantage of being less dependent on prior knowledge of the domain. However, feature learning requires that enough training data is available to learn the representations. Deep neural networks are, in general, highly complex nonlinear models having many free parameters. Sufficient training data is needed to learn these parameters and to constrain the model such that it generalizes well to unseen data. Therefore, deep learning methods excel in applications where there is an abundance of training data, which is typically not the case in medical imaging where data is often scarce.

Accordingly, so far there are only rather few applications of deep learning methods in medical imaging, specifically brain segmentation. A prominent example of CNNs for classification applied to 2D medical images is mitosis detection – winning the MICCAI 2013 Grand Challenge on Mitosis Detection [5,33,32]. Some more

recent notable applications of deep learning in the field of medical image analysis are collected in [12]. Highlights include classification of brain lesions [4,23], microbleeds in brain [8], histopathological images [29], colonic polyp [31], lung nodules [28], and mammograms [15].

Limitations of deep learning applications in medical image analysis are often attributed to computational complexity and large variance in the data. In order to address these, methodological contributions have been made to improve both the robustness, and computational complexity of CNN. For instance, the tri-planar approach was proposed to overcome the computational complexity of the 3D CNNs [24], and later Roth et al. [25] proposed random rotations of the tri-planar views as a way to obtain representations of the 3D data. Brosch et al. [4] use a combination of convolutional and deconvolutional layers to learn different feature scales.

---

## 10.3 CONVOLUTIONAL NEURAL NETWORK ARCHITECTURE

In the following, we describe the convolutional neural network (CNN) architecture we considered in this study, which is depicted in Fig. 10.1. An in-depth introduction to neural networks and CNNs is beyond the scope of this chapter, we refer to [11].

### 10.3.1 BASIC CNN ARCHITECTURE

The basic principles of CNNs have been introduced by LeCun et al. in the late 1990s [18]. Deep neural networks consist of several layers. Different types of layers can be distinguished in a CNN. First, we have convolutional layers. This type of layer computes a convolution of its input with a linear filter (which has been suggested for modeling neural responses in the visual cortex a long time ago, e.g., [34]). The coefficients of each convolution kernel are learned in the same way as weights in a neural network. Typically, after the convolution a nonlinear (activation) function is applied to each element of the convolution result. Thus, a convolutional layer can be viewed as a layer in a standard neural network in which the neurons share weights. The output of the layer is referred to as a *feature map*. The input to the convolutional layer can be the input image or the output of a preceding layer, that is, another feature map. The particular type of weight sharing in a convolutional layer does not only reduce the degrees of freedom, it ensures *equivariance to translation* [11], which means that, ignoring boundary effects, if we translate the input image, the resulting feature map is translated in the same way.

Often convolutional layers are followed by a pooling layer. Pooling layers compute the maximum or average over a region of a feature map. This results in a subsampling of the input. Pooling reduces the dimensionality, increases the scale, and also supports translation invariance in the sense that a slight translation of the input does not (or only marginally) change the output.

Convolutional layers and pooling layers take the spatial structure of their input into account, which is the main reason for their good performance in computer vision

and image analysis. In contrast, when standard non-convolutional neural networks are applied to raw (or only slightly preprocessed) images, the images are vectorized. The resulting vectors serve as the input to the neural network. In this process, information about the spatial relation between pixels or voxels is discarded and there is neither inherent equi-variance to translation nor translation invariance, which are important properties to achieve generalization in object recognition tasks.

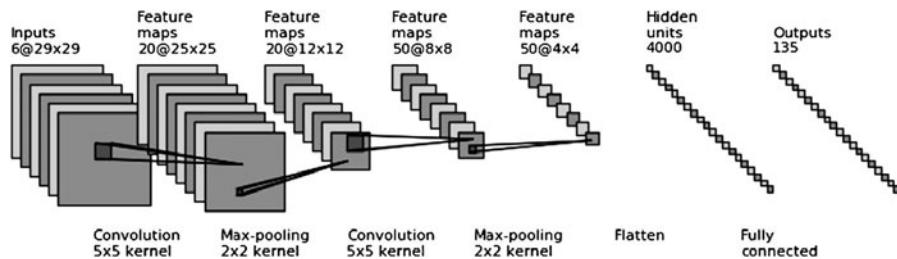
The final layers in a CNN correspond to a standard multi-layer perceptron neural network [2], which receives the vectorized (or *flattened*) feature maps of the preceding layer as input. There can be zero, one, or more hidden layers, which compute an affine linear combination of their inputs and apply a nonlinear activation function to it. The final layer is the output layer. For multi-class classification, it typically computes the softmax function, turning the input into a probability distribution over the classes. For learning, we can then consider the cross-entropy error function, which corresponds to minimizing the negative logarithmic likelihood of the training data [2].

### 10.3.2 TRI-PLANAR CNN FOR 3D IMAGE ANALYSIS

Processing 3D input images, such as brain scans, with CNNs is computationally demanding even if parallel hardware is utilized. To classify (e.g., segment) a voxel with a 3D CNN, an image patch around the voxel is extracted. This patch, which is a small 3D image, serves as the input to the CNN, which computes 3D feature maps and applies 3D convolution kernels. To reduce the computational complexity and the degrees of freedom of the model, *tri-planar* CNNs can be used instead of a 3D CNN. A tri-planar CNN for 3D image analysis considers three two-dimensional image patches, which are processed independently in the convolutional and pooling layers; all feature maps and convolutions are two-dimensional. For each voxel, one  $n \times n$  patch is extracted in each image plane (sagittal, coronal, and transverse), centered around the voxel to be classified.

This tri-planar approach [24,26] reduces the computational complexity compared to using three-dimensional patches, while still capturing spatial information in all three planes. Due to the reduction in computation time, it is often possible to consider larger patches, which incorporate more distant information.

The tri-planar architecture considered in this study is sketched in Fig. 10.1. It consists of two convolutional layers (including applying nonlinear activation functions) each followed by a pooling layer. The latter compute the maximum over the pooling regions. Then the feature maps are *flattened* and fed into a hidden layer with 4000 neurons. As activation functions we use the unbounded rectifier linear unit (ReLU)  $a \mapsto \max(0, a)$ , which has become the standard activation function in deep CNNs. The absolute values of the derivatives of standard sigmoid activation functions are always smaller than one, which leads to the *vanishing gradient* effect in deep networks (e.g., see [1]), which is addressed by ReLUs. The number of neurons in the softmax output layer corresponds to the 135 different classes we consider in our segmentation.

**FIGURE 10.1**

Convolutional neural network with 3 input channels.  $n@k \times k$  indicates  $n$  feature maps with size  $k \times k$ .

The efficient tri-planar architecture allows us to consider input patches at two different scales, one for local and one for more global information. Accordingly, we have in total six two-dimensional inputs to our CNN architecture (2D patches from three planes at two scales).

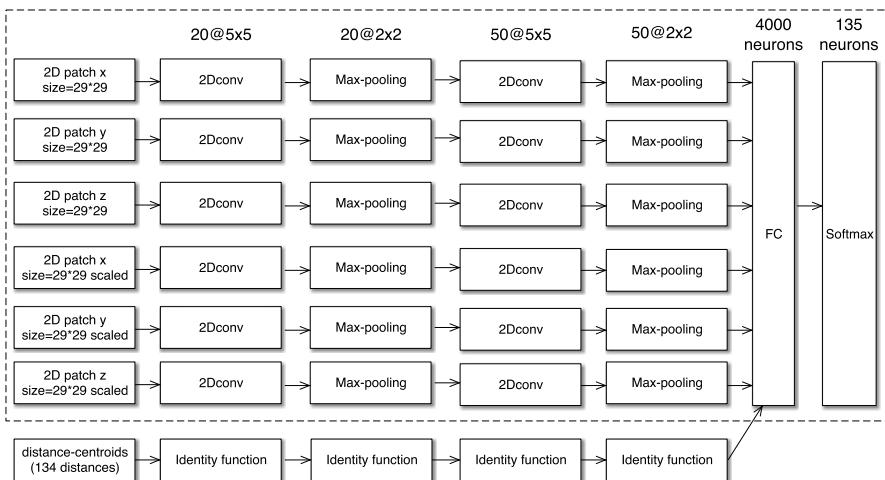
## 10.4 EXPERIMENTS

In this section, we will use three different feature representations for CNNs for segmenting structures in a normal brain using T1 weighted magnetic resonance images (MRI). To evaluate characteristics of the errors made by CNN, we compare the segmentations obtained to those obtained using a common multi-atlas registration-based method.

We will first evaluate the performance of CNN-based segmentation method (with and without centroid distances) in comparison with the registration-based segmentation. After that, we will characterize the errors from both methods to understand the limitations of CNNs in segmenting brain MRIs with limited training data. For a fair evaluation of the methods, we exclude the outliers i.e., images where the registration-based method fail.

### 10.4.1 DATASET

All the experiments were performed on a dataset that was released for the MICCAI 2012 multi-atlas challenge.<sup>1</sup> The dataset contains 15 atlases in the training set, and 20 atlases in the testing set. All 35 images are T1-weighted structural MRIs selected from the OASIS project [19]. The cortical regions were labeled using the BrainCOLOR protocol; the non-cortical regions were labeled using the NeuroMorphometric protocol. These images were acquired in the sagittal plane and the voxel size is  $1.0 \times 1.0 \times 1.0 \text{ mm}^3$ . Only the labels appearing in all images are used. The labels consist of 40 cortical and 94 non-cortical regions.

**FIGURE 10.2**

The convolutional neural network used in the experiment. The dotted box represents the architecture of CNN without centroids and the solid box (outermost) represents the architecture of the CNN with centroids as an input feature. There are 135 neurons in the softmax layer (instead of 134) since the background class is also included.

### 10.4.2 CNN PARAMETERS

All voxels in the brain were considered during the training phase. Due to similar slice thickness relative to the in-plane voxel size, we use information from both 2D and tri-planar planes for segmenting the volumetric images. The first three rows of the input layers are a set of tri-planar patches. The next three rows are also tri-planar, however, with a different patch area. First, a patch with a larger (than the first three rows) size is chosen, and then the patch is downsampled to match the dimensions of the patches from the first three rows.

A CNN with multiple convolution layers was used. A schematic representation of the CNN architecture can be found in Fig. 10.2. In the first convolutional layer, we trained 20 kernels. For patches of size  $29 \times 29$ , we used a kernel size of  $5 \times 5$ . Max-pooling with kernels of  $2 \times 2$  was performed on the convolved images. The output images of the max-pooling layer served as input to the next layer, where 50 kernels of size  $5 \times 5$  were used. In addition, max-pooling with kernels of size  $2 \times 2$  was performed on the convolved images. The output of the second layer of max-pooling served as input to a fully-connected layer with 4000 nodes. The output layer of the CNN computed the soft-max activation function. This function maps its inputs to positive values summing to 1, so that the outputs can be interpreted as the posterior probabilities for the classes. The class with the highest probability is finally chosen for prediction.

### 10.4.3 TRAINING

In order not to overfit the training set when choosing the network architecture, we randomly chose 10 images as a training set and the remaining 5 images as a validation set. We split the training dataset into  $n = 4$  parts (mini-batches) so that each part is small enough to fit GPU memory while avoiding too much of I/O. We trained for 60 epochs (one epoch corresponds to training on  $n$  mini-batches). Before training, we randomly sampled 400,000 voxels evenly from the training set, 200,000 voxels evenly from the validation set for each epoch. We then extract patch features from those voxels as inputs to the network.

The validation set was used to detect overfitting. No overfitting was observed in the initial experiment and the validation error did not decrease after 10 epochs. Thus, the data from the training and validation set were again combined and the network was retrained on all 15 images for 10 epochs.

After the network was trained, images in the test set were classified. We evaluate the performance using Dice coefficients between true and the CNN-classified labels across all the 134 anatomical regions:

$$\text{Dice}(A, B) = 2 \times \frac{|A \cap B|}{|A| + |B|}, \quad (10.1)$$

where  $A$  and  $B$  are the true labels and predicted labels, respectively. A Dice score of 1 indicates perfect match.

The loss function used in this chapter can be found in [3]. The optimization was performed using stochastic gradient descent with a momentum term [3].

The code we used in this chapter is based on Theano,<sup>2</sup> a Python-based library that enables symbolic expressions to be evaluated on GPUs [30].

### 10.4.4 ESTIMATION OF CENTROID DISTANCES

Unlike computer vision problems where segmentation tasks are more or less unstructured, brain regions are consistent in their spatial position. To incorporate this information, Brebisson et al. [3] used relative centroid distances as an input feature. Relative centroid distance is the Euclidean distance between each voxel and the centroid of each class.

In the training phase, the true labels were used to generate the centroids. In the testing phase, the trained network without the centroids (see the dotted box in Fig. 10.2), was used to generate an initial segmentation from which centroid distances were extracted. This layer may be replaced by any segmentation method or a layer that provides relevant geometric information (e.g., Freesurfer [10]). We tried replacing the initial segmentations used to generate centroid distances by a registration-based segmentation. However, this did not yield any improvement in the Dice scores. This indicates that including geometric information must be more intricate than just incorporating spatial positions. In the experiments, we therefore used CNNs to generate initial segmentations and the corresponding centroid distances during testing. The final CNN network is illustrated in Fig. 10.2.

**Table 10.1** Confusion matrix used to compute error metrics such as Dice and Jaccard. TP, true positive; FP, false positive; TN, true negative; and FN, false negative

Predicted/True	Segment	Non-segment
Segment	TP	FP
Non-segment	FN	TN

#### 10.4.5 REGISTRATION-BASED SEGMENTATION

In the MICCAI challenge, the top performing methods relied significantly on nonlinear image registration. Hence, we chose to compare to a registration-based segmentation method. Since the purpose of the chapter is to only evaluate the types of error made by a registration-based method, we stuck to a simple majority voting scheme for classification.

We first linearly registered the images using an inverse consistent affine transformation with 12 degrees of freedom and mutual information as a similarity measure. After that, we performed an inverse consistent diffeomorphic nonlinear registration using the kernel bundle framework and stationary velocity fields [22]. The parameters of the registration were the same as in [22]. We used a simple majority voting to fuse the labels. More sophisticated voting schemes were avoided since they can be viewed as an ensemble layer which optimizes classifications obtained from registration.

#### 10.4.6 CHARACTERIZATION OF ERRORS

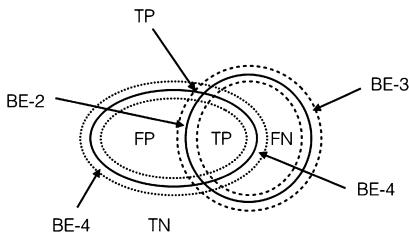
In order to assess performances of both the CNN and the registration-based method in depth, we characterize the errors made by the two methods. Typically, segmentation errors are computed via either Dice or Jaccard indices, which are computed from the confusion matrix, see [Table 10.1](#):

$$\text{DiceScore} = \frac{2\text{TP}}{|\text{TP} + \text{FN}| + |\text{TP} + \text{FP}|} = \frac{2\text{TP}}{2\text{TP} + \text{FN} + \text{FP}} \\ \sim \frac{2\text{TP} + 1}{2\text{TP} + \text{FN} + \text{FP} + 1}, \quad (10.2)$$

$$\text{Jaccard} = \frac{\text{TP}}{\text{TP} + \text{FN} + \text{FP}} \sim \frac{\text{TP} + 1}{\text{TP} + \text{FN} + \text{FP} + 1}. \quad (10.3)$$

For instance, the Dice is a size of the overlap of the two segmentations divided by the mean size of the two objects. This can be expressed in terms of the entries of [Table 10.1](#). Similarly, measures such as sensitivity, specificity may also be expressed in terms of the table entries:

$$\text{sensitivity} = \frac{\text{TP}}{\text{TP} + \text{FN}} \sim \frac{\text{TP} + 1}{\text{TP} + \text{FN} + 1}, \quad (10.4)$$

**FIGURE 10.3**

Augmentation of the region by width of  $\epsilon$ . Defining an object's boundary from  $\epsilon$ -erosion and  $\epsilon$ -dilation,  $\frac{A \cup S_\epsilon}{A \setminus S_\epsilon}$ , where  $S_\epsilon$  is a structural element defined as a sphere of radius  $\epsilon$ .

**Table 10.2** Updated confusion matrix including boundaries indecisions. BE, boundary errors; and TB, true boundary. With an additional definition of boundary errors, one can come up with new measures that characterize different types of errors made by the segmentation method

Predicted/True	Segment	$\epsilon$ region	Non-segment
Segment	TP	BE-1	FP
$\epsilon$ region	BE-2	TB	BE-3
Non-segment	FN	BE-4	TN

$$\text{specificity} = \frac{\text{TN}}{\text{TN} + \text{FP}} \sim \frac{\text{TN} + 1}{\text{TN} + \text{FP} + 1}. \quad (10.5)$$

Errors in segmentations may have several characteristics. Sometimes, boundaries of regions are a source of noise. Other errors arise when lumps of regions are given the wrong-label. Let us consider  $\epsilon$ -boundary regions defined by  $\epsilon$ -erosion and  $\epsilon$ -dilation as shown in Fig. 10.3. When we exclude the  $\epsilon$ -boundary when assessing the segmentation quality, see Table 10.2 and Fig. 10.3, we get new quality measures:

$$\text{Dice}_\epsilon = \frac{2\text{TP}_\epsilon + 1}{2\text{TP}_\epsilon + \text{FN}_\epsilon + \text{FP}_\epsilon + 1}, \quad (10.6)$$

$$\text{Jaccard}_\epsilon = \frac{\text{TP}_\epsilon + 1}{\text{TP}_\epsilon + \text{FN}_\epsilon + \text{FP}_\epsilon + 1}, \quad (10.7)$$

$$\text{sensitivity}_\epsilon = \frac{\text{TP}_\epsilon + 1}{\text{TP}_\epsilon + \text{FN}_\epsilon + 1}. \quad (10.8)$$

Typically, the boundary errors can be identified by moving the boundaries of the segmentations through morphological operations. Examples illustrating these errors will be given in the results section. In a boundary error, since the core (excluding the

indecisive boundaries) of the segmentations are correctly classified, the Dice scores are expected to be high. However, in labeling error, in addition to the boundary errors, there may be lumps of regions with wrong labels which will negatively affect the Dice score. In order to formalize these effects, we define a measure called the core-Dice score (Eq. (10.9)):

$$\text{cDice}(A, B) = \frac{1 + 2 \times |\text{core}(A) \cap \text{core}(B)|}{1 + |\text{core}(A) \cap \text{non-boundary}(B)| + |\text{core}(B) \cap \text{non-boundary}(A)|} \quad (10.9)$$

where  $\text{core}$  is the inner segmentation after excluding  $\epsilon$  boundaries,  $\text{non-boundary}(B) = B \setminus \text{boundary}(B)$ , and  $\text{non-boundary}(A) = A \setminus \text{boundary}(A)$ . The constant 1 added to both the numerator and denominator is to avoid the divide-by-zero error when the width of boundary is so large that all the regions are ignored. The value converges to 1 when the core regions are perfectly matched or totally isolated. We can, therefore, estimate where the errors come from by calculating the core dice score with increasing width of the boundary. Typically, core-Dice measure will tend to increase as a function of the boundary width (see Fig. 10.3) in contrast to segmentations with labeling errors.

## 10.5 RESULTS

This section will present results of the overall performance of the CNN algorithm in comparison to the registration-based algorithm. The results were divided into two categories: (a) overall performance of the method in comparison to the registration-based method and (b) evaluation of the types of errors made by the CNN and registration-based methods.

### 10.5.1 OVERALL PERFORMANCE

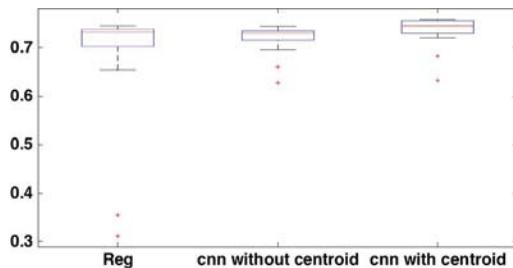
Table 10.3 lists the mean Dice scores obtained by the CNN and registration-based methods across the 134 regions and all the images. The CNN with spatial information, in the form of centroid, gave better Dice scores compared to CNN without centroids. The registration-based method still achieved a Dice score close to that of CNN, indicating the importance of information about the underlying topology of the anatomies. Fig. 10.4 details the Dice differences. We observed that the standard deviation in dice error decreased with the inclusion of centroid distances as a feature.

### 10.5.2 ERRORS

In order to dig into the performance analysis of CNN-based segmentation system, we look at the kind of segmentation errors the method made. We broadly classify the errors into two classes as specified in Section 10.4.6.

**Table 10.3** The mean Dice score of the test set using different methods. Registration is the multi-atlas registration method; CNN without centroids is the CNN with 2 sets of tri-planar patches; CNN with centroids is the CNN with 2 sets of tri-planar patches and the distances of centroids

Method	Mean Dice score
Registration	0.724
CNN without centroids	0.720
CNN with centroids	0.736



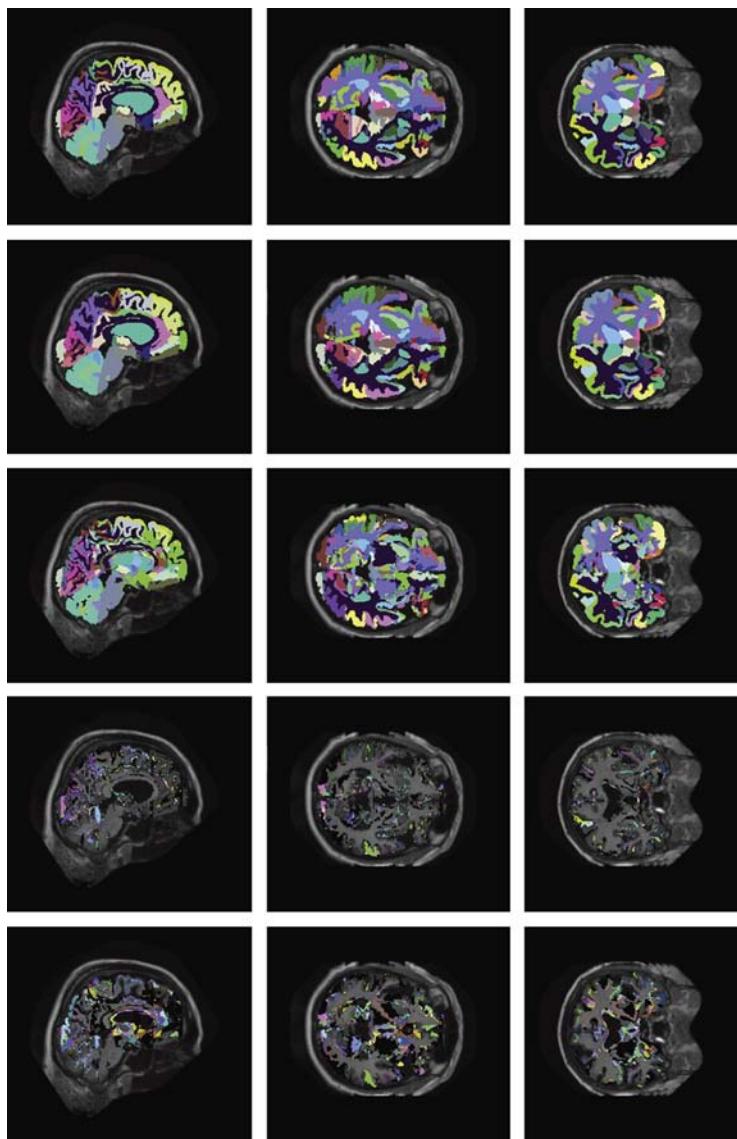
**FIGURE 10.4**

The Dice score of different methods. This box plot illustrates the variance in the Dice measure for each method.

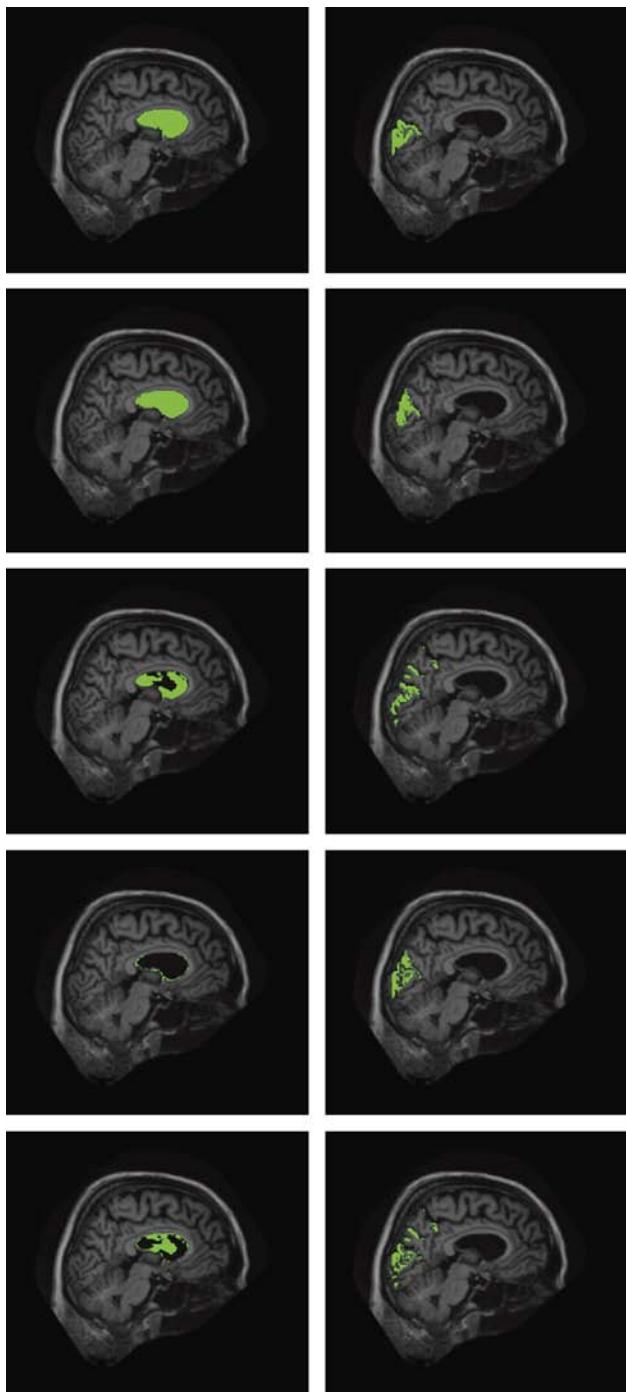
[Fig. 10.5](#) illustrates the segmentation of a test case (number 1119) using both CNN and registration. We can see that overall the CNN led to more lumps of misclassifications compared to registration. The misclassifications of registration-based segmentations were generally on the boundaries of the object.

[Fig. 10.6](#) illustrates an example of both labeling and boundary errors. The first column is a segmentation of the left lateral ventricle and the second column represents the segmentation of the left cuneus. The first row illustrates the ground truth, the second row illustrates segmentation by the registration-based method, the third row illustrates segmentation by the CNN-based method, the fourth row represents the difference between ground truth and registration, and finally the last row represents difference between ground truth and CNN segmentation. As illustrated in the first column, CNN-based segmentation has misclassification of similar looking voxels for instance misclassification of left lateral ventricles around insular cortex, see [Fig. 10.6](#). In contrast, the errors from registration-based method is on the boundary of the ventricle.

The second column illustrates more severe boundary errors made by both CNN and registration in segmenting the left cuneus. While the segmentation is very spe-

**FIGURE 10.5**

Segmentation results from both CNN and registration. The first row illustrates the true labels. Different colors indicate different types of labels. The second and third rows are predictions from registration and CNN, respectively. The last two rows are differences of each method (registration and CNN) from true labels. Difference illustrates that CNN has more misclassifications compared to CNN. The CNN results are based on CNN with centroids.

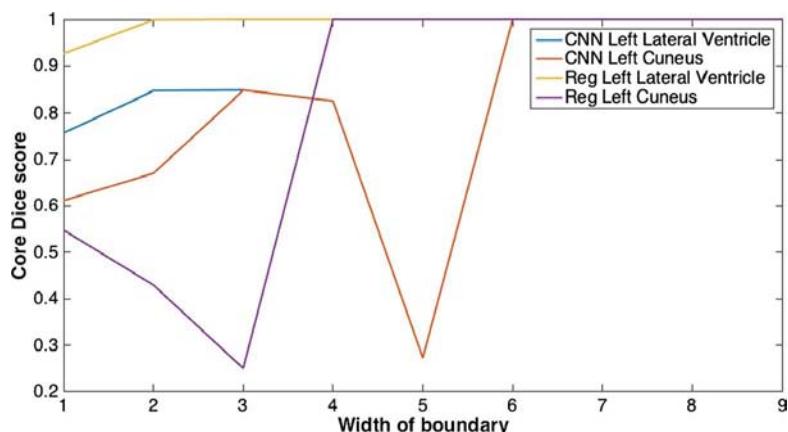


**FIGURE 10.6**

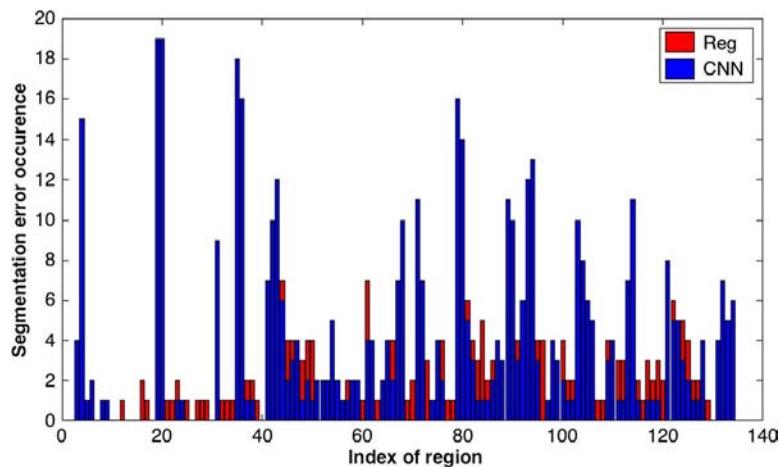
Illustration of segmentation errors for a single class. The left column shows segmentation of the left ventricle. CNN (last row, left image) shows clear misclassification (labeling error) of regions, in comparison the errors by registration (fourth row, left image) are strictly boundary based. The right column illustrates segmentation of the left cuneus, where both methods make more severe boundary errors. The second row represents segmentation using registration. Third row represents segmentation using CNN. Fourth row represents difference of registration-based segmentation and the true segmentation, and finally the fifth row represents difference of CNN-based segmentation and the true segmentation.

cific, the sensitivity is low. In such regions, combining both methods may not improve the Dice score.

A graph to further illustrate the difference between the different segmentation errors is shown in Fig. 10.7. Segmentation of left lateral ventricle using CNN has labeling errors since the core dice drops on change of boundary width whereas the registration-based segmentation has 1 core dice upon the change of boundaries. In segmenting left cuneus, both registration and CNN make labeling errors. Labeling errors are errors with lumps of misclassified voxels that are not specific to the anatomy, i.e., false positives. Typically, with such errors the Dice coefficient changes negatively with a change in the boundary width. In contrast, a change in the boundary width makes a positive change in the Dice score when the errors are of the boundary type. This is the logic we use to classify the errors. A positive slope in the graph is classified as a boundary error and a negative slope is classified as a labeling error.

**FIGURE 10.7**

The core-Dice score with different boundary width with respect to two classes. The x-axis indicates width of boundary in pixels, while the y-axis shows core-Dice score.

**FIGURE 10.8**

The histogram of labeling error occurrence in 134 classes across all the test images. The X axis shows index of class, while the Y axis gives labeling error occurrence. The results are from CNN with centroid. The CNN results are based on CNN with centroids.

Fig. 10.8 shows the histograms of the occurrence of labeling errors for each class across the test data. As expected, the CNN made more labeling errors on average compared to the registration-based method. This is expected since the training dataset is small, and the registration-based methods have stronger topological constraints. On the contrary, the registration-based method made more boundary errors than labeling errors.

## 10.6 DISCUSSION

This chapter presented a characterization of errors made by a CNN for automatic segmentation of brain MRI images. Fairly accurate segmentations were obtained. However, the performance was inferior to both a standard registration plus majority voting-based method and other model-based methods presented at the MICCAI 2012 challenge.<sup>3</sup> Note that the best performing methods in the challenge rely on nonlinear image registration. With this as motivation, we compared the results of segmentation using a CNN to a registration-based segmentation methodology.

The performance of both the registration and the CNN-based methods could possibly be improved, which might lead to different results in the comparison. For instance, Moeskops et al. [21] use convolution kernels at different scales. This results in similar mean Dice scores (0.73). Studies like [3] include 3D patches in the feature stack. Including 3D patches is not trivial because of intensive memory requirements

depending on the size of the patches involved. In order to avoid memory intensive representations, the tri-planar approach [24] was proposed.

As illustrated in various examples in the results section, our CNN made more labeling errors than segmentation or boundary errors. This can be caused by limited training size, lack of spatial information, or a combination of both. Unlike images arising in many computer vision problems, medical images are structured, that is, there is a spatial consistency in the location of objects. This gives a particular advantage to methods that use spatial information. The gains from the use of spatial information is apparent when the centroid distances are included as a feature. The centroid distances can be obtained from any popular segmentation algorithm. In such a case, centroid distances in the training must also be computed (as opposed to using the true labels) using the segmentation method so that the network can learn the errors made by the method.

Lack of training data is in many cases a limiting factor for the application of deep learning in medical image analysis. For instance, the specific application presented in this chapter aims at obtaining segmentations with 134 classes from just 15 training images. Lowering the number of classes with the same dataset yields significantly better Dice scores [21]. When large training sets are available, CNN-based methods generally give better results [13]. In computer vision problems such as the PASCAL challenge [9], CNN-methods perform particularly well due to the presence of large and relatively cheap ground truth data. In the area of medical imaging analysis, however, obtaining ground truth is expensive. In cases where there is an upper limit to the availability of both data and corresponding annotations (what we use as ground truth), using deep learning as a regression tool to a clinical outcome may be more useful. An example application of using CNNs in regression may be found in [20].

In our experiments, the registration-based method made errors complementary to that of the CNN-based method. Thus, combining the two approaches may improve Dice scores. The choice of where to include the geometric information in the network is a research topic in itself. For example, one may add information on the top via centroid distances or have a more intricate way of adding geometric information as a priori inside the network at some level. In addition, one may even consider combining classifications obtained by both methods via an ensemble learning to improve performances for large category classification (large number of classes). With limited validation data, unsupervised/semi-supervised learning as an initialization [15] in conjunction with model-driven approaches such as image registration may also improve results.

Even though we only chose one architecture in this chapter, the choices are many, for examples see the special issue [12]. The choices may, for example, vary in the number of layers or in a fusion of multiple architectures. An increase of the network depth proved effective in the recent ILSVRC challenge where the authors use as many as 152 hidden layers [13]. Once the architecture is fixed, small variations in the settings may not necessarily yield significantly better results [21].

---

## 10.7 CONCLUSION

Deep learning methods have already improved performances in medical tasks such as classification of tumors (3/4 classes), detection of microbleeds, or even classification of specific structures in histological images. However, CNN-based methods for segmentation of brain MRI images with a large number of classes may need more training data than used in our study to outperform model-driven approaches. With the advent of big data, connecting healthcare centers may increase the availability of medical data significantly. However, it is likely that obtaining hand annotations for such data may be intractable. To leverage the use of data, the field of deep learning in medical imaging may have to move toward combining the benefits of model-driven techniques, unsupervised learning, and semi-supervised learning. In this chapter, we have shown that model-driven and data-driven approaches can make complementary errors. This encourages using techniques, for example, ensemble learning, to combine the two approaches.

---

## REFERENCES

1. Y. Bengio, P. Simard, P. Frasconi, Learning long-term dependencies with gradient descent is difficult, *IEEE Trans. Neural Netw.* 5 (2) (1994) 157–166.
2. C.M. Bishop, *Pattern Recognition and Machine Learning*, Springer-Verlag, 2006.
3. A. Brébiisson, G. Montana, Deep neural networks for anatomical brain segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2015, pp. 20–28.
4. T. Brosch, L. Tang, Y. Yoo, D. Li, A. Traboulsee, R. Tam, Deep 3D convolutional encoder networks with shortcuts for multiscale feature integration applied to multiple sclerosis lesion segmentation, *IEEE Trans. Med. Imaging* 35 (6) (2016).
5. D.C. Ciresan, A. Giusti, L.M. Gambardella, J. Schmidhuber, Mitosis detection in breast cancer histology images using deep neural networks, in: *Medical Image Computing and Computer-Assisted Intervention (MICCAI 2013)*, Springer, 2013, pp. 411–418.
6. P. Coupé, J. Manjón, V. Fonov, J. Pruessner, M. Robles, D. Collins, Patch-based segmentation using expert priors: application to hippocampus and ventricle segmentation, *NeuroImage* 54 (2) (2011) 940–954.
7. A. de Brébiisson, G. Montana, Deep neural networks for anatomical brain segmentation, in: *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW 2015)*, 2015, pp. 20–28.
8. Q. Dou, H. Chen, L. Yu, L. Zhao, J. Qin, D. Wang, V. Mok, L. Shi, P.A. Heng, Automatic detection of cerebral microbleeds from MRI images via 3D convolutional neural networks, *IEEE Trans. Med. Imaging* 35 (6) (2016).
9. M. Everingham, L. Van Gool, C.K.I. Williams, J. Winn, A. Zisserman, The PASCAL Visual Object Classes Challenge 2012 (VOC2012) results, <http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html>.
10. B. Fischl, D. Salat, E. Busa, M. Albert, M. Dieterich, C. Haselgrove, A. van der Kouwe, R. Killiany, D. Kennedy, S. Klaveness, A. Montillo, N. Makris, B. Rosen, A. Dale, Whole

- brain segmentation: automated labeling of neuroanatomical structures in the human brain, *Neuron* 33 (2002) 341–355.
- 11. I. Goodfellow, Y. Bengio, A. Courville, *Deep Learning*, MIT Press, 2016, in press.
  - 12. H. Greenspan, B. van Ginneken, R. Summers, in: Special Issue on Deep Learning in Medical Imaging, *IEEE Trans. Med. Imaging* 35 (4) (2016).
  - 13. K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2016)*, 2016.
  - 14. C.J. Jack, R. Petersen, Y. Xu, P. O'Brien, G. Smith, R. Ivnik, B. Boeve, E. Tangalos, E. Kokmen, Rates of hippocampal atrophy correlate with change in clinical status in aging and AD, *Neurology* 55 (4) (2000) 484–489.
  - 15. M. Kallenberg, K. Petersen, M. Nielsen, A. Ng, P. Diao, C. Igel, C. Vachon, K. Holland, R.R. Winkel, N. Karssemeijer, M. Lillholm, Unsupervised deep learning applied to breast density segmentation and mammographic risk scoring, *IEEE Trans. Med. Imaging* 35 (6) (2016).
  - 16. N. Krüger, P. Janssen, S. Kalkan, M. Lappe, A. Leonardis, J. Piater, A.J. Rodriguez-Sánchez, L. Wiskott, Deep hierarchies in the primate visual cortex: what can we learn for computer vision?, *IEEE Trans. Pattern Anal. Mach. Intell.* 35 (8) (2013) 1847–1871.
  - 17. Y. LeCun, Y. Bengio, G. Hinton, Deep learning, *Nature* 521 (2015) 436–444.
  - 18. Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, *Proc. IEEE* 86 (11) (1998) 2278–2324.
  - 19. D.S. Marcus, T.H. Wang, J. Parker, J.G. Csernansky, J.C. Morris, R.L. Buckner, Open access series of imaging studies (OASIS): cross-sectional MRI data in young, middle aged, nondemented, and demented older adults, *J. Cogn. Neurosci.* 19 (9) (2007) 1498–1507.
  - 20. S. Miao, W.J. Zhang, R. Liao, A CNN regression approach for real-time 2D/3D registration, *IEEE Trans. Med. Imaging* 35 (2016) 1352–1363.
  - 21. P. Moeskops, M.A. Viergever, A.M. Mendrik, L.S. de Vries, M.J.N.L. Benders, I. Išgum, Automatic segmentation of MR brain images with a convolutional neural network, *IEEE Trans. Med. Imaging* 35 (5) (2016).
  - 22. A. Pai, S. Sommer, L. Sørensen, S. Darkner, J. Sporring, M. Nielsen, Kernel bundle diffeomorphic image registration using stationary velocity fields and Wendland basis functions, *IEEE Trans. Med. Imaging* 35 (6) (2016).
  - 23. S. Pereira, A. Pinto, V. Alves, C.A. Silva, Brain tumor segmentation using convolutional neural network in MRI images, *IEEE Trans. Med. Imaging* 35 (6) (2016).
  - 24. A. Prasoon, K. Petersen, C. Igel, F. Lauze, E. Dam, M. Nielsen, Deep feature learning for knee cartilage segmentation using a triplanar convolutional neural network, in: *Medical Image Computing and Computer Assisted Intervention (MICCAI)*, in: *Lect. Notes Comput. Sci.*, vol. 8150, Springer, 2013, pp. 246–253.
  - 25. H.R. Roth, L. Lu, J. Liu, J. Yao, K.M. Cherry, L. Kim, R.M. Summers, Improving computer-aided detection using convolutional neural networks and random view aggregation, *IEEE Trans. Med. Imaging* 35 (6) (2016).
  - 26. H.R. Roth, L. Lu, A. Seff, K.M. Cherry, J. Hoffman, S. Wang, J. Liu, E. Turkbey, R.M. Summers, A new 2.5 D representation for lymph node detection using random sets of deep convolutional neural network observations, in: *Medical Image Computing and Computer Assisted Intervention (MICCAI)*, in: *Lect. Notes Comput. Sci.*, Springer, 2014, pp. 520–527.
  - 27. J. Schmidhuber, Deep learning in neural networks: an overview, *Neural Netw.* 61 (2015) 85–117.

28. A.A. Setio, F. Ciompi, G. Litjens, P. Gerke, C. Jacobs, S. van Riel, M. Winkler Wille, M. Naqibullah, C. Sanchez, B. van Ginneken, Pulmonary nodule detection in CT images: false positive reduction using multi-view convolutional neural networks, *IEEE Trans. Med. Imaging* 35 (6) (2016).
29. K. Sirinukunwattana, S. Raza, Y. Tsang, D. Snead, I. Cree, N. Rajpoot, Locality sensitive deep learning for detection and classification of nuclei in routine colon cancer histology images, *IEEE Trans. Med. Imaging* 35 (6) (2016).
30. Theano Development Team, Theano: a Python framework for fast computation of mathematical expressions, arXiv:1605.02688, 2016.
31. M.J.J.P. van Grinsven, B. van Ginneken, C.B. Hoyng, T. Theelen, C.I. Sánchez, Fast convolutional neural network training using selective data sampling: application to hemorrhage detection in color fundus images, *IEEE Trans. Med. Imaging* 35 (6) (2016).
32. M. Veta, P.J. van Diest, S.M. Willems, H. Wang, A. Madabhushi, A. Cruz-Roa, F.A. González, A.B.L. Larsen, J.S. Vestergaard, A.B. Dahl, D.C. Ciresan, J. Schmidhuber, A. Giusti, L.M. Gambardella, F.B. Tek, T. Walter, C. Wang, S. Kondo, B.J. Matuszewski, F. Precioso, V. Snell, J. Kittler, T.E. de Campos, A.M. Khan, N.M. Rajpoot, E. Arkoumani, M.M. Lacle, M.A. Viergever, J.P.W. Pluim, Assessment of algorithms for mitosis detection in breast cancer histopathology images, *Med. Image Anal.* 20 (1) (2015) 237–248.
33. M. Veta, M. Viergever, J. Pluim, N. Stathonikos, P.J. van Diest, Grand challenge on mitosis detection, in: Medical Image Computing and Computer-Assisted Intervention (MICCAI 2013), 2013.
34. W. von Seelen, Informationsverarbeitung in homogenen Netzen von Neuronenmodellen, *Kybernetik* 5 (4) (1968) 133–148.

---

## NOTES

1. [https://masi.vuse.vanderbilt.edu/workshop2012/index.php/Main\\_Page](https://masi.vuse.vanderbilt.edu/workshop2012/index.php/Main_Page).
2. <http://deeplearning.net/software/theano/>.
3. <https://masi.vuse.vanderbilt.edu/workshop2012>.

PART

# Medical Image Registration

4

This page intentionally left blank

# Scalable High Performance Image Registration Framework by Unsupervised Deep Feature Representations Learning

# 11

Shaoyu Wang<sup>\*,†</sup>, Minjeong Kim<sup>\*</sup>, Guorong Wu<sup>\*</sup>, Dinggang Shen<sup>\*</sup>

University of North Carolina at Chapel Hill, Chapel Hill, NC, United States<sup>\*</sup> Donghua University, Shanghai, China<sup>†</sup>

## CHAPTER OUTLINE

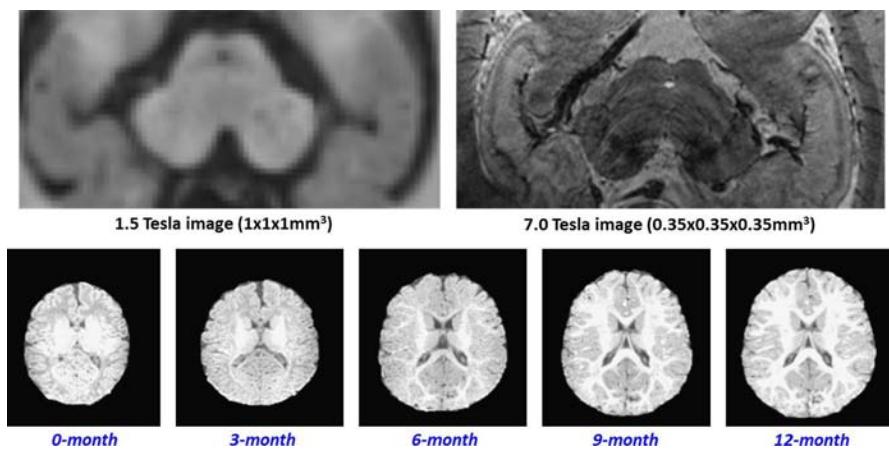
11.1	Introduction .....	246
11.2	Proposed Method .....	248
11.2.1	Related Research .....	248
11.2.1.1	Linear vs. Nonlinear Model .....	248
11.2.1.2	Shallow vs. Deep Model .....	250
11.2.2	Learn Intrinsic Feature Representations by Unsupervised Deep Learning .....	250
11.2.2.1	Auto-Encoder .....	250
11.2.2.2	Stacked Auto-Encoder .....	252
11.2.2.3	Convolutional SAE Network (CSAE) .....	253
11.2.2.4	Patch Sampling .....	255
11.2.3	Registration Using Learned Feature Representations .....	255
11.2.3.1	Advantages of Feature Representations Learned by Deep Learning Network .....	255
11.2.3.2	Learning-Based Image Registration Framework .....	256
11.3	Experiments .....	258
11.3.1	Experimental Result on ADNI Dataset .....	259
11.3.2	Experimental Result on LONI Dataset .....	260
11.3.3	Experimental Result on 7.0-T MR Image Dataset .....	263
11.4	Conclusion .....	265
	References .....	265

## 11.1 INTRODUCTION

Deformable image registration is defined as the process of establishing anatomical correspondences between two medical images, which is a fundamental task in medical image processing [5–12]. To name a few among its most important applications, image registration is widely used in (a) population studies toward delineating the group difference [14,15], (b) longitudinal projects for measuring the evolution of structure changes over time [1–4,16], and (c) multi-modality fusion to facilitate the diagnosis [17,18].

**Challenges of developing the state-of-the-art image registration method.** In general, there are two main challenges.

1. *Lack of discriminative feature representation to determine anatomical correspondences.* Image intensity is widely used to reveal the anatomical correspondences [19–24], however, two image patches that show similar, or even the same, distribution of intensity values do not guarantee that the two points correspond from an anatomical point of view [25–28]. Handcrafted features, such as geometric moment invariants [11] or Gabor filters [29], are also widely used by many state-of-the-art image registration methods [24,26–28,30]. In general, the major pitfall of using those handcrafted features is that the developed model tends to be ad-hoc. In other words, the model is only intended to recognize image patches specific to an image modality or a certain imaging application [31]. Supervised learning-based methods have been proposed to select the best set of features from a large feature pool that may include plenty of redundant handcrafted features [31–33]. However, for this approach, the ground-truth data with known correspondences across the set of training images is required. In general, image registration methods that use supervised learning for feature selection are biased and aggravate the risk of over-fitting the learning process by either unintentionally removing many relevant features, or selecting irrelevant features due to lack of ground truth.
2. *Lack of fast development framework of image registration method for specific medical image application.* Typically, it takes months, or even years, to develop a new image registration method that has acceptable performance for a new imaging modality or a new imaging application. As shown in the top of Figs. 11.1, 7.0 T images have more texture patterns than 1.5 T images since the high signal-to-noise ratio in image acquisition [34]. Thus, it is not as straightforward to use feature representations developed for 1.5 T image in the image registration for 7.0 T images. Another example is infant image registration. Although tons of image registration methods have been developed for adult brain images, early brain development studies still lack efficient image registration methods since image appearance of infant MR brain images change dramatically from 2-week-old to 2-year-old, due to myelination [35]. Rapid development of scalable high performance image registration methods that are applicable for new modalities and new applications becomes urgent in neuroscience and clinical investigations.



**FIGURE 11.1**

Challenging brain image registration cases (7.0-T MR images and infant brain images) that are in great need of distinctive feature representations.

The bottleneck, in the above challenges, is due to the insufficient number of training samples with ground truth. To overcome this difficulty, unsupervised learning-based feature selection methods are worthwhile to investigate. Because of the complexity of the data, the conventional unsupervised approaches that use simple linear models, such as PCA and ICA [36,37], are typically not suitable because they are unable to preserve the highly nonlinear relationships, when projected to low-dimensional space. More advanced methods, such as ISOMAP [38], kernel SVM [39,40], locally linear embedding (LLE) [41,42], and sparse coding [43], can learn the nonlinear embedding directly from the observed data within a single layer of projection. However, since the learned features are found using only a single layer, or a *shallow model*, the selected features may lack high-level perception knowledge (e.g., shape and context information), and may not be suitable for correspondence detection. To model complex systems with nonlinear relationships and learn high-level representations of features, unsupervised deep learning has been successfully applied to feature selection in many nontrivial computer vision problems, where deep models significantly outperform shallow models [13,40,44–52].

In this book chapter, we introduce how to learn the hierarchical feature representations directly from the observed medical images by using unsupervised deep learning paradigm. Specifically, we describe a stacked auto-encoder (SAE) [13,45, 47,51] with convolutional network architecture [50,53] into our unsupervised learning framework. The inputs to train the convolutional SAE are 3D image patches. Generally speaking, our learning-based framework consists of two components, i.e., the encoder and decoder networks. On the one hand, the multi-layer encoder network is used to transfer the high-dimensional 3D image patches into the low-dimensional

feature representations, where a single auto-encoder is the building block to learn nonlinear and high-order correlations between two feature representation layers. On the other hand, the decoder network is used to recover 3D image patches from the learned low-dimensional feature representations by acting as feedback to refine inferences in the encoder network. Since the size of 3D image patches can be as large as  $\sim 10^4$ , it is computationally expensive to directly use an SAE to learn useful features in each layer. To overcome this problem, we use a convolutional network [50] to efficiently learn the translational invariant feature representations [50,54] such that the learned features are shared among all image points in a certain region. Finally, we present a general image registration framework with high accuracy by allowing the learned feature representations to steer the correspondence detection between two images.

To assess the performance of the proposed registration framework, we evaluate its registration performance on the conventional 1.5 T MR brain images (i.e., the elderly brains from ADNI dataset and the young brains from LONI dataset [55]), which show that the proposed registration framework achieves much better performance than exiting state-of-the-art registration methods with handcrafted features. We also demonstrate the scalability of the proposed registration framework on 7.0 T MR brain images, where we develop an image registration method for the new modality with satisfactory registration results.

The remaining sections are organized as follows: In Section 11.2, we first present the deep learning approach for extracting hierarchical feature representations and the new learning-based registration framework based on the deep learning feature. In Section 11.3, we evaluate our proposed registration framework by comparing with the conventional registration methods, and we provide a brief conclusion in Section 11.4.

---

## 11.2 PROPOSED METHOD

### 11.2.1 RELATED RESEARCH

Unsupervised learning is an important research topic in machine learning. For example, principle component analysis (PCA) [56] and sparse representation [57] can be used to learn intrinsic feature representations in computer vision and medical image applications. In the following, we will brief on several typical unsupervised learning methods and investigate deep learning as the reliable feature representations in medical imaging scenario.

#### 11.2.1.1 Linear vs. Nonlinear Model

K-means [58] and Gaussian mixture model (GMM) [59] are two well-known clustering methods based upon linear learning models. In particular, given a set of training data  $X_{L \times M}$ , where  $L$  is the dimension of the data and  $M$  is the number of samples, the clustering methods learn  $K$  centroids such that each sample can be assigned to the closest centroid. Suppose the observed feature vectors (such as image patches and SIFT [60]) form a feature space and the appropriate  $K$  centroids in

the high-dimensional feature space are known. A typical pipeline defines a function  $f : \mathcal{R}^L \rightarrow \mathcal{R}^K$  that maps the observed  $L$ -dimensional feature vector to a  $K$ -dimensional feature vector ( $K < L$ ) [61]. For instance, we first calculate the affiliations for each observed feature vector (w.r.t. the  $K$  centroids) and then use such affiliations as morphological signatures to represent each key point in the feature space. However, the limitation of K-means and GMM is that the number of centroids is required to be larger as the input dimension grows. Thus, these clustering-based methods may not be applicable in learning the intrinsic representations for high-dimensional medical images.

PCA [56] is one of the most commonly used methods for dimension reduction. PCA extracts a set of basis vectors from the observed data, which maximizes the data variance of the projected subspace (spanned by the basis vectors). These basis vectors are obtained by calculating the eigenvectors of the covariance matrix of the input data. Given the observed data  $X = [x_1, \dots, x_m, \dots, x_M]$ , the following steps are sequentially applied in the training stage: (i) calculate the mean by  $\hat{x} = \frac{1}{M} \sum_{m=1}^M x_m$ ; (ii) compute the eigenvectors  $E = [e_j]_{j=1,\dots,L}$  for the covariance matrix  $\frac{1}{M-1} \overline{X} \overline{X}^T$ , where  $\overline{X} = [x_m - \hat{x}]_{m=1,\dots,M}$  and  $E$  are sorted in the decreasing order of eigenvalues; (iii) determine the first  $Q$  largest eigenvalues such that  $\sum_{j=1}^Q (\lambda_j)^2 > f_Q \sum_{j=1}^L (\lambda_j)^2$ , where  $f_Q$  defines the proportion of the remaining energy. In the end, each training data  $x_m$  can be approximately reconstructed as  $x_m = \hat{x} + E_Q b$ , where  $E_Q$  contains the first  $Q$  largest eigenvectors of  $E$  and  $b = E_Q^T (x_m - \hat{x})$ . In the testing stage, given the new testing data  $x_{new}$ , its low-dimensional feature representation can be obtained by  $b_{new} = E_Q^T (x_{new} - \hat{x})$ . This classic approach for finding low-dimensional representations has achieved many successes in medical image analysis area [62,63]. However, PCA is only an orthogonal linear transform and is not optimal for finding structures with highly non-Gaussian distributions. As shown in the experiment, such feature representations learned by PCA can hardly assist image registration to establish more accurate correspondences in feature matching.

Since there are huge variations in anatomical structures across individuals, the above linear models might have limitations in finding intrinsic feature representations [40,44,64]. A more flexible representation can be obtained by learning nonlinear embedding of features. To name a few of them, ISOMAP [38], local linear embedding [41,42], and sparse dictionary learning [49,64,65] have been extensively investigated in several medical imaging applications. In many of these applications, such nonlinear methods show much more power than their simpler linear counterpart versions [34,66–68]. However, as pointed next, the depth of inferring model is another important factor in learning optimal feature representations. Unfortunately, those learning models can only infer low-level feature representations within a single layer. Thus, the learned features may lack high-level perception knowledge, such as shape and contextual information.

### 11.2.1.2 Shallow vs. Deep Model

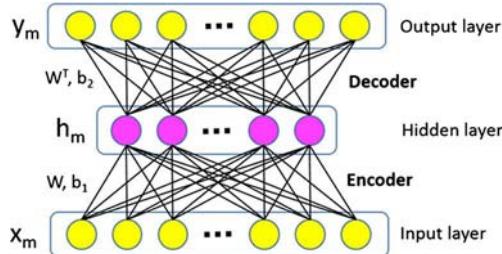
As stated above, the morphological patterns in various medical images are very complex. To describe complex anatomical structure at each image point, it is necessary to use multi-layer representations to hierarchically capture the image information from different perspectives. Therefore, many references encourage using multi-resolution frameworks, since it is simple and efficient to obtain hierarchical feature representations. For instance, multi-resolution histogram is used in [69] as the morphological signature to drive the correspondence detection in brain MR image registration. Most of these feature learning methods (including PCA, decision trees, and SVMs) use shallow models, including neural networks with only one hidden layer. Theoretical results show that the internal feature representations learned by these methods are simple and incapable of extracting intrinsic types of complex structure from medical images. Training these shallow models also requires large amounts of labeled data. Thus, the development of a learning model, which has deep architecture that can discover more abstract feature representations, is of crucial importance.

In studies on visual cortex, the brain is found to have a deep architecture consisting of several layers. In visual activities, signals flow from one brain layer to the next to obtain different levels of abstraction. By simulating the function of the deep architecture of the brain, Hinton et al. [13] introduced a deep learning model, which was composed of several layers of nonlinear transformations to stack multiple layers on top of each other to discover more abstract feature representations in higher layers. Compared with shallow models that learn feature representations in a single layer, deep learning can encode multi-level information from simple to complex. To this end, deep learning is more powerful for learning hierarchical feature representations directly from high-dimensional medical images. Among various deep learning models, we propose an intrinsic feature representation mechanism based on convolutional stacked auto-encoder [13,48,50,51,54] to learn the features from 3D image patches, as described below.

## 11.2.2 LEARN INTRINSIC FEATURE REPRESENTATIONS BY UNSUPERVISED DEEP LEARNING

### 11.2.2.1 Auto-Encoder

As one typical neural network, three sequential layers structurally define the auto-encoder (AE): the input layer, the hidden layer, and the output layer (as shown in Fig. 11.2). In existing neural network algorithms, the labeled data were required as training data that are essential to the backpropagation-based fine-tuning pass, as those labels were used to readjust the parameters. However, AE, developed by Hinton and Rumelhart [70], performs backpropagation by setting the output equal to the input, and thus is trained to minimize the discrepancy between the actual output and the expected output, where the expected output is the same as the input. Hinton et al. defined the AE as a nonlinear generalization of PCA, which uses an adaptive, multilayer “encoder” network to transform the high-dimensional input data into a low-dimensional code, while a similar “decoder” network is used to recover the data from the code

**FIGURE 11.2**

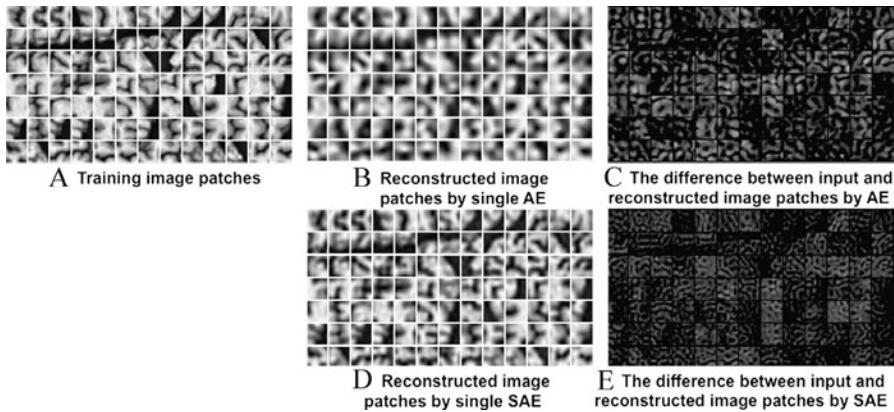
The architecture of an AE.

[13]. As a result, AE is able to learn compressed feature representations of the input data, without supervision by constraining the number of hidden nodes. Furthermore, sparsity is a useful constraint when the number of hidden nodes is larger than the number of input values that can discover the intrinsic features of the data. As a result of the sparse constraint, few hidden nodes can be active most of the time, and even with many hidden nodes, the network is able to learn important features of the data to reconstruct it at output layer.

Here, the goal of AE is to learn the latent feature representations from the 3D image patches collected from images. Let  $D$  and  $L$  respectively denote the dimensions of hidden representations and input patches. Given an input image patch  $x_m \in \mathcal{R}^L$  ( $m = 1, \dots, M$ ), AE maps an activation vector  $h_m = [h_m(j)]_{j=1, \dots, D}^T \in \mathcal{R}^D$  by  $h_m = f(Wx_m + b_1)$ , where the weight matrix  $W \in \mathcal{R}^{D \times L}$  and the bias vector  $b_1 \in \mathcal{R}^D$  are encoder parameters. Here,  $f$  is the logistic sigmoid function  $f(a) = 1/(1 + \exp(-a))$ . It is worth noting that  $h_m$  is considered as the representation vector of the particular input training patch  $x_m$  via AE. Next, the representation  $h_m$  from the hidden layer is decoded to a vector  $y_m \in \mathcal{R}^L$ , which approximately reconstructs the input image patch vector  $x$  by another deterministic mapping  $y_m = f(W^T h_m + b_2) \approx x_m$ , where the bias vector  $b_2 \in \mathcal{R}^L$  contains the decoder parameters. Therefore, the energy function in AE can be formulated as

$$\{W, b_1, b_2\} = \arg \min_{W, b_1, b_2} \sum_{m=1}^M \|f(W^T(f(Wx_m + b_1))) + b_2 - x_m\|_2^2. \quad (11.1)$$

The sparsity constraint upon the hidden nodes in the network usually leads to more interpretable features. Specifically, we regard each hidden node  $h_m(j)$  as being “active,” if the degree of  $h_m(j)$  is close to 1, or “inactive,” if the degree is close to 0. Thus, the sparsity constraint requires most of the hidden nodes to remain “inactive” for each training patch  $x_m$ . The Kullback–Leibler divergence is used to impose the sparsity constraint to each hidden node by enforcing the average activation degree over the whole training data, i.e.,  $\hat{\rho}_j = \sum_{m=1}^M h_m(j)$ , to be close to a very small

**FIGURE 11.3**

The reconstructed image patches by single auto-encoder (B) and stacked auto-encoder (D).

value  $\rho$  ( $\rho$  is set to 0.001 in the experiments):

$$KL(\rho \parallel \hat{\rho}_j) = \rho \log \frac{\rho}{\hat{\rho}_j} + (1 - \rho) \log \frac{1 - \rho}{1 - \hat{\rho}_j}. \quad (11.2)$$

Then, the overall energy function of AE with sparsity constraint is defined as

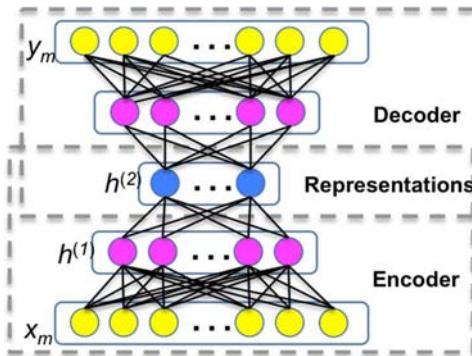
$$\begin{aligned} \{W, b_1, b_2\} = \operatorname{argmin}_{W, b_1, b_2} & \sum_{m=1}^M \|f(W^T(f(Wx_m + b_1))) + b_2 - x_m\|_2^2 \\ & + \beta \sum_{j=1}^D KL(\rho \parallel \hat{\rho}_j), \end{aligned} \quad (11.3)$$

where  $\beta$  controls the strength of sparsity penalty term. Typical gradient based back-propagation algorithm can be used for training single AE [44,45].

#### 11.2.2.2 Stacked Auto-Encoder

A single AE is limited in what it can present, since it is a shallow learning model. As shown in Fig. 11.3A, a set of training image patches are sampled from brain MR images, each sized at  $15 \times 15$  (for demonstration, we use 2D patches as examples). We set the number of hidden nodes to 100 in this single AE. The reconstructed image patches are shown in Fig. 11.3B. It is obvious that many details have been lost after reconstruction from low-dimensional representations, as displayed in Fig. 11.3C.

The power of deep learning emerges when several AEs are stacked to form a stacked auto-encoder (SAE), where each AE becomes a building block in the deep learning model. In order to train the SAE model, we use greedy layer-wise learning



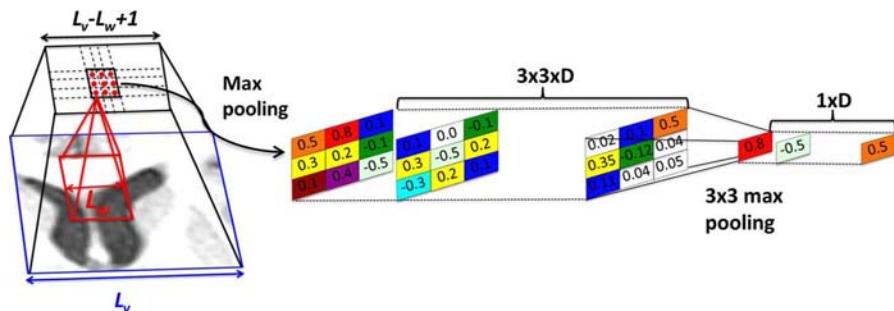
**FIGURE 11.4**

The hierarchical architecture of stacked auto-encoder (SAE).

[46,47] to train a single AE at each time. Specifically, there are three steps in training SAE, i.e., (i) pre-training, (ii) unrolling, and (iii) fine-tuning [13]. In the pre-training step, we train the first AE with all image patches as the input. Then, we train the second AE by using the activations  $h^{(1)}$  of the first AE (pink circles in Fig. 11.4) as the input. In this way, each layer of the features captures strong, high-order correlations based on outputs from the layer below. This layer-by-layer learning can be repeated many times. After pre-training, we build a deep learning network by stacking the AE in each layer, with the higher layer AE nested within the lower layer AE. Fig. 11.4 shows an SAE consisting of 2-layer stacked AEs. Since the layer-by-layer pre-training procedure provides a good initialization for the multi-level network, we can efficiently use the gradient-based optimization method (e.g., L-BFGS or Conjugate Gradient [71]) to further refine the parameters in the fine-tuning stage. Due to the deep and hierarchical nature of the network structure, SAE can discover highly non-linear and complex feature representations for patches in medical images. As shown in Figs. 11.3D and 11.3E, the patch reconstruction performance by SAE is much better than using a single AE, where the SAE consists of only 2 layers and the numbers of hidden nodes in each layer are 255 and 100, respectively.

### 11.2.2.3 Convolutional SAE Network (CSAE)

Due to the complex nature of medical images, learning the latent feature representations by employing deep learning is much more difficult than similar applications in computer vision and machine learning areas. In particular, the dimension of input training patch is often very high. For example, the intensity vector of a 3D image patch with the size of  $21 \times 21 \times 21$  has 9261 elements. Thus, the training of SAE network becomes very challenging. To alleviate this issue, we resort to CSAE, which is based on efficient convolution layers to both model the locality of the pixel correlations as shared features and significantly increase the computational performance.

**FIGURE 11.5**

The  $3 \times 3$  max pooling procedure in convolutional network.

CSAE differs from SAE as it uses convolution to take advantage of the locality of image features, and shares its weights among locations in the image patches. The convolution of an  $m \times m$  matrix with an  $n \times n$  window may in fact result in an  $(m+n-1) \times (m+n-1)$  matrix (full convolution) or in an  $(m-n+1) \times (m-n+1)$  (valid convolution). The convolution filter strides over the entire 3D image patch without overlap. The process of successive filtration finally can make a much less noisy representation of the input 3D image. To make the feature representation more spatially invariant and reduce the numbers of dimensions, we use a max pooling technique to down-sample the convolutional representation by a constant factor to take the maximum value over non-overlapping sub-regions. As shown in Fig. 11.5, the input to the convolutional SAE network is the large image patch  $P_v$  with patch size  $L_v$ . To make it simple, we explain the convolutional SAE network with 2D image patch as the example. Since the dimension of the image patch  $P_v$  is too large, we let an  $L_w \times L_w$  ( $L_w < L_v$ ) sliding window  $P_w$  (red box in Fig. 11.5) go through the entire large image patch  $P_v$ , thus obtaining  $(L_v - L_w + 1) \times (L_v - L_w + 1)$  small image patches. Eventually, we use these small image patches  $P_w$  to train the auto-encoder in each layer, instead of the entire image patch  $P_v$ . Given the parameters of network (weight matrix  $W$  and bias vector  $b_1$  and  $b_2$ ), we can compute  $(L_v - L_w + 1) \times (L_v - L_w + 1)$  activation vectors, where we use the red dots in Fig. 11.5 to denote the activation vectors in a  $3 \times 3$  neighborhood. Then, the max pooling is used to shrink the representations by a factor of  $C$  in each direction (horizontal or vertical). The right part of Fig. 11.5 demonstrates the  $3 \times 3$  max pooling procedure ( $C = 3$ ). Specifically, we compute the representative activation vector among these 9 activation vectors in the  $3 \times 3$  neighborhood by choosing the maximum absolute value for each vector element. Thus, the number of activation vector significantly reduces to  $\frac{L_v - L_w + 1}{C} \times \frac{L_v - L_w + 1}{C}$ . Since we apply the maximum operation, shrinking the representation with max pooling allows high-level representation to be invariant to small translations of the input image patches and reduces the computational burden. This translation

invariance is advantageous for establishing anatomical correspondences between two images, as is demonstrated in our experiments.

#### **11.2.2.4 Patch Sampling**

Typically, one brain MR image, with  $1 \times 1 \times 1 \text{ mm}^3$  spatial resolution, has over 8 million voxels in the brain volume. As a result, there are too many image patches to train the deep learning network. Therefore, adaptive sampling strategy is necessary to secure not only using enough image patches, but also selecting the most representative image patches to learn the latent feature representations for the entire training set.

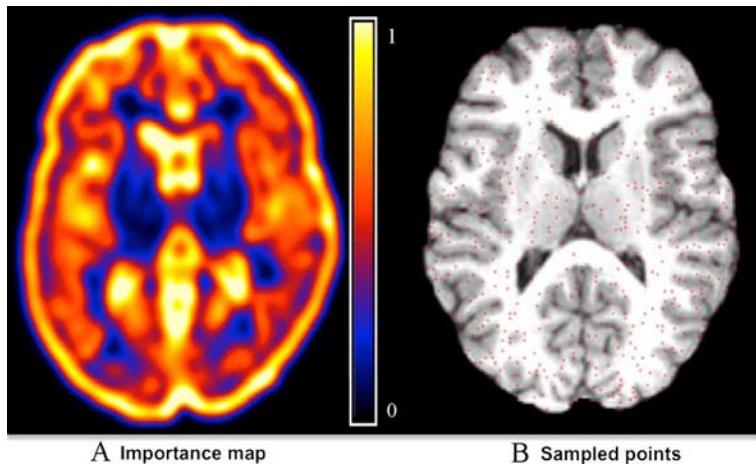
To this end, there are two criteria for sampling image patches: (i) In a local view, the selected image patches should locate at distinctive regions in the image, such as sulcal roots and gyral crowns in MR brain images, since they are relatively easy to identify their correspondences; (ii) In a global view, the selected image patches should cover the entire brain volume, while the density of sampled points could be low in uniform regions and high in context-rich regions. To meet the criteria, we use the importance sampling strategy [24] to hierarchically sample the image patches. Specifically, we smooth and normalize the gradient magnitude values over the whole image domain of each training image. Then, we use the obtained values as the importance degree (or probability) of each voxel to be sampled for deep learning. Note that, although more sophisticated method [72] could be used here for guiding sample selection, we use a simple gradient guided strategy since it is computationally fast. Based on this importance (probability) map, a set of image patches can be sampled, via Monte Carlo simulation in each image. Fig. 11.6A shows the non-uniform sampling based on the importance (probability) map in learning the intrinsic feature representations for MR brain images. It can be observed from Fig. 11.6 that the sampled image patches (with the center point of each sampled patch denoted by the red dot in Fig. 11.6B) are more concentrated at the context-rich (or edge-rich) regions, where the values of importance (or probability) are high.

### **11.2.3 REGISTRATION USING LEARNED FEATURE REPRESENTATIONS**

#### **11.2.3.1 Advantages of Feature Representations Learned by Deep Learning Network**

The advantage of using deep learning-based features in registration is demonstrated in Fig. 11.7. A typical image registration result for the elderly brain images is shown in the top of Fig. 11.7, where the deformed subject image (Fig. 11.7C) is way of being well registered with the template image (Fig. 11.7A), especially for ventricles. Obviously, it is very difficult to learn meaningful features given the inaccurate correspondences derived from imperfect image registration, as suffered by many supervised learning methods.

The discriminative power of our learned features is shown in Fig. 11.7F. For a template point (indicated by the red cross in Fig. 11.7A), we can successfully find its corresponding point in the subject image, whose ventricle is significantly larger.

**FIGURE 11.6**

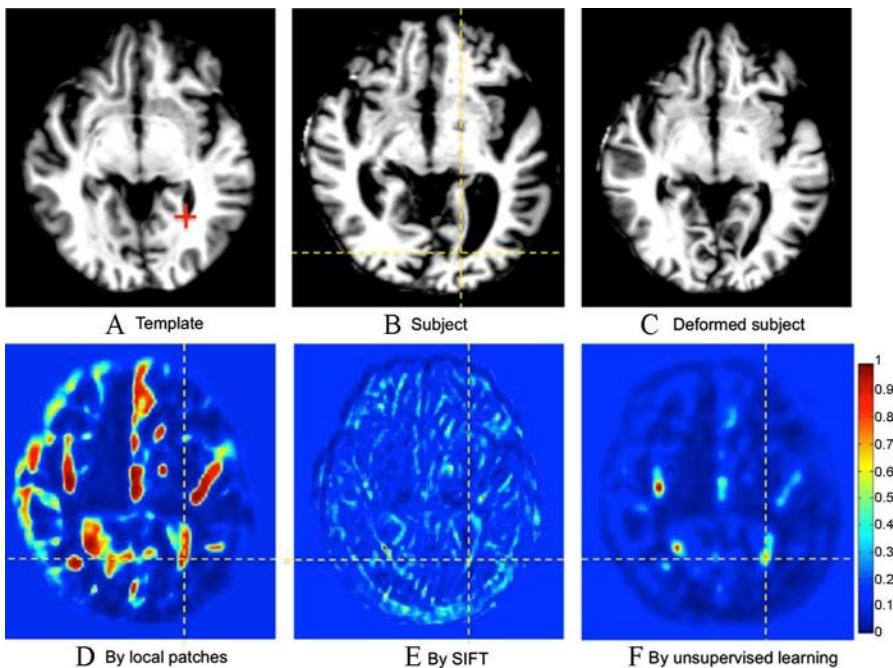
The importance map and the sampled image patches (denoted by the red dots) for deep learning. The color bar indicates the varying importance values for individual voxels.

Other handcrafted features either detect too many non-corresponding points (when using the entire intensity patch as the feature vector as shown in Fig. 11.7D) or have too low responses, and thus miss the correspondence (when using SIFT features as shown in Fig. 11.7E). In general, our method reveals the least confusing correspondence information for the subject point under consideration, and implies the best correspondence detection performance.

#### **11.2.3.2 Learning-Based Image Registration Framework**

After training the convolutional SAE on a large amount of 3D image patches, it is efficient to obtain the low-dimensional feature representations (blue circles in Fig. 11.4) by simple matrix multiplication and addition in each encoder layer. Such low-dimensional feature representation, regarded as the morphological signature, allows each point to accurately identify the correspondence during image registration, as demonstrated above. Here, we use normalized cross correlation as the similarity measurement between the feature representation vectors of the two different points under comparison.

Since the convolutional SAE can directly learn feature representations from the observed data, the learning procedure can be free of the limitation of requiring ground-truth data. Thus, it is straightforward to learn optimal feature representations for specific dataset, with little or even no human intervention. Then, we can incorporate the state-of-the-art deformation mechanisms developed in many registration methods in a learning-based registration framework by replacing with the learned feature representations and still inheriting the existing deformation mechanism to derive the deformation pathway.

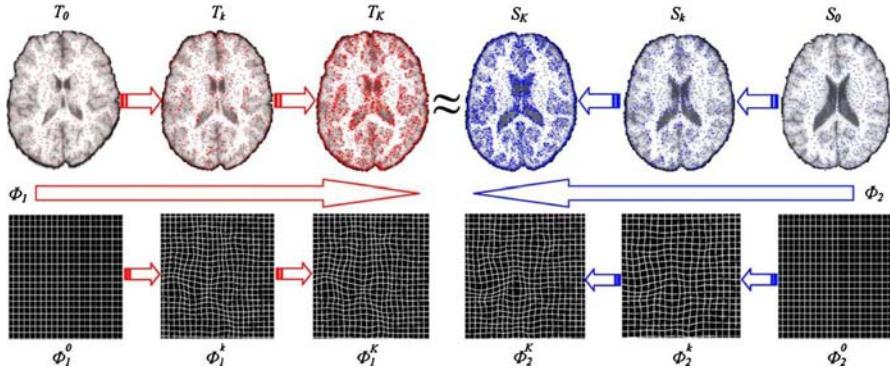


**FIGURE 11.7**

The similarity maps of identifying the correspondence of the red-crossed point in the template (A) w.r.t. the subject (B) by handcraft features (D)–(E) and the learned features by unsupervised deep learning (F).

Without loss of generalization, we show two examples by integrating the feature representations learned via deep learning using state-of-the-art registration methods. First, many image registration methods, e.g., diffeomorphic Demons [23], use the gradient-based optimization approach to iteratively estimate the deformation fields. To utilize multiple image information, such as multi-modality images, a multi-channel Demons algorithm was proposed by allowing one channel carrying one information source [73–75]. Therefore, it is straightforward to deploy multi-channel Demons, by regarding all elements of the learned feature representations as multiple channels.

Second, we can replace the handcrafted attribute vectors (i.e., the low-order statistics in multi-resolution histograms) in the feature-based registration method, e.g., symmetric HAMMER [27], with the learned feature representations by the convolutional SAE. Specifically, after training the convolution SAE upon large amount of 3D image patches, it is efficient to obtain the low-dimensional feature representations (blue circles in Fig. 11.2 and Fig. 11.4) by simple matrix multiplication and addition in each encoder layer. Such low-dimensional feature representation, regarded as the morphological signature, allows each point to accurately identify the correspondence

**FIGURE 11.8**

The demonstration of hierarchical feature matching mechanism with symmetric estimation of deformation pathway.

by robust feature matching [28,76]. Next, we simultaneously deform template and subject images toward each other until they meet at the middle point. Thus, we can obtain two deformation pathways,  $\phi_1$  from template  $T$ , and  $\phi_2$  from subject  $S$ , as denoted by red and blue arrows in Fig. 11.8, respectively. In the end, the deformation field from template to subject can be calculated by  $F = \phi_1 \circ \phi_2^{-1}$ , where  $\circ$  denotes the composition of two deformation pathways [20]. To further improve the correspondence detection, a small number of key points (denoted in red and blue for template and subject in Fig. 11.8, respectively) with more distinctive features than other image points will be selected to drive the entire deformation field by requiring deformation on less distinctive points to follow the correspondences of nearby key points. The key points are hierarchically selected during registration by using non-uniform sampling strategy, which ensures that most key points are located at salient regions and also cover the whole image. Since  $\phi_1$  and  $\phi_2$  are iteratively refined during registration, we use  $k$  to denote the iteration. Thus, in the beginning of registration ( $k = 0$ ),  $T_0 = T$  and  $S_0 = S$ , along with identity deformation pathway  $\phi_1^0$  and  $\phi_2^0$ . With the progress of registration, the template image  $T$  gradually deforms to  $T^k = T(\phi_1^k)$ . Similarly, subject image  $S$  deforms to  $S^k = S(\phi_2^k)$ . In the meantime, more key points are selected to refine the deformation pathways  $\phi_1^k$  and  $\phi_2^k$  w.r.t.  $T^k$  and  $S^k$ , which is repeated until the deformed template  $T^k$  and deformed subject  $S^k$  become very similar at the end of registration.

### 11.3 EXPERIMENTS

Here, we evaluate the performance of deformable image registration algorithm that uses deep learning for feature selection. For comparison, we set diffeomorphic

Demons and HAMMER as baselines for intensity-based and feature-based registration methods, respectively. Then, we extend the diffeomorphic Demons from a single channel (i.e., image intensity) to multi-channel Demons by adapting the learned feature representations via deep learning to multiple channels (M + DP). Similarly, we modify HAMMER to use the feature representations learned via deep learning (H + DP). Since PCA is widely used for unsupervised learning, we apply PCA to infer the latent low-dimensional feature representations. After integrating such low-dimensional feature representations by PCA into multi-channel Demons and HAMMER, we can obtain two other new registration methods, denoted as M + PCA and H + PCA, respectively.

### 11.3.1 EXPERIMENTAL RESULT ON ADNI DATASET

We randomly select 66 MR images from the ADNI dataset (<http://adni.loni.ucla.edu/>), where 40 images are used to learn feature representations and the other 26 images are used to test image registration. The preprocessing steps include skull removal [77], bias correction [78], and intensity normalization [79]. For each training image, we sample around 7000 image patches, where the patch size is set to  $21 \times 21 \times 21$ . The convolutional SAE consists of 8 layers (stacked with 4 AEs). We only apply the max pooling in the lowest layer with the pooling factor  $C = 3$ . From the lowest to the highest level, the numbers of hidden nodes in each stacked AE are 512, 512, 256, and 128, respectively. Thus, the dimension of final feature representations after deep learning is 128. To keep the similar dimension of learned features by PCA, we set the portion of remaining energy  $f_Q$  to 0.7 in this experiment.

In image registration, one image is selected as the template and the other 25 images are considered as subject images. Before deploying deformable image registration, FLIRT in FSL package (<http://fsl.fmrib.ox.ac.uk/fsl/fslwiki/>) is used to linearly align all subject images to the template image. After that, we apply 6 registration methods, i.e., diffeomorphic Demons (simply named as Demons below), M + PCA, M + DP, HAMMER, H + PCA, and H + DP, to normalize those 25 subject images to the template image space, respectively. To quantitatively evaluate the registration accuracy, we first use FAST in FSL package to segment each image into white matter (WM), gray matter (GM), and cerebral-spinal fluid (CSF). After that, we further separate the ventricle (VN) from the CSF segmentation. Here, we use these segmentation results to evaluate the registration accuracy by comparing the Dice ratio of tissue overlap degrees between the template and each registered subject image. Specifically, the Dice ratio is defined as

$$D(R_A, R_B) = \frac{2|R_A \cap R_B|}{|R_A| + |R_B|}, \quad (11.4)$$

where  $R_A$  and  $R_B$  denote two ROIs and  $|\cdot|$  stands for the volume of the region. The Dice ratios on WM, GM, and VN by 6 registration methods are shown in [Table 11.1](#). It is clear that (i) the registration methods, integrated with the feature representations by deep learning, consistently outperform the counterpart baseline methods and also

**Table 11.1** The Dice ratios of WN, GM, and VN on ADNI dataset (in %)

Method	WM	GM	VN	Overall
Demons	85.7	76.0	90.2	84.0
M + PCA	85.5	76.6	90.2	84.1
M + DP	85.8	76.5	90.9	84.4
HAMMER	85.4	75.5	91.5	84.1
H + PCA	86.5	76.9	91.7	85.0
H + DP	88.1	78.6	93.0	86.6

**Table 11.2** The Dice ratio of hippocampus on ADNI dataset (in %)

Method	Demons	M + PCA	M + DP	HAMMER	H + PCA	H + DP
Dice ratio	$72.2 \pm 3.1$	$72.3 \pm 2.9$	$72.5 \pm 2.8$	$75.5 \pm 2.9$	$75.6 \pm 2.5$	$76.8 \pm 2.2$

use PCA-based feature representations only; (ii) H + DP achieves the highest registration accuracy with almost 2.5% improvement in overall Dice ratio, compared to the baseline HAMMER registration method. Since ADNI provides hippocampus labeling for the template and all 25 subject images, we can further evaluate the Dice overlap ratio on hippocampus. The mean and the standard deviation of the Dice ratios on hippocampus by 6 registration methods are shown in Table 11.2. Compared to the baseline methods, M + DP and H + DP obtain 0.3% and 1.3% improvements in terms of Dice ratios, respectively. Particularly, the reason of the less improvement by M + DP compared to H + DP might be related with the high number of channels (128 channels) used in M + DP, compared with only less than 10 channels used in [73–75].

### 11.3.2 EXPERIMENTAL RESULT ON LONI DATASET

In this experiment, we use the LONI LPBA40 dataset [80], which consists of 40 brain images, each with 56 manually labeled ROIs. We randomly use 20 images to learn the latent feature representations, and the other 20 images to test registration performance. The preprocessing procedures include bias correction, intensity normalization, and linear registration by FLIRT, which are the same as in Section 11.3.1. For each training image, we sample around 9000 image patches, where the patch size is again set to  $21 \times 21 \times 21$ . Other parameters in training convolutional SAE are also the same as in Section 11.3.1. Therefore, the dimension of feature representations after deep learning is 128. To keep the similar dimension of learned features by PCA, we set  $f_Q$  to 0.65 in this experiment.

One of the 20 testing images is selected as the template, and then we apply 6 registration methods to register the rest of 19 testing images to the selected template. The averaged Dice ratio in each ROI by the 6 registration methods is shown in Fig. 11.9. The overall Dice ratios across all 56 ROIs by the 6 registration methods are provided

	Demons	M+PCA	M+DP	HAMMER	H+PCA	H+DP
<i>L sup. frontal gyrus</i>	80.2	79.5	79.1	77.3	77.1	78.5
<i>R sup. frontal gyrus</i>	79.7	79.8	80.0	77.5	77.2	78.4
* <i>L middle frontal gyrus</i>	77.4	78.1	78.2	79.5	78.5	82.3
* <i>R middle frontal gyrus</i>	77.0	76.5	77.1	78.2	78.6	80.4
<i>L inf. frontal gyrus</i>	72.2	72.8	72.6	72.8	73.0	74.6
<i>R inf. frontal gyrus</i>	72.0	72.1	72.3	72.6	72.4	74.1
<i>L precentral gyrus</i>	67.9	68.5	68.4	68.6	69.1	71.5
<i>R precentral gyrus</i>	68.6	68.5	69.0	65.0	65.2	65.1
* <i>L middle orbitofrontal gyrus</i>	66.9	66.1	67.1	69.8	69.9	73.5
* <i>R middle orbitofrontal gyrus</i>	66.8	67.5	67.7	69.4	69.5	73.9
* <i>L lateral orbitofrontal gyrus</i>	58.1	58.1	58.5	60.5	61.5	64.9
<i>R lateral orbitofrontal gyrus</i>	55.4	55.6	55.7	64.7	64.1	68.9
<i>L gyrus rectus</i>	66.7	66.1	66.9	67.9	67.2	69.8
<i>R gyrus rectus</i>	68.1	67.7	68.1	65.5	65.0	65.0
<i>L postcentral gyrus</i>	60.5	60.7	61.4	60.5	61.2	63.0
* <i>R postcentral gyrus</i>	62.9	62.4	62.5	63.5	63.1	65.2
* <i>L sup. parietal gyrus</i>	70.7	70.9	70.6	72.6	72.7	74.7
* <i>R sup. parietal gyrus</i>	70.9	70.6	71.2	71.8	71.9	73.5
<i>L supramarginal gyrus</i>	63.8	63.4	64.1	65.1	65.5	67.8
<i>R supramarginal gyrus</i>	63.3	63.7	63.8	65.1	66.4	68.5
* <i>L angular gyrus</i>	63.2	62.8	63.5	66.9	66.8	70.0
<i>R angular gyrus</i>	65.0	65.1	65.7	65.4	65.8	67.5
* <i>L precuneus</i>	65.9	65.7	66.4	70.6	70.9	74.0
* <i>R precuneus</i>	67.3	67.2	67.8	70.8	71.5	76.5
<i>L sup. occipital gyrus</i>	58.1	58.0	58.2	61.2	62.4	65.5
<i>R sup. occipital gyrus</i>	55.4	55.9	56.2	64.5	65.4	67.7
* <i>L middle occipital gyrus</i>	68.7	68.5	68.4	72.6	74.9	80.6
<i>R middle occipital gyrus</i>	67.9	67.4	67.8	71.1	72.0	73.1
<i>L inf. occipital gyrus</i>	67.2	67.8	67.9	65.8	65.5	66.4
<i>R inf. occipital gyrus</i>	66.1	66.5	67.1	62.0	62.0	62.1
<i>L cuneus</i>	63.4	63.4	63.9	64.1	64.5	64.7
* <i>R cuneus</i>	62.2	62.4	62.5	66.0	67.2	70.1
<i>L sup. temporal gyrus</i>	72.5	72.5	72.7	69.5	69.5	70.7
* <i>R sup. temporal gyrus</i>	72.6	73.1	73.4	74.1	74.5	76.4
* <i>L middle temporal gyrus</i>	66.4	66.8	66.8	67.1	66.9	69.5
<i>R middle temporal gyrus</i>	67.9	67.5	67.9	68.3	68.4	69.7
<i>L inf. temporal gyrus</i>	65.6	65.2	65.9	65.8	65.1	66.3
* <i>R inf. temporal gyrus</i>	66.4	66.4	66.5	66.9	67.8	70.0
<i>L parahippocampal gyrus</i>	68.1	68.2	68.5	68.0	69.1	70.1
<i>R parahippocampal gyrus</i>	66.9	66.7	67.2	67.5	69.0	71.0
* <i>L lingual gyrus</i>	69.7	69.8	68.9	69.4	69.2	71.8
* <i>R lingual gyrus</i>	70.6	70.5	70.6	73.6	74.3	77.5
<i>L fusiform gyrus</i>	68.9	68.8	69.1	66.5	66.1	66.2
<i>R fusiform gyrus</i>	68.3	68.3	68.5	67.5	67.5	67.8
<i>L insular cortex</i>	76.4	76.1	76.5	77.5	77.9	79.7
<i>R insular cortex</i>	74.2	74.6	74.7	75.1	76.0	76.1
* <i>L cingulate gyrus</i>	68.1	68.2	68.8	69.5	69.9	71.4
* <i>R cingulate gyrus</i>	67.5	67.4	67.2	69.2	70.5	72.2
* <i>L caudate</i>	73.4	73.4	73.8	74.5	75.0	77.8
* <i>R caudate</i>	73.1	73.0	73.5	76.2	76.4	78.3
* <i>L putamen</i>	76.3	76.5	76.7	77.0	77.7	80.0
* <i>R putamen</i>	76.5	76.3	76.4	76.5	78.6	80.6
* <i>L hippocampus</i>	72.7	72.6	72.8	74.7	75.8	77.7
* <i>R hippocampus</i>	72.8	72.6	73.1	75.9	77.1	81.3
* <i>cerebellum</i>	84.9	85.1	85.9	86.0	87.8	90.3
* <i>brainstem</i>	80.6	81.4	83.1	85.5	86.6	89.1

FIGURE 11.9

The Dice ratios of 56 ROIs on LONI dataset by 6 registration methods.

**Table 11.3** The overall Dice ratio of 54 ROIs on LONI dataset (in %)

Method	Demons	M + PCA	M + DP	HAMMER	H + PCA	H + DP
Dice ratio	68.9	68.9	69.2	70.2	70.6	72.7

**FIGURE 11.10**

The Dice ratios of 56 ROIs in LONI dataset by HAMMER (blue), H + DP-LONI (red), and H + DP-ADNI (green), respectively. Note, H + DP-LONI denotes for the HAMMER registration integrating with the feature representations learned directly from LONI dataset, while H + DP-ADNI stands for applying HAMMER registration on LONI dataset but using the feature representations learned from ADNI dataset, respectively.

in Table 11.3. Again, H + DP achieves the largest improvement (2.5%) in comparison to the baseline HAMMER registration method. Specifically, we perform the paired *t*-test between H + DP and all other 5 registration methods, respectively. The results indicate that H + CP has the statistically significant improvement over all other 5 registration methods in 28 out of 54 ROIs (designated by the red stars in Fig. 11.9).

Recall that we have obtained the feature representations by deep learning on the ADNI dataset in Section 11.3.1. It is interesting to evaluate the generality of deep learning by applying the learned feature representations from the ADNI dataset (which mostly contains elderly brains) to register the images in the LONI dataset (i.e., young brains). Fig. 11.10 shows the Dice ratio in each ROI in the LONI dataset by (i) the baseline HAMMER registration method (in blue), (ii) H + DP-LONI (in red), which denotes we learned the feature representations from LONI dataset and integrated the learned feature representations with HAMMER, and (iii) H+DP-ADNI (in

green) where we apply the learned feature representations from ADNI dataset as the morphological signature to drive the registration on the LONI dataset. It is apparent that the registration performance by H + DP-ADNI is comparable to H + DP-LONI, where the average improvements over the baseline HAMMER registration method are 1.99% by H + DP-ADNI, and 2.55% by H + DP-LONI, respectively. It indicates that the learned feature representations by convolutional SAE network are general, although the appearances of two dataset can be quite different (i.e., due to aging).

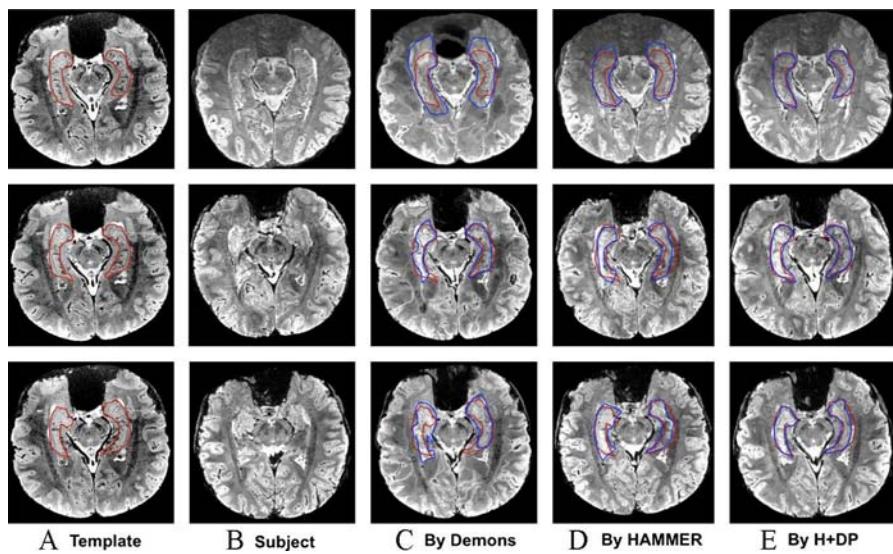
### 11.3.3 EXPERIMENTAL RESULT ON 7.0-T MR IMAGE DATASET

In previous experiments, we demonstrated the power of learned feature representations by deep learning in terms of the improved registration accuracy, which is represented by an overlap ratio of structures. As mentioned earlier, another attractive advantage of deep learning is that it can rapidly learn intrinsic feature representations for a new imaging modality. In this section, we apply the convolutional SAE to 7.0 T MR brain images. The learned feature representations are then integrated with HAMMER, which allows us to develop a specific registration method for 7.0 T MR brain images.

The advent of 7.0-T MR imaging technology [81] achieves high signal-to-noise ratio (SNR), as well as the dramatically increased tissue contrast, compared to the 1.5 or 3.0 T MR image. A typical 7.0-T MR brain image (with the image spatial resolution of  $0.35 \times 0.35 \times 0.35 \text{ mm}^3$ ) is shown in Fig. 11.11B, along with a similar slice from a 1.5 T scanner (with the resolution of  $1 \times 1 \times 1 \text{ mm}^3$ ) in Fig. 11.11A for comparison. As demonstrated in [82], 7.0 T MR image can reveal brain structures with resolution equivalent to that obtained from thin slices *in vitro*. Thus, researchers are able to clearly observe the fine brain structures on the  $\mu\text{m}$  scale, which was only possible with *in vitro* imaging previously. Without a doubt, 7.0 T MR imaging technique has the potential to become the standard method in discovering morphological patterns of human brain in the near future.

Unfortunately, all existing state-of-the-art deformable registration methods, developed for 1.5 or 3.0 T MR images, do not work well for the 7.0 T MR images, mainly because (i) severe intensity inhomogeneity issue in 7.0 T MR images, and because (ii) much richer texture information than that in 1.5 or 3.0 T MR images, as displayed in Fig. 11.1.

Overall, 20 7.0 T MR images acquired by the method in [81] were used in this experiment, where 10 are used to train deep learning, and the other 10 images are used to test registration performance. We randomly select one image as the template. For the 7.0 T scanner (Magnetom, Siemens), an optimized multichannel radiofrequency (RF) coil and a 3D fast low-angle shot (Spoiled FLASH) sequence were utilized, with TR = 50 ms, TE = 25 ms, flip angle 10°, pixel band width 30 Hz/pixel, field of view (FOV) 200 mm, matrix size  $512 \times 576 \times 60$ , 3/4 partial Fourier, and number of average (NEX) 1. The image resolution of the acquired images is isotropic, e.g.,  $0.35 \times 0.35 \times 0.35 \text{ mm}^3$ . The hippocampi were manually segmented by a neurologist [81]. All images were pre-processed by the following steps: (i) inhomogeneity cor-



**FIGURE 11.11**

Typical registration results on 7.0-T MR brain images by Demons, HAMMER, and H + DP, respectively. Three rows represent three different slices in the template, subject, and registered subjects.

rection using N4 bias correction [78]; (ii) intensity normalization for making image contrast and luminance consistent across all subjects [79]; (iii) affine registration to the selected template by FSL.

For each training image, we sample around 9000 image patches, where the patch size is set to  $27 \times 27 \times 27$ . The convolutional SAE consists of 10 layers (stacked with 5 AEs). We only apply the max pooling in the lowest layer, with the pooling factor  $C = 3$ . From low level to high level, the numbers of hidden nodes in each stacked AE are 1024, 512, 512, 256, and 128, respectively. Thus, the dimension of deep learning-based feature representations after deep learning is 128. In order to achieve the best registration performance, we integrate the learned feature representations trained from 7.0 T MR images with the HAMMER registration method.

Several typical registration results on 7.0 T MR images are displayed in Fig. 11.11, where the template and subject images are shown in Figs. 11.11A and 11.11B, respectively. Here, we compare the registration results with diffeomorphic Demons (Fig. 11.11C) and HAMMER (Fig. 11.11D). The registration results by H + DP, i.e., integrating the learned feature representations by deep learning with HAMMER, are display in Fig. 11.11E, where the manually labeled hippocampus on the template image, and the deformed subject's hippocampus, by different registration methods, are shown by red and blue contours, respectively. Through visual inspection (the overlap of red and blue contours), the registration result by

H + DP is much better than both diffeomorphic Demons and HAMMER. Since we have manually labeled hippocampus for the template and all subject images, we can further quantitatively measure the registration accuracy. The mean and standard deviation of Dice ratios on hippocampus are  $(53.5 \pm 4.9)\%$  by diffeomorphic Demons,  $(64.9 \pm 3.1)\%$  by HAMMER, and  $(75.3 \pm 1.2)\%$  by H + DP, respectively. Obviously, H + DP achieves significant improvement in registration accuracy. This experiment demonstrates that (i) the latent feature representations inferred by deep learning can well describe the local image characteristics; (ii) we can rapidly develop image registration for new medical imaging modalities by using deep learning framework to learn the intrinsic feature representations; and (iii) the whole learning-based framework is fully adaptive to learn the image data and reusable to various medical imaging applications.

---

## 11.4 CONCLUSION

In this book chapter, a new deformable image registration approach that uses deep learning for feature selection was introduced. Specifically, we proposed an unsupervised deep learning feature selection framework that implements a convolutional-stacked auto-encoder network (SAE) to identify the intrinsic features in the 3D image patches. Using the LONI and ADNI datasets, the image registration performance was compared to two existing state-of-the-art deformable image registration frameworks that use handcrafted features. The results show that the new image registration framework consistently demonstrated better Dice ratio scores, when compared to state-of-the-art methods. In short, because the trained deep learning network selected features more accurately capture the complex morphological patterns in the image patches, it is allowed to produce better anatomical correspondences, which ultimately resulted in better image registration performance.

To demonstrate the scalability of the proposed registration framework, image registration experiments were also conducted on 7.0 T brain MR images. Likewise, the results showed that the new image registration framework consistently demonstrated better Dice ratio scores, when compared to state-of-the-art methods. Unlike those existing image registration frameworks, the deep learning architecture can be quickly developed, and trained using no ground-truth data with superior performance, which allows deep learning architecture to be eventually applied to new imaging modalities with the least effort.

---

## REFERENCES

1. T. Paus, et al., Structural maturation of neural pathways in children and adolescents: in vivo study, *Science* 283 (5409) (1999) 1908–1911.

2. E.R. Sowell, et al., Localizing age-related changes in brain structure between childhood and adolescence using statistical parametric mapping, *NeuroImage* 9 (6 (Part 1)) (1999) 587–597.
3. P.M. Thompson, et al., Growth patterns in the developing brain detected by using continuum mechanical tensor maps, *Nature* 404 (2000) 190–193.
4. S. Resnick, P. Maki, Effects of hormone replacement therapy on cognitive and brain aging, *Ann. N.Y. Acad. Sci.* 949 (2001) 203–214.
5. P.M. Thompson, et al., Tracking Alzheimer's Disease, *Ann. N.Y. Acad. Sci.* 1097 (1) (2007) 183–214.
6. J. Lerch, et al., Automated cortical thickness measurements from MRI can accurately separate Alzheimer's patients from normal elderly controls, *Neurobiol. Aging* 29 (1) (2008) 23–30.
7. N. Schuff, et al., MRI of hippocampal volume loss in early Alzheimer's disease in relation to ApoE genotype and biomarkers, *Brain* (2009) 1067–1077.
8. G. Frisoni, et al., In vivo neuropathology of the hippocampal formation in AD: a radial mapping MR-based study, *NeuroImage* 32 (1) (2006) 104–110.
9. L. Apostolova, et al., Conversion of mild cognitive impairment to Alzheimer disease predicted by hippocampal atrophy maps, *Arch. Neurol.* 64 (9) (2007) 1360–1361.
10. A.D. Leow, et al., Alzheimer's Disease Neuroimaging Initiative: a one-year follow up study using tensor-based morphometry correlating degenerative rates, biomarkers and cognition, *NeuroImage* 45 (3) (2009) 645–655.
11. D. Shen, C. Davatzikos, HAMMER: hierarchical attribute matching mechanism for elastic registration, *IEEE Trans. Med. Imaging* 21 (11) (2002) 1421–1439.
12. J. Maintz, M. Viergever, A survey of medical image registration, *Med. Image Anal.* 2 (1) (1998) 1–36.
13. G.E. Hinton, R.R. Salakhutdinov, Reducing the dimensionality of data with neural networks, *Science* 313 (5786) (2006) 504–507.
14. P. Thompson, A. Toga, A framework for computational anatomy, *Comput. Vis. Sci.* 5 (2002) 13–34.
15. C. Davatzikos, et al., Voxel-based morphometry using the RAVENS maps: methods and validation using simulated longitudinal atrophy, *NeuroImage* 14 (6) (2001) 1361–1369.
16. Y. Li, et al., Discriminant analysis of longitudinal cortical thickness changes in Alzheimer's disease using dynamic and network features, *Neurobiol. Aging* 33 (2012) 415–427.
17. C. Saw, et al., Multimodality image fusion and planning and dose delivery for radiation therapy, *Med. Dosim.* 33 (2008) 149–155.
18. J. Cízek, et al., Fast and robust registration of PET and MR images of human brain, *NeuroImage* 22 (1) (2004) 434–442.
19. V. Arsigny, et al., A log-euclidean framework for statistics on diffeomorphisms, in: *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2006*, 2006, pp. 924–931.
20. J. Ashburner, A fast diffeomorphic image registration algorithm, *NeuroImage* 38 (1) (2007) 95–113.
21. B. Avants, M. Grossman, J. Gee, Symmetric diffeomorphic image registration: evaluating automated labeling of elderly and neurodegenerative cortex and frontal lobe, in: *Biomedical Image Registration*, 2006.
22. D. Rueckert, et al., Nonrigid registration using free-form deformations: application to breast MR images, *IEEE Trans. Med. Imaging* 18 (8) (1999) 712–721.

23. T. Vercauteren, et al., Diffeomorphic demons: efficient non-parametric image registration, *NeuroImage* 45 (1, Suppl. 1) (2009) S61–S72.
24. Q. Wang, et al., Attribute vector guided groupwise registration, *NeuroImage* 50 (4) (2010) 1485–1496.
25. T. Rohlfing, Image similarity and tissue overlaps as surrogates for image registration accuracy: widely used but unreliable, *IEEE Trans. Med. Imaging* 31 (2) (2012) 153–160.
26. D. Shen, C. Davatzikos, HAMMER: hierarchical attribute matching mechanism for elastic registration, *IEEE Trans. Med. Imaging* 21 (11) (2002) 1421–1439.
27. G. Wu, et al., S-HAMMER: hierarchical attribute-guided, symmetric diffeomorphic registration for MR brain images, *Hum. Brain Mapp.* 35 (3) (2014) 1044–1060.
28. G. Wu, et al., TPS-HAMMER: improving HAMMER registration algorithm by soft correspondence matching and thin-plate splines based deformation interpolation, *NeuroImage* 49 (3) (2010) 2225–2233.
29. Y. Zhan, D. Shen, Deformable segmentation of 3-D ultrasound prostate images using statistical texture matching method, *IEEE Trans. Image Process.* 25 (3) (2006) 256–272.
30. G. Wu, et al., Feature-based groupwise registration by hierarchical anatomical correspondence detection, *Hum. Brain Mapp.* 33 (2) (2012) 253–271.
31. G. Wu, F. Qi, D. Shen, Learning-based deformable registration of MR brain images, *IEEE Trans. Med. Imaging* 25 (9) (2006) 1145–1157.
32. G. Wu, F. Qi, D. Shen, Learning best features and deformation statistics for hierarchical registration of MR brain images, in: *Information Processing in Medical Imaging*, 2007, pp. 160–171.
33. Y. Ou, et al., DRAMMS: deformable registration via attribute matching and mutual-saliency weighting, *Med. Image Anal.* 15 (4) (2011) 622–639.
34. M. Kim, et al., Automatic hippocampus segmentation of 7.0 tesla MR images by combining multiple atlases and auto-context models, *NeuroImage* 83 (2013) 335–345.
35. R.C. Knickmeyer, et al., A structural MRI study of human brain development from birth to 2 years, *J. Neurosci.* 28 (47) (2008) 12176–12182.
36. P. Comon, Independent component analysis, a new concept?, *Signal Process.* 36 (3) (1994) 287–314.
37. A. Hyvärinen, J. Karhunen, E. Oja, *Independent Component Analysis*, John Wiley & Sons, 2001.
38. J.B. Tenenbaum, V. Silva, J.C. Langford, A global geometric framework for nonlinear dimensionality reduction, *Science* 290 (5500) (2000) 640–646.
39. E.-N. I, et al., A support vector machine approach for detection of microcalcifications, *IEEE Trans. Med. Imaging* 21 (12) (2002) 1552–1563.
40. H. Larochelle, et al., An empirical evaluation of deep architectures on problems with many factors of variation, in: *Proceedings of the 24th International Conference on Machine Learning*, Corvalis, Oregon, ACM, 2007, pp. 473–480.
41. S.T. Roweis, L.K. Saul, Nonlinear dimensionality reduction by locally linear embedding, *Science* 290 (5500) (2000) 2323.
42. M. Belkin, P. Niyogi, Laplacian eigenmaps for dimensionality reduction and data representation, *Neural Comput.* 15 (6) (2003) 1373–1396.
43. B.A. Olshausen, D.J. Field, Emergence of simple-cell receptive field properties by learning a sparse code for natural images, *Nature* 381 (1996) 607–609.
44. L. Arnold, et al., An introduction to deep-learning, in: *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, ESANN, 2011.

45. Y. Bengio, A. Courville, P. Vincent, Representation learning: a review and new perspectives, arXiv:1206.5538v3 [cs.LG], 23 Apr 2014.
46. Y. Bengio, et al., Greedy layer-wise training of deep networks, in: Advances in Neural Information Processing Systems, NIPS, 2006.
47. G.E. Hinton, S. Osindero, Y.-W. Teh, A fast learning algorithm for deep belief nets, *Neural Comput.* 18 (7) (2006) 1527–1554.
48. Q.V. Le, et al., Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis, in: IEEE Conference on Computer Vision and Pattern Recognition, CVPR, 2011.
49. H. Lee, C. Ekanadham, A.Y. Ng, Sparse deep belief net model for visual area V2, in: Advances in Neural Information Processing Systems, NIPS, 2008.
50. H. Lee, et al., Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations, in: Proceedings of the 26th Annual International Conference on Machine Learning, Montreal, Quebec, Canada, ACM, 2009, pp. 609–616.
51. H. Lee, et al., Unsupervised learning of hierarchical representations with convolutional deep belief networks, *Commun. ACM* 54 (10) (2011) 95–103.
52. Y.F. Li, C.Y. Chen, W.W. Wasserman, Deep feature selection: theory and application to identify enhancers and promoters, in: Recomb 2015, *Res. Comput. Mol. Biol.* 9029 (2015) 205–217.
53. Y. LeCun, Y. Bengio, Convolutional network for images, speech, and time series, in: *The Handbook of Brain Theory and Neural Networks*, 1995.
54. H.-C. Shin, et al., Stacked autoencoders for unsupervised feature learning and multiple organ detection in a pilot study using 4D patient data, *IEEE Trans. Pattern Anal. Mach. Intell.* 35 (8) (2013) 1930–1943.
55. D.W. Shattuck, et al., Construction of a 3D probabilistic atlas of human cortical structures, *NeuroImage* 39 (3) (2008) 1064–1080.
56. S. Mika, et al., Kernel PCA and denoising in feature space, *Adv. Neural Inf. Process. Syst.* 11 (1) (1999) 536–542.
57. A. Hyvarinen, P. Hoyer, Emergence of phase- and shift-invariant features by decomposition of natural images into independent feature subspaces, *Neural Comput.* 12 (7) (2000) 1705–1720.
58. J. MacQueen, Some methods for classification and analysis of multivariate observations, in: Proceeding of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, vol. 1, 1967, pp. 281–297.
59. D.M. Titterington, A.F.M. Smith, U.E. Makov, *Statistical Analysis of Finite Mixture Distributions*, John Wiley & Sons, 1985.
60. K. Mikolajczyk, C. Schmid, A performance evaluation of local descriptors, *IEEE Trans. Pattern Anal. Mach. Intell.* 27 (10) (2005) 1615–1630.
61. A. Coates, A.Y. Ng, Learning feature representations with K-means, in: *Neural Networks: Tricks of the Trade*, in: *Lect. Notes Comput. Sci.*, Springer, 2012.
62. T.F. Cootes, et al., Active shape models-their training and application, *Comput. Vis. Image Underst.* 16 (1) (1995) 38–159.
63. T.F. Cootes, G.J. Edwards, C.J. Taylor, Active appearance models, *IEEE Trans. Pattern Anal. Mach. Intell.* 23 (6) (2001) 681–685.
64. M.A. Ranzato, et al., Efficient learning of sparse representations with an energy-based model, in: Advances in Neural Information Processing Systems, NIPS, 2006.
65. R. Raina, et al., Self-taught learning: transfer learning from unlabeled data, in: 24th International Conference on Machine Learning, Corvalis, OR, 2007.

66. J. Hamm, et al., GRAM: a framework for geodesic registration on anatomical manifolds, *Med. Image Anal.* 14 (5) (2010) 633–642.
67. R. Wolz, et al., LEAP: learning embeddings for atlas propagation, *NeuroImage* 49 (2) (2010) 1316–1325.
68. R. Wolz, et al., Manifold learning for biomarker discovery, in: MICCAI Workshop on Machine Learning in Medical Imaging, Beijing, China, 2010.
69. D. Shen, Image registration by local histogram matching, *Pattern Recognit.* 40 (2007) 1161–1171.
70. D.E. Rumelhart, G.E. Hinton, R.J. Williams, Learning representations by back-propagating errors, *Nature* 323 (6088) (1986) 533–536.
71. Q.V. Le, et al., On optimization methods for deep learning, in: ICML, 2011.
72. T. Kadir, M. Brady, Saliency, scale and image description, *Int. J. Comput. Vis.* 45 (2) (2001) 83–105.
73. J.M. Peyrat, et al., Registration of 4D time-series of cardiac images with multichannel Diffeomorphic Demons, *Med. Image Comput. Comput. Assist. Interv.* 11 (Pt 2) (2008) 972–979.
74. J.M. Peyrat, et al., Registration of 4D cardiac CT sequences under trajectory constraints with multichannel diffeomorphic demons, *IEEE Trans. Med. Imaging* 29 (7) (2010) 1351–1368.
75. D. Forsberg, et al., Improving registration using multi-channel diffeomorphic demons combined with certainty maps, in: Multimodal Brain Image Registration, in: Lect. Notes Comput. Sci., vol. 7012, 2011.
76. H. Chui, A. Rangarajan, A new point matching algorithm for non-rigid registration, *Comput. Vis. Image Underst.* 89 (2–3) (2003) 114–141.
77. F. Shi, et al., LABEL: pediatric brain extraction using learning-based meta-algorithm, *NeuroImage* 62 (2012) 1975–1986.
78. N. Tustison, et al., N4ITK: improved N3 bias correction, *IEEE Trans. Med. Imaging* 29 (6) (2010) 1310–1320.
79. A. Madabhushi, J. Udupa, New methods of MR image intensity standardization via generalized scale, *Med. Phys.* 33 (9) (2006) 3426–3434.
80. D.W. Shattuck, M.M., V. Adisetiyo, C. Hojatkashani, G. Salamon, K.L. Narr, R.A. Poldrack, R.M. Bilder, A.W. Toga, Construction of a 3D probabilistic atlas of human cortical structures, *NeuroImage* 39 (3) (2008) 1064–1080.
81. Z.-H. Cho, et al., Quantitative analysis of the hippocampus using images obtained from 7.0 T MRI, *NeuroImage* 49 (3) (2010) 2134–2140.
82. Z.-H. Cho, et al., New brain atlas – mapping the human brain in vivo with 7.0 T MRI and comparison with postmortem histology: will these images change modern medicine?, *Int. J. Imaging Syst. Technol.* 18 (1) (2008) 2–8.

This page intentionally left blank

# Convolutional Neural Networks for Robust and Real-Time 2-D/3-D Registration

# 12

Shun Miao\*, Jane Z. Wang<sup>†</sup>, Rui Liao\*

*Siemens Medical Solutions USA, Inc., Princeton, NJ, United States\** *University of British Columbia, Vancouver, BC, Canada<sup>†</sup>*

## CHAPTER OUTLINE

<b>12.1</b>	<b>Introduction</b>	272
<b>12.2</b>	<b>X-Ray Imaging Model</b>	274
<b>12.3</b>	<b>Problem Formulation</b>	275
<b>12.4</b>	<b>Regression Strategy</b>	276
12.4.1	Parameter Space Partitioning	276
12.4.2	Marginal Space Regression	277
<b>12.5</b>	<b>Feature Extraction</b>	277
12.5.1	Local Image Residual	277
12.5.2	3-D Points of Interest	279
<b>12.6</b>	<b>Convolutional Neural Network</b>	280
12.6.1	Network Structure	280
12.6.2	Training Data	281
12.6.3	Solver	282
<b>12.7</b>	<b>Experiments and Results</b>	283
12.7.1	Experiment Setup	283
12.7.2	Hardware & Software	285
12.7.3	Performance Analysis	286
12.7.3.1	Results	287
12.7.4	Comparison with State-of-the-Art Methods	288
12.7.4.1	Evaluated Methods	289
12.7.4.2	Results	289
<b>12.8</b>	<b>Discussion</b>	292
<b>Disclaimer</b>		294
<b>References</b>		294

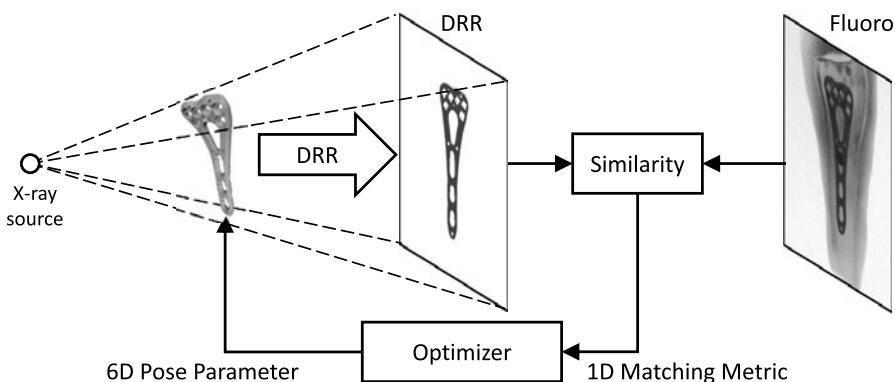
---

## 12.1 INTRODUCTION

Two-dimensional (2-D) to three-dimensional (3-D) registration represents one of the key enabling technologies in medical imaging and image-guided interventions [1]. It can bring the pre-operative 3-D data and intra-operative 2-D data into the same coordinate system, to facilitate accurate diagnosis and/or provide advanced image guidance. The pre-operative 3-D data generally includes Computed Tomography (CT), Cone-Beam CT (CBCT), Magnetic Resonance Imaging (MRI), and Computer Aided Design (CAD) model of medical devices, while the intra-operative 2-D data is dominantly X-ray images. In this paper, we focus on registering a 3-D X-ray attenuation map provided by CT or CBCT with a 2-D X-ray image in real-time. Depending on the application, other 3-D modalities (e.g., MRI and CAD model) can be converted to a 3-D X-ray attenuation map before performing 2-D/3-D registration.

Accurate 2-D/3-D registration is typically achieved by intensity-based methods, where a simulated X-ray image, referred to as Digitally Reconstructed Radiograph (DRR), is derived from the 3-D X-ray attenuation map by simulating the attenuation of virtual X-rays, and an optimizer is employed to maximize an intensity-based similarity measure between the DRR and X-ray images [2–5] (see Fig. 12.1). This is indeed an ineffective formulation because the information provided by the DRR and fluoroscopic image are compressed to the scalar-valued similarity measure, while other higher-dimensional information from the images is completely unexploited. The unexploited information includes the appearance of the mismatch, the direction of the offset, etc., which could potentially provide a clue on how the registration should be adjusted to align the two images. In addition, the matching metrics employed are often ineffective in continuously measuring the alignment of the target object in the two images, as they are typically highly non-convex (i.e., have local maxima in false positions), which makes optimization a challenging task. Due to the above shortcomings, intensity-based 2-D/3-D registration methods typically have low computational efficiency (due to iterative optimization over a 1-D metric) and small capture range (due to optimization over a highly non-convex metric).

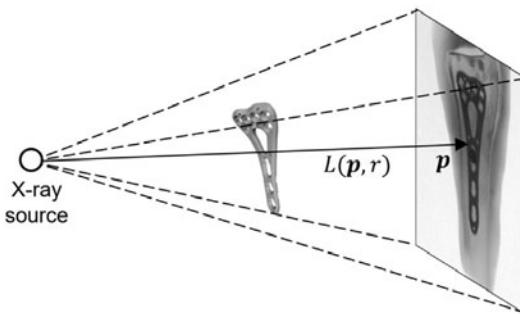
In intensity-based 2-D/3-D registration, DRRs need to be repeatedly calculated during the iterative optimization, which means heavy computation. Some efforts have been made toward accelerating DRR generation. One strategy for faster DRR generation is sparse sampling, where a subset of the pixels are statistically chosen for DRR rendering and similarity measure calculation [6,7]. However, only a few similarity measures are suitable to be calculated on a random subset of the image, e.g., Mutual Information (MI) [6] and Stochastic Rank Correlation (SRC) [7]. Another strategy is splatting, which is a voxel-based volume rendering technique that directly projects single voxels to the imaging plane [8,9]. Splatting allows us to only use voxels with intensity above certain threshold for rendering, which significantly reduces the number of voxels to be visited. However, one inherent problem of splatting is that, due to aliasing artifacts, the image quality of the generated DRR is significantly degraded compared to the DRR generated by the standard Ray Casting algorithm [10], which subsequently degrades the registration accuracy.

**FIGURE 12.1**

Formulation of intensity-based 2-D/3-D registration.

Motivated by the success of machine learning in computer vision, supervised learning has also been explored for 2-D/3-D registration. Several metric learning methods have been proposed to learn similarity measures using supervised learning [11,12]. While learned metrics could have better capture range and/or accuracy over general purpose similarity measures on specific applications or image modalities, 2-D/3-D registration methods using learned metrics still fall into the category of intensity-based methods with a high computational cost. As a new direction, several attempts have been made recently toward learning regressors to solve 2-D/3-D registration problems in real-time [13,14]. Gouveia et al. [13] extracted a handcrafted feature from the X-ray image and trained a Multi-Layer Perceptron (MLP) regressor to estimate the 3-D transformation parameters. However, the reported accuracy is much lower than can be achieved using intensity-based methods, suggesting that the handcrafted feature and MLP are unable to accurately recover the underlying complex transformation. Chou et al. [14] computed the residual between the DRR and X-ray images as a feature and trained linear regressors to estimate the transformation parameters to reduce the residual. Since the residual is a low-level feature, the mapping from it to the transformation parameters is highly nonlinear, which cannot be reliably recovered using linear regressors, as will be shown in our experiment.

In recent years, promising results on object matching for computer vision tasks have been reported using machine learning methods [15–18]. While these methods are capable of reliably recovering the object's location and/or pose for computer vision tasks, they are unable to meet the accuracy requirement of 2-D/3-D registration tasks in medical imaging, which often require a very high accuracy (i.e., sub-millimeter) for diagnosis and surgery guidance purposes. For example, Wohlhart et al. [15] proposed to train a Convolutional Neural Network (CNN) to learn a pose differentiating descriptor from range images, and use  $k$ -Nearest Neighbor for pose estimation. While global pose estimation can be achieved using this method, its ac-

**FIGURE 12.2**

X-ray imaging geometry.

curacy is relatively low, i.e., the success rate for angle error less than 5 degrees is below 60% for  $k = 1$ . Dollár et al. [16] proposed to train cascaded regressors on a pose-indexed feature that is only affected by the difference between the ground truth and initial pose parameters for pose estimation. This method aims to solve 2-D pose estimation from RGB images, but is not applicable for 2-D/3-D registration problems.

In this chapter, we propose a CNN regression-based method, referred to as Pose Estimation via Hierarchical Learning (PEHL), to achieve real-time 2-D/3-D registration with a large capture range and high accuracy. The key of our method is to train CNN regressors to recover the mapping from the DRR and X-ray images to the difference of their underlying transformation parameters. This mapping is highly complex, and training regressors to recover the mapping is far from being trivial. In the proposed method, we achieve this by first simplifying the nonlinear relationship using three algorithmic strategies and then capturing the mapping using CNN regressors with a strong nonlinear modeling capability.

## 12.2 X-RAY IMAGING MODEL

[Fig. 12.2](#) shows the geometry of X-ray imaging. Assuming that the X-ray imaging system corrects the beam divergence and the X-ray sensor has a logarithm static response, X-ray image generation can be described by the following model:

$$I(\mathbf{p}) = \int \mu(L(\mathbf{p}, r)) dr, \quad (12.1)$$

where  $I(\mathbf{p})$  is the intensity of the X-ray image at point  $\mathbf{p}$ ,  $L(\mathbf{p}, r)$  is the ray from the X-ray source to point  $\mathbf{p}$ , parameterized by  $r$ , and  $\mu(\cdot)$  is the X-ray attenuation coefficient. Denoting the X-ray attenuation map of the object to be imaged as  $J : \mathbb{R}^3 \rightarrow \mathbb{R}$ , and the 3-D transformation from the object coordinate system to the X-ray imaging coordinate system as  $T : \mathbb{R}^3 \rightarrow \mathbb{R}^3$ , the attenuation coefficient at point  $\mathbf{x}$  in

the X-ray imaging coordinate system is

$$\mu(\mathbf{x}) = J(T^{-1} \circ \mathbf{x}). \quad (12.2)$$

Combining Eq. (12.1) and Eq. (12.2), we have

$$I(\mathbf{p}) = \int J(T^{-1} \circ \mathbf{L}(\mathbf{p}, r)) dr. \quad (12.3)$$

In 2-D/3-D registration problems,  $\mathbf{L}$  is determined by the X-ray imaging system,  $J$  is provided by the 3-D data (e.g., CT intensity), and the transformation  $T$  is to be estimated from the input X-ray image  $I$ . Given  $J$ ,  $\mathbf{L}$ , and  $T$ , a synthetic X-ray image  $I(\cdot)$  (i.e., DRR) can be computed following Eq. (12.3) using the Ray-casting algorithm [10].

## 12.3 PROBLEM FORMULATION

Given the ineffectiveness of intensity-based formulation of 2-D/3-D registration, we propose a more effective regression-based formulation, where a CNN regressor takes the DRR and fluoroscopic image as input, and produces an update of the transformation parameters. Instead of converting the two images to a 1-D metric, in the regression-based formulation, we employ CNNs to learn representations from the image intensities, and reveal the mapping from the representations to the parameter updates. Using this formulation, the information in the DRR and fluoroscopic image can be fully exploited to guide the update of the transformation parameters. The regression-based formulation is mathematically described as follows.

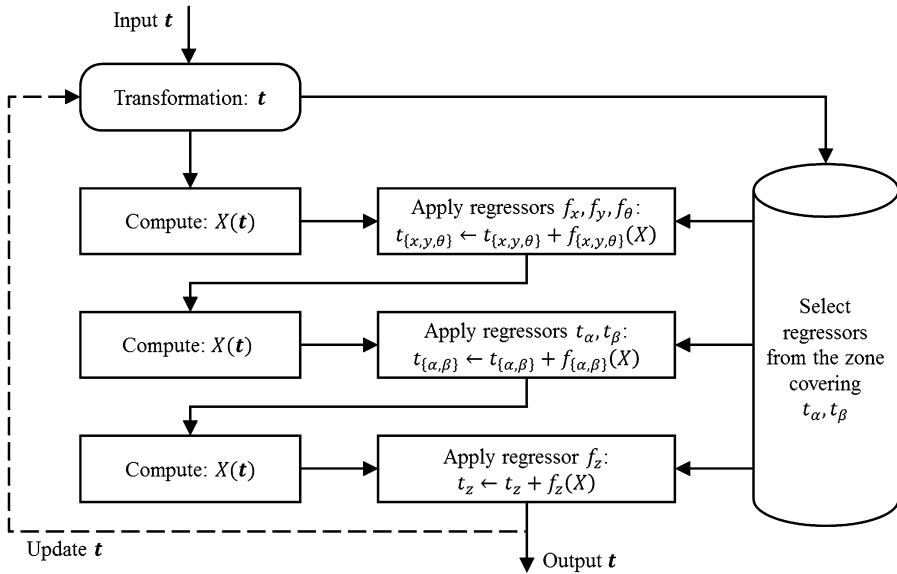
Based on Eq. (12.3), we denote the X-ray image with transformation parameters  $\mathbf{t}$  as  $I_{\mathbf{t}}$ , where the variables  $\mathbf{L}$  and  $J$  are omitted for simplicity because they are non-varying for a given 2-D/3-D registration task. The inputs for 2-D/3-D registration are: (i) a 3-D object described by its X-ray attenuation map  $J$ , (ii) an X-ray image  $I_{\mathbf{t}_{gt}}$ , where  $\mathbf{t}_{gt}$  denotes the unknown ground truth transformation parameters, and (iii) initial transformation parameters  $\mathbf{t}_{ini}$ . The 2-D/3-D registration problem is formulated as a regression problem, where a set of regressors  $f(\cdot)$  are trained to reveal the mapping from a feature  $X(\mathbf{t}_{ini}, I_{\mathbf{t}_{gt}})$  extracted from the inputs to the parameter residuals,  $\mathbf{t}_{gt} - \mathbf{t}_{ini}$ , as long as it is within a capture range  $\epsilon$ :

$$\mathbf{t}_{gt} - \mathbf{t}_{ini} \approx f(X(\mathbf{t}_{ini}, I_{\mathbf{t}_{gt}})), \quad \forall \mathbf{t}_{gt} - \mathbf{t}_{ini} \in \epsilon. \quad (12.4)$$

An estimate of  $\mathbf{t}_{gt}$  is then obtained by applying the regressors and incorporating the estimated parameter residuals into  $\mathbf{t}_{ini}$ :

$$\hat{\mathbf{t}}_{gt} = \mathbf{t}_{ini} + f(X(\mathbf{t}_{ini}, I_{\mathbf{t}_{gt}})). \quad (12.5)$$

It is worth noting that the range  $\epsilon$  in Eq. (12.4) is equivalent to the capture range of optimization-based registration methods. Based on Eq. (12.4), our problem formu-

**FIGURE 12.3**

Workflow of the proposed regression strategy.

lution can be expressed as designing a feature extractor  $X(\cdot)$  and training regressors  $f(\cdot)$  such that

$$\delta t \approx f(X(t, I_{t+\delta t})), \quad \forall \delta t \in \epsilon. \quad (12.6)$$

In the next section, we will discuss in detail (i) how the feature  $X(t, I_{t+\delta t})$  is calculated and (ii) how the regressors  $f(\cdot)$  are designed, trained, and applied.

## 12.4 REGRESSION STRATEGY

In this section, we describe the proposed regression strategy. We first partition the parameter space into multiple zones, for which the regressors are trained and applied separately. We then divide the parameters into three groups and regress them in marginal spaces. Fig. 12.3 shows the workflow of the proposed regression strategy.

### 12.4.1 PARAMETER SPACE PARTITIONING

We parameterize the transformation by 3 in-plane and 3 out-of-plane transformation parameters [19]. In particular, in-plane transformation parameters include 2 translation parameters,  $t_x$  and  $t_y$ , and 1 rotation parameter,  $t_\theta$ . The effects of in-plane transformation parameters are approximately 2-D rigid-body transformations. Out-

of-plane transformation parameters include 1 out-of-plane translation parameter,  $t_z$ , and 2 out-of-plane rotation parameters,  $t_\alpha$  and  $t_\beta$ . The effects of out-of-plane translation and rotations are scaling and shape changes, respectively.

To simplify the problem, partition the  $360 \times 360$  degrees parameter space spanned by  $t_\alpha$  and  $t_\beta$  into an  $18 \times 18$  grid (empirically selected in our experiment). Each square in the grid covers a  $20 \times 20$  degrees area, and is referred to as a *zone*. For each zone, the parameter regressors are trained separately, so that each regressor only need to solve a constrained and simplified problem. In the application stage, the regressors corresponding to the current  $t_\alpha$  and  $t_\beta$  (provided as initial registration parameters) are retrieved and applied.

### 12.4.2 MARGINAL SPACE REGRESSION

Instead of jointly regressing the 6 parameters together, they are divided into 3 groups, and regressed in the marginal space, which reduces confounding factors in the regression tasks and hence simplifies the problem. The 3 groups of parameters are defined as follows:

- *Group 1.* In-plane parameters,  $\delta t_x, \delta t_y, \delta t_\theta$
- *Group 2.* Out-of-plane rotation parameters,  $\delta t_\alpha, \delta t_\beta$
- *Group 3.* Out-of-plane translation parameter,  $\delta t_z$

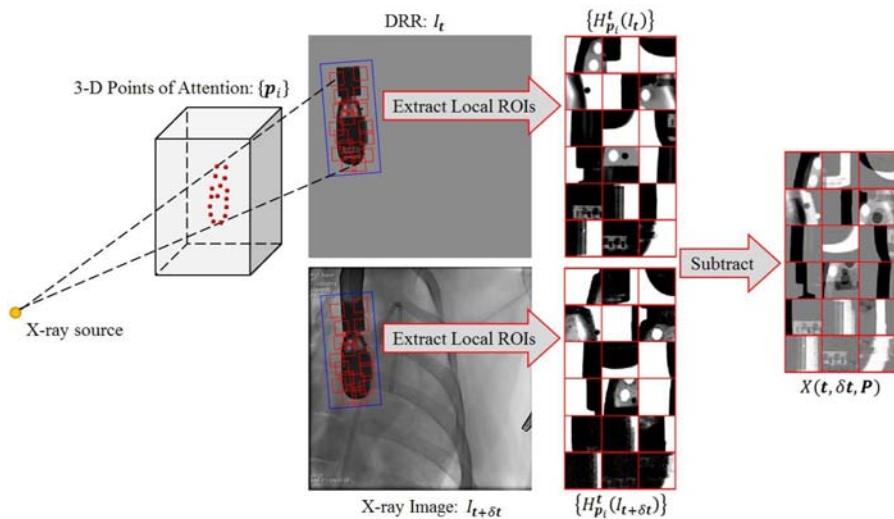
Among the 3 groups, the parameters in Group 1 are considered to be the easiest to be estimated because they cause simple while dominant rigid-body 2-D transformation of the object in the projection image that is less affected by the variations of the parameters in the other two groups. The parameter in Group 3 is the most difficult to be estimated because it only causes subtle scaling of the object in the projection image. The difficulty in estimating parameters in Group 2 falls in-between. Therefore we regress the 3 groups of parameters sequentially, from the easiest group to the most difficult. After a group of parameters are regressed, the feature  $X(t, I_{t+\delta t})$  is re-calculated using the already-estimated parameters for the regression of the parameters in the next group. This way the mapping to be regressed for each group is simplified by limiting the dimension and removing the compounding factors coming from those parameters in the previous groups.

---

## 12.5 FEATURE EXTRACTION

### 12.5.1 LOCAL IMAGE RESIDUAL

We calculate the residual between DRR rendered using transformation parameters  $t$ , denoted by  $I_t$  and the X-ray image  $I_{t+\delta t}$  in local patches of attention as the input feature for regression. To determine the locations, sizes, and orientations of the local patches of attention, a number of 3-D points of interest are extracted from the 3-D model of the target object following the steps described in Section 12.5.2. Given a point  $p$  and parameters  $t$ , a square local ROI is uniquely determined in the 2-D

**FIGURE 12.4**

Workflow of LIR feature extraction, demonstrated on X-ray Echo Fusion data. The local ROIs determined by the 3-D points  $P$  and the transformation parameters  $t$  are shown as red boxes. The blue box shows a large ROI that covers the entire object, used in compared methods as will be discussed in Section 12.7.3.

imaging plane, which can be described by a triplet,  $(q, w, \phi)$ , denoting the ROI's center, width, and orientation, respectively. The center  $q$  is the 2-D projection of  $p$  using transformation parameters  $t$ . The width  $w = w_0 \cdot D/t_z$ , where  $w_0$  is the size of the ROI in mm and  $D$  is the distance between the X-ray source and detector. The orientation  $\phi = t_\theta$ , so that it is always aligned with the object. We define an operator  $H_p^t(\cdot)$  that extracts the image patch in the ROI determined by  $p$  and  $t$ , and re-sample it to a fixed size ( $52 \times 52$  in our applications). Given  $N$  3-D points,  $P = \{p_1, \dots, p_N\}$ , the LIR feature is then computed as

$$X(t, I_{t+\delta t}, P) = \{H_{p_i}^t(I_t) - H_{p_i}^t(I_{t+\delta t})\}_{i=1, \dots, N}. \quad (12.7)$$

In a local area of  $I_t$ , the effect of varying  $t_\alpha$  and  $t_\beta$  within a zone is approximately a 2-D translation. Therefore, by extracting local patches from ROIs selected based on  $t$ , the effects of all 6 transformation parameters in  $t$  are compensated, making  $H_p^t(I_t)$  approximately invariant to  $t$ . Since the difference between  $H_p^t(I_{t+\delta t})$  and  $H_p^t(I_t)$  is merely additional 2-D transformation caused by  $\delta t$ ,  $H_p^t(I_{t+\delta t})$  is also approximately invariant to  $t$ . The workflow of LIR feature extraction is shown in Fig. 12.4.

### 12.5.2 3-D POINTS OF INTEREST

The 3-D points of interest used for calculating the LIR feature are extracted separately for each zone in two steps. First, 3-D points that correspond to 2-D edges are extracted as candidates. Specifically, the candidates are extracted by thresholding pixels with high gradient magnitudes in a synthetic X-ray image (i.e., generated using DRR) with  $t_\alpha$  and  $t_\beta$  at the center of the zone, and then back-projecting them to the corresponding 3-D structures. The formation model of gradients in X-ray images has been shown in [20] as

$$g(\mathbf{p}) = \int \eta(\mathbf{L}(\mathbf{p}, r)) dr, \quad (12.8)$$

where  $g(\mathbf{p})$  is the magnitude of the X-ray image gradient at the point  $\mathbf{p}$ , and  $\eta(\cdot)$  can be computed from  $\mu(\cdot)$  and the X-ray perspective geometry [20]. We back-project  $\mathbf{p}$  to  $\mathbf{L}(\mathbf{p}, r_0)$ , where

$$r_0 = \arg \max_r \mathbf{L}(\mathbf{p}, r), \quad (12.9)$$

if

$$\int_{r_0-\sigma}^{r_0+\sigma} \eta(\mathbf{L}(\mathbf{p}, r)) dr \geq 0.9 \cdot g(\mathbf{p}). \quad (12.10)$$

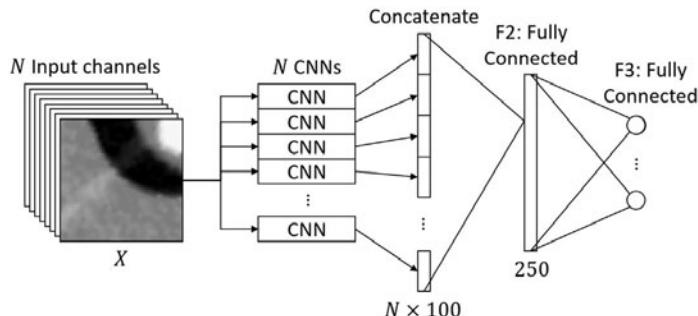
The condition in Eq. (12.10) ensures that the 3-D structure around  $\mathbf{L}(\mathbf{p}, r_0)$  “essentially generates” the 2-D gradient  $g(\mathbf{p})$ , because the contribution of  $\eta(\cdot)$  within a small neighborhood (i.e.,  $\sigma = 2$  mm) of  $\mathbf{L}(\mathbf{p}, r_0)$  leads to the majority (i.e.,  $\geq 90\%$ ) of the magnitude of  $g(\mathbf{p})$ . In other words, we find the dominant 3-D structure corresponding to the gradient in the X-ray image.

Second, the candidates are filtered so that only those leading to the most discriminative LIRs are kept. To achieve this, we randomly generate  $\{\mathbf{t}_j\}_{j=1}^M$  with  $t_\alpha$  and  $t_\beta$  within the zone and  $\{\delta t_k\}_{k=1}^M$  within the capture range  $\epsilon$  ( $M = 1000$  in our applications). The intensity of the  $n$ th pixel of  $H_{\mathbf{p}_i}^{t_j}(I_{t_j}) - H_{\mathbf{p}_i}^{t_j}(I_{t_j + \delta t_k})$  is denoted as  $h_{n,i,j,k}$ . The following two measurements are computed for all candidates:

$$E_i = \left\langle (h_{n,i,j,k} - \langle h_{n,i,j,k} \rangle_k)^2 \right\rangle_{n,j,k}, \quad (12.11)$$

$$F_i = \left\langle (h_{n,i,j,k} - \langle h_{n,i,j,k} \rangle_k)^2 \right\rangle_{n,j,k}, \quad (12.12)$$

where  $\langle \cdot \rangle$  is an average operator with respect to all indexes in the subscript. Since  $E_i$  and  $F_i$  measure the sensitivity of  $H_{\mathbf{p}_i}^t(I_t) - H_{\mathbf{p}_i}^t(I_{t+\delta t})$  with respect to  $t$  and  $\delta t$ , respectively, an ideal LIR should have a small  $E_i$  so that it is less affected by  $t$  and a large  $F_i$  for regressing  $\delta t$ . Therefore, the candidate list is filtered by picking the candidate with the largest  $F_i/E_i$  in the list, and then removing other candidates with ROIs that have more than 25% overlapping area. This process repeats until the list is empty.

**FIGURE 12.5**

Structure of the CNN regression model.

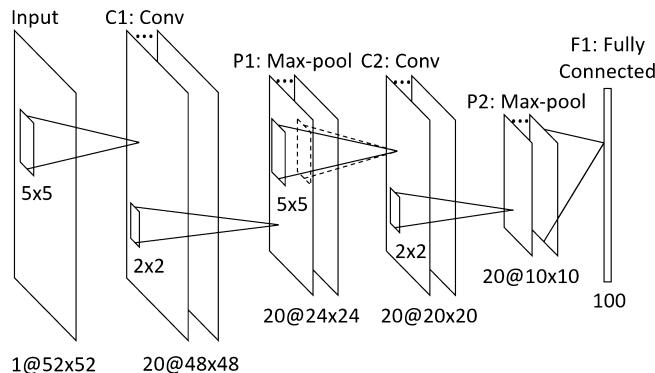
## 12.6 CONVOLUTIONAL NEURAL NETWORK

In our regression problem, there are two challenges in designing the CNN regression model: (i) it needs to be flexible enough to capture the complex mapping from  $X(t, I_{t+\delta t})$  to  $\delta t$ , and (ii) it needs to be light-weighted enough to be forwarded in real-time and stored in Random-Access Memory (RAM). Managing memory footprint is particularly important because regressors for all zones (in total 324) need to be loaded to RAM for optimal speed. Another challenge is the training of CNN requires a large number of labeled data, which is often difficult to obtain for many clinical applications. We employ the following CNN regression model and training procedures to address the above challenges.

### 12.6.1 NETWORK STRUCTURE

A CNN regression model with the architecture shown in Fig. 12.5 is trained for each group in each zone. According to Eq. (12.7), the input of the regression model consists of  $N$  channels, corresponding to  $N$  LIRs. The CNN shown in Fig. 12.6 is applied on each channel for feature extraction. The CNN consists of five layers, including two  $5 \times 5$  convolutional layers (C1 and C2), each followed by a  $2 \times 2$  max-pooling layers (P1 and P2) with stride 2, and a fully-connected layer (F1) with 100 Rectified Linear Unit (ReLU) activations neurons. The feature vectors extracted from all input channels are then concatenated and connected to another fully-connected layer (F2) with 250 ReLU activations neurons. The output layer (F3) is fully-connected to F2, with each output node corresponding to one parameter in the group. Since all input channels are LIR, it is reasonable to extract feature from them using the same CNN. Therefore, the feature extraction CNNs in Fig. 12.5 share weights.

In our experiment, we empirically selected the size of the ROI, which led to  $N \approx 18$ . Using the CNN model shown in Fig. 12.5 with weight sharing, there are 660,500 weights in total for each group in each zone, excluding the output layer, which only has  $250 \times N_t$  weights, where  $N_t$  is the number of parameters in the group.

**FIGURE 12.6**

Structure of the CNN applied for each input channel.

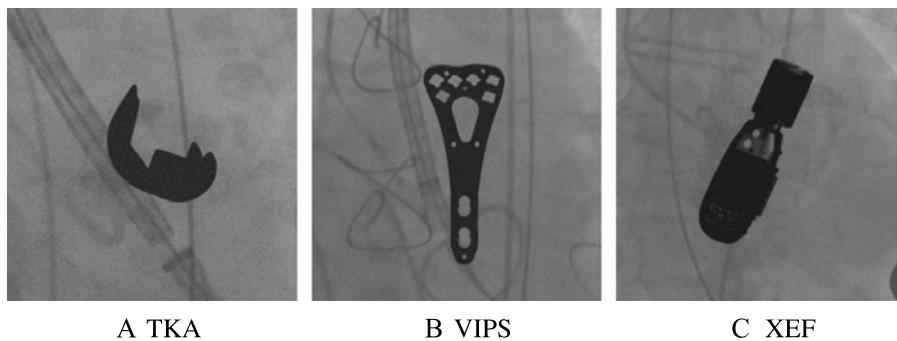
If the weights are stored as 32-bit float, around 2.5 MB is required for each group in each zone. Given 3 groups and 324 zones, there are 972 CNN regression models in total, and pre-loading all of them into RAM requires 2.39 GB, which is manageable for modern computers.

## 12.6.2 TRAINING DATA

The CNN regression models are trained exclusively on synthetic X-ray images, because they provide reliable ground truth labels without the need of laborious manual annotation, and the amount of real X-ray images could be limited. The synthetic X-ray images were generated by blending a DRR of the object with a background from real X-ray images:

$$I = I_{Xray} + \gamma \cdot G_\sigma * I_{DRR} + \mathcal{N}(a, b), \quad (12.13)$$

where  $I_{Xray}$  is the real X-ray image,  $I_{DRR}$  is the DRR,  $G_\sigma$  denotes a Gaussian smoothing kernel with a standard deviation  $\sigma$  simulating X-ray scattering effect,  $f * g$  denotes the convolution of  $f$  and  $g$ ,  $\gamma$  is the blending factor, and  $\mathcal{N}(a, b)$  is a random noise uniformly distributed on  $[a, b]$ . The parameters  $(\gamma, \sigma, a, b)$  were empirically tuned for each object (i.e., implants and TEE probe) to make the appearance of the synthetic X-ray image realistic. These parameters were also randomly perturbed within a neighborhood for each synthetic X-ray image to increase the variation of the appearance of the synthetic X-ray images, so that the regressors trained on them can be generalized well on real X-ray images. The background image used for a given synthetic image was randomly picked from a group of real X-ray images irrespective of the underlying clinical procedures so that the trained network would not be over-fitted for any specific type of background, which could vary sig-

**FIGURE 12.7**

Example synthetic X-ray images used for training.

**Table 12.1** Distributions of randomly generated  $\delta t$ .  $\mathcal{U}(a, b)$  denotes the uniform distribution between  $a$  and  $b$ . The units for translation and rotations are mm and degree, respectively

Group 1	Group 2	Group 3
$\delta t_x \sim \mathcal{U}(-1.5, 1.5)$	$\delta t_x \sim \mathcal{U}(-0.2, 0.2)$	$\delta t_x \sim \mathcal{U}(-0.15, 0.15)$
$\delta t_y \sim \mathcal{U}(-1.5, 1.5)$	$\delta t_y \sim \mathcal{U}(-0.2, 0.2)$	$\delta t_y \sim \mathcal{U}(-0.15, 0.15)$
$\delta t_z \sim \mathcal{U}(-15, 15)$	$\delta t_z \sim \mathcal{U}(-15, 15)$	$\delta t_z \sim \mathcal{U}(-15, 15)$
$\delta t_\theta \sim \mathcal{U}(-3, 3)$	$\delta t_\theta \sim \mathcal{U}(-0.5, 0.5)$	$\delta t_\theta \sim \mathcal{U}(-0.5, 0.5)$
$\delta t_\alpha \sim \mathcal{U}(-15, 15)$	$\delta t_\alpha \sim \mathcal{U}(-15, 15)$	$\delta t_\alpha \sim \mathcal{U}(-0.75, 0.75)$
$\delta t_\beta \sim \mathcal{U}(-15, 15)$	$\delta t_\beta \sim \mathcal{U}(-15, 15)$	$\delta t_\beta \sim \mathcal{U}(-0.75, 0.75)$

nificantly from case to case clinically. Examples of synthetic X-ray images are shown in Fig. 12.7.

For each group in each zone, we randomly generate 25,000 pairs of  $t$  and  $\delta t$ . The parameters  $t$  follow a uniform distribution with  $t_\alpha$  and  $t_\beta$  constrained in the zone. The parameter errors  $\delta t$  also follow a uniform distribution, while 3 different distribution ranges are used for the 3 groups, as shown in Table 12.1. The distribution ranges of  $\delta t$  for Group 1 are the target capture range that the regressors are designed for. The distribution ranges of  $\delta t_x$ ,  $\delta t_y$ , and  $\delta t_\theta$  are reduced for Group 2, because they are reduced by applying the regressors in the first group. For the same reason, the distribution ranges of  $\delta t_\alpha$  and  $t_\beta$  are reduced for Group 3. For each pair of  $t$  and  $\delta t$ , a synthetic X-ray image  $I_{t+\delta t}$  is generated, and the feature  $X(t, I_{t+\delta t})$  is calculated following Eq. (12.7).

### 12.6.3 SOLVER

The objective function to be minimized during the training is Euclidean loss, defined as

$$\Phi = \frac{1}{K} \sum_{i=1}^K \|y_i - f(X_i; \mathbf{W})\|_2^2, \quad (12.14)$$

where  $K$  is the number of training samples,  $y_i$  is the label for the  $i$ th training sample,  $\mathbf{W}$  is a vector of weights to be learned,  $f(X_i; \mathbf{W})$  is the output of the regression model parameterized by  $\mathbf{W}$  on the  $i$ th training sample. The weights  $\mathbf{W}$  are learned using Stochastic Gradient Descent (SGD) [21], with a batch size of 64, momentum of  $m = 0.9$ , and weight decay of  $d = 0.0001$ . The update rule for  $\mathbf{W}$  is:

$$\mathbf{V}_{i+1} := m \cdot \mathbf{V}_i - d \cdot \kappa_i \cdot \mathbf{W}_i - \kappa_i \cdot \left\langle \frac{\partial \Phi}{\partial \mathbf{W}} \Big|_{\mathbf{W}_i} \right\rangle_{D_i}, \quad (12.15)$$

$$\mathbf{W}_{i+1} := \mathbf{W}_i + \mathbf{V}_{i+1}, \quad (12.16)$$

where  $i$  is the iteration index,  $\mathbf{V}$  is the momentum variable,  $\kappa_i$  is the learning rate at the  $i$ th iteration, and  $\left\langle \frac{\partial \Phi}{\partial \mathbf{W}} \Big|_{\mathbf{W}_i} \right\rangle_{D_i}$  is the derivative of the objective function computed on the  $i$ th batch  $D_i$  with respect to  $\mathbf{W}$ , evaluated at  $\mathbf{W}_i$ . The learning rate  $\kappa_i$  is decayed in each iteration following

$$\kappa_i = 0.0025 \cdot (1 + 0.0001 \cdot i)^{-0.75}. \quad (12.17)$$

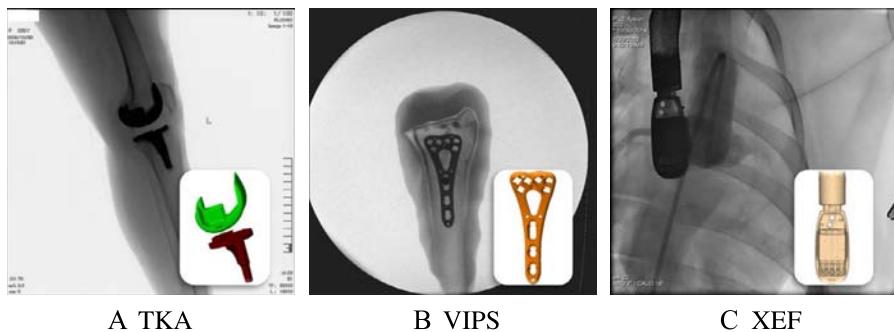
The derivative  $\frac{\partial \Phi}{\partial \mathbf{W}}$  is calculated using back-propagation. For weights shared in multiple paths, their derivatives in all paths are back-propagated separately and summed up for the weight update. The weights are initialized using the Xavier method [22], and mini-batch SGD is performed for 12,500 iterations (32 epochs).

## 12.7 EXPERIMENTS AND RESULTS

### 12.7.1 EXPERIMENT SETUP

We conducted experiments on datasets from the following 3 clinical applications:

1. *Total Knee Arthroplasty (TKA) kinematics.* In the study of the kinematics of TKA, 3-D kinematics of knee prosthesis can be estimated by matching the 3-D model of the knee prosthesis with the fluoroscopic video of the prosthesis using 2-D/3-D registration [23]. We evaluated PEHL on a fluoroscopic video consisting of 100 X-ray images of a patient's knee joint taken at the phases from full extension to maximum flexion after TKA. The size of the X-ray images is  $1024 \times 1024$  with a pixel spacing of 0.36 mm. A 3-D surface model of the prosthesis was acquired by a laser scanner, and was converted to a binary volume for registration.
2. *Virtual Implant Planning System (VIPS).* VIPS is an intraoperative application that was established to facilitate the planning of implant placement in terms of orientation, angulation and length of the screws [24]. In VIPS, 2-D/3-D registration is performed to match the 3-D virtual implant with the fluoroscopic image of

**FIGURE 12.8**

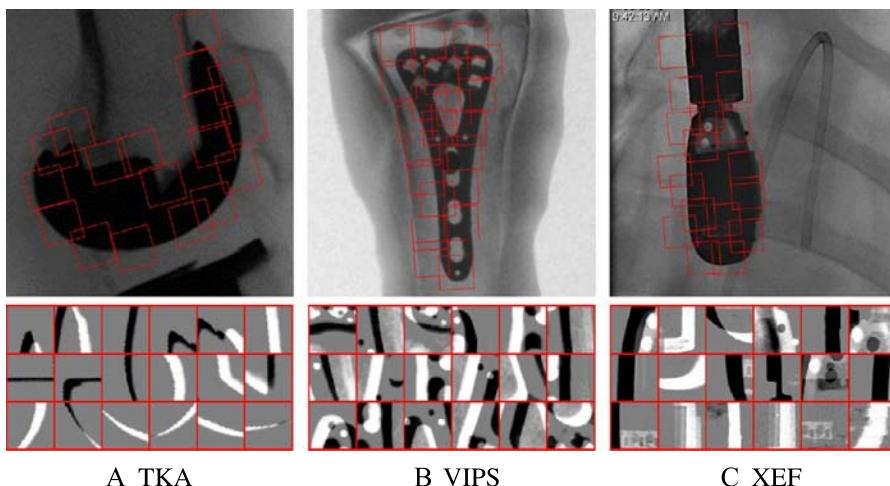
Example data, including a 3-D model and a 2-D X-ray image of the object.

the real implant. We evaluated PEHL on 7 X-ray images of a volar plate implant mounted onto a phantom model of the distal radius. The size of the X-ray images is  $1024 \times 1024$  with a pixel spacing of 0.223 mm. A 3-D CAD model of the volar plate was converted to a binary volume for registration.

3. *X-ray Echo Fusion (XEF)*. 2-D/3-D registration can be applied to estimate the 3-D pose of a transesophageal echocardiography (TEE) probe from X-ray images, which brings the X-ray and TEE images into the same coordinate system and enables the fusion of the two modalities [25]. We evaluated PEHL on 2 fluoroscopic videos with in total 94 X-ray images acquired during an animal study using a Siemens Artis Zeego C-Arm system. The size of the X-ray images is  $1024 \times 1024$  with a pixel spacing of 0.154 mm. A micro-CT scan of the Siemens TEE probe was used for registration.

Example datasets of the above 3 clinical applications are shown in Fig. 12.8. Examples of local ROIs and LIRs extracted from the 3 datasets are also shown in Fig. 12.9.

Ground truth transformation parameters used for quantifying registration error were generated by first manually registering the target object and then applying an intensity-based 2-D/3-D registration method using Powell's method combined with Gradient Correlation (GC) [9]. Perturbations of the ground truth were then generated as initial transformation parameters for 2-D/3-D registration. For TKA and XEF, 10 perturbations were generated for each X-ray image, leading to 1000 and 940 test cases, respectively. Since the number of X-ray images for VIPS is limited (i.e., 7), 140 perturbations were generated for each X-ray image to create 980 test cases. The perturbation for each parameter followed the normal distribution with a standard deviation equal to 2/3 of the training range of the same parameter (i.e., Group 1 in Table 12.1). In particular, the standard deviations for  $(t_x, t_y, t_z, t_\theta, t_\alpha, t_\beta)$  are 1 mm, 1 mm, 10 mm, 2 degrees, 10 degrees, 10 degrees, respectively. With this distribution, 42.18% of the perturbations have all 6 parameters within the training range, while the other 57.82% have at least one parameter outside of the training range.

**FIGURE 12.9**

Examples of local ROIs and LIRs.

The registration accuracy was assessed with the mean Target Registration Error in the projection direction (mTREproj) [26], calculated at the 8 corners of the bounding box of the target object. We regard mTREproj less than 1% of the size of the target object (i.e., diagonal of the bounding box) as a successful registration. For TKA, VIPS, and XEF, the sizes of the target objects are 110, 61, and 37 mm, respectively. Therefore, the success criterion for the three applications was set to mTREproj less than 1.10, 0.61, and 0.37 mm, which is equivalent to 2.8, 3.7, and 3.5 pixels on the X-ray image, respectively. Success rate was defined as the percentage of successful registrations. Capture range was defined as the initial mTREproj for which 95% of the registrations were successful [26]. Capture range is only reported for experiments where there are more than 20 samples within the capture range.

### 12.7.2 HARDWARE & SOFTWARE

The experiments were conducted on a workstation with Intel Core i7-4790k CPU, 16 GB RAM and Nvidia GeForce GTX 980 GPU. For intensity-based methods, the most computationally intensive component, DRR renderer, was implemented using the Ray-casting algorithm with hardware-accelerated 3-D texture lookups on GPU. Similarity measures were implemented in C++ and executed in a single CPU core. Both DRRs and similarity measures were only calculated within an ROI surrounding the target object, for better computational efficiency. In particular, ROIs of size  $256 \times 256$ ,  $512 \times 512$ , and  $400 \times 400$  were used for TKA, VIPS, and XEF, respectively. For PEHL, the neural network was implemented with cuDNN acceleration using an open-source deep learning framework, Caffe [27].

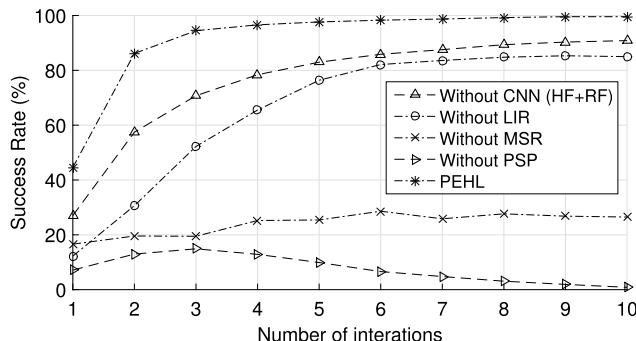
**FIGURE 12.10**

HAAR features used in the experiment “Without CNN”.

### 12.7.3 PERFORMANCE ANALYSIS

We first conducted the following experiments for detailed analysis of the performance and property of PEHL. The dataset from XEF was used for the demonstration of performance analysis because the structure of the TEE probe is more complex than the implants in TKA and VIPS, leading to an increased difficulty for an accurate registration. As described in Section 12.4.2, PEHL can be applied for multiple iterations. We demonstrate the impact of the number of iterations on performance, by applying PEHL for 10 iterations and showing the registration success rate after each iteration. We also demonstrate the importance of the individual core components of PEHL, i.e., the CNN regression model and 3 algorithmic strategies, Local Image Residual (LIR), Marginal Space Regression (MSR), and Parameter Space Partitioning (PSP), by disabling them and demonstrating the detrimental effects on performance. The following 4 scenarios were evaluated for 10 iterations to compare with PEHL:

- **Without CNN.** We implemented a companion algorithm using HAAR feature with Regression Forest as an alternative to the proposed CNN regression model. We extract 8 HAAR features as shown in Fig. 12.10 from the same training data used for training the CNNs. We mainly used edge and line features because  $\delta t$  largely corresponds to lines and edges in LIR. On these HAAR features, we trained a Regression Forest with 500 trees.
- **Without LIR.** A global image residual covering the whole object was used as the input for regression (shown in Fig. 12.4 as blue boxes). The CNN regression model was adapted accordingly. It has five hidden layers: two  $5 \times 5$  convolutional layers, each followed by a  $3 \times 3$  max-pooling layer with stride 3, and a fully-connected layer with 250 ReLU activation neurons. For each group in each zone, the network was trained on the same dataset used for training PEHL.
- **Without MSR.** The proposed CNN regression model shown in Fig. 12.6 was employed, but the output layer has 6 nodes, corresponding to the 6 parameters for rigid-body transformation. For each zone, the network was trained on the dataset used for training PEHL for Group 1.
- **Without PSP.** For each group of parameters, one CNN regression model was applied for the whole parameter space. Because LIP cannot be applied without PSP, the CNN regression model described in “without LIR” was employed in this scenario. The network was trained on 500,000 synthetic training samples with  $t_\alpha$  and  $t_\beta$  uniformly distributed in the parameter space.

**FIGURE 12.11**

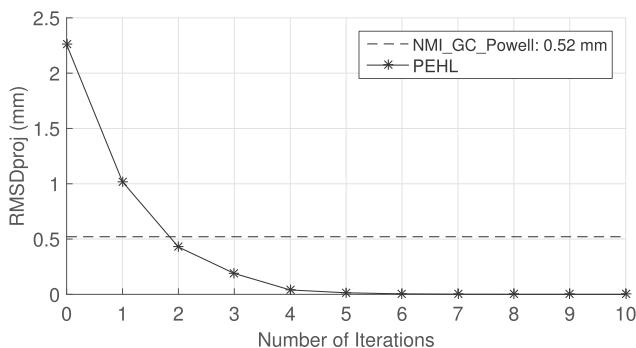
Success rates of PEHL with 1 to 10 iterations. Four individual core components of PEHL, i.e., CNN, LIR, MSR, and PSP, were disabled one at a time to demonstrate their detrimental effects on performance. Harr feature (HR) + Regression Forest (RF) was implemented to show the effect on performance without CNN. These results were generated on the XEF dataset.

We also conducted an experiment to analyze the precision of PEHL (i.e., the ability to generate consistent results starting from different initial parameters). To measure the precision, we randomly selected an X-ray image from the XEF dataset, and generated 100 perturbations of the ground truth following the same distribution described in Section 12.7.1. PEHL and the best performed intensity-based method, MI\_GC\_Powell (which will be detailed in Section 12.7.4), were applied starting from the 100 perturbations. The precision of the registration method was then quantified by the root mean squared distance in the projection direction (RMSDproj) from the registered locations of each target to their centroid. Smaller RMSDproj indicates higher precision.

### 12.7.3.1 Results

Fig. 12.11 shows the success rate as the number of iterations increases from 1 to 10 for the five analyzed scenarios. The results show that the success rate of PEHL increased rapidly in the first 3 iterations (i.e., from 44.6% to 94.8%), and kept rising slowly afterward until 9 iterations (i.e., to 99.6%). The computation time of PEHL is linear to the number of iterations, i.e., each iteration takes  $\sim 34$  ms. Therefore, applying PEHL for 3 iterations is the optimal setting for the trade-off between accuracy and efficiency, which achieves close to the optimal success rate and a real-time registration of  $\sim 10$  frames per second (fps). Therefore, in the rest of the experiment, PEHL was tested with 3 iterations unless stated otherwise.

The results show that the 3 proposed strategies, LIR, MSR, and PSP, and the use of CNN all noticeably contributed to the final registration accuracy of PEHL. In particular, if the CNN regression model is replaced with HAAR feature + Regression Forest, the success rate at the third iteration dropped to 70.7%, indicating that the

**FIGURE 12.12**

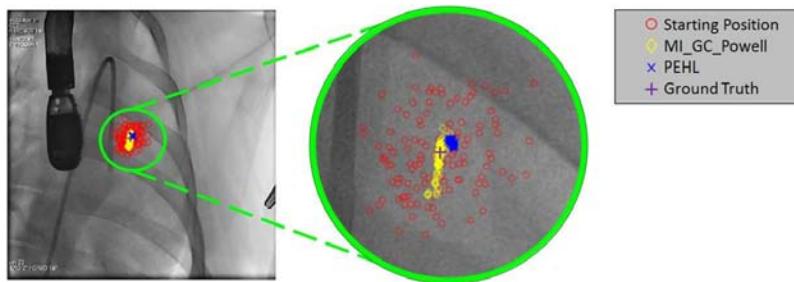
RMSDproj from the registered locations of each target to their centroid using MI\_GC\_Powell and PEHL with 1 to 10 iterations. At Number of Iterations = 0, the RMSEproj at the perturbed positions without registration is shown. These results were generated on the XEF dataset.

strong nonlinear modeling capability of CNN is critical to the success of PEHL. If the LIP is replaced with a global image residual, the success rate at the third iteration dropped significantly to 52.2%, showing that LIR is a necessary component to simplify the target mapping so that it can be robustly regressed with the desired accuracy. When MSR and PSP are disabled, the system almost completely failed, dropping the success rate at the third iteration to 19.5% and 14.9%, respectively, suggesting that MSR and PSP are key components that make the regression problem solvable using the proposed CNN regression model.

Fig. 12.12 shows the RMSDproj from registered target points to their corresponding centroid using both MI\_GC\_Powell and PEHL with 1 to 10 iterations. The results show that as the number of iteration increases, the RMSDproj of PEHL approaches zero, indicating that with sufficient number of iterations, PEHL can reliably reproduce the same result starting from different positions (e.g., 6 iterations lead to RMSEproj = 0.005 mm). At the third iteration, the RMSDproj of PEHL is 0.198 mm, which is 62% smaller than that of MI\_GC\_Powell, i.e., 0.52 mm. In Fig. 12.13, we show the distribution of a landmark at the center of the imaging cone of the TEE probe before registration, after registration using MI\_GC\_Powell and after 3 iterations of PEHL. Both the RMSDproj result and the distribution of the landmarks suggest that PEHL has a significant advantage over MI\_GCPowell in terms of precision.

#### 12.7.4 COMPARISON WITH STATE-OF-THE-ART METHODS

We also conducted experiments to compare PEHL with state-of-the-art 2-D/3-D registration methods, including several variations of intensity-based methods and a recent linear regression-based method.

**FIGURE 12.13**

Distribution of projections of a landmark after registration using MI\_GC\_Powell and PEHL.

#### 12.7.4.1 *Evaluated Methods*

We first evaluated intensity-based 2-D/3-D registration methods, where we have multiple options optimizer and similarity measure. We used Powell's method as the optimizer, as it is shown to be the most effective optimizer for intensity-based 2-D/3-D registration in a recent study [28]. We evaluated three popular similarity measures, MI, Cross-Correlation (CC), and GC, which have also been reported to be effective in recent literature [3,4,9]. The above three intensity-based methods are referred to as *MI\_Powell*, *CC\_Powell*, and *GC\_Powell*, indicating the adopted similarity measure and optimization method. We also implemented another intensity-based method combining MI and GC to achieve improved robustness and accuracy for comparison. MI focuses on the match of the histograms at the global scale, which leads to a relatively large capture range, but lacks fine accuracy. GC focuses on matching image gradients, which leads to high registration accuracy, but limits the capture range. The combined method, referred to as *MI\_GC\_Powell*, first applies *MI\_Powell* to bring the registration into the capture range of GC, and then applies *GC\_Powell* to refine the registration.

We also evaluated CLARET, a linear regression-based 2-D/3-D registration method introduced in [14], which is closely related to PEHL, as it iteratively applies regressors on the image residual to estimate the transformation parameters. In [14], the linear regressors were reported to be trained on X-ray images with fixed ground truth transformation parameters, and therefore can only be applied on X-ray images with poses within a limited range. Since the input X-ray images used in our experiment do not have such limitation, we applied the PSP strategy to train linear regressors separately for each zone. For each zone, the linear regressor was trained on the dataset used for training PEHL for Group 1.

#### 12.7.4.2 *Results*

We first observed that the linear regressor in CLARET completely failed in our experiment setup. Table 12.2 shows the root mean squared error (RMSE) of the 6 parameters yielded by PEHL and CLARET on the synthetic training data for XEF.

**Table 12.2** RMSE of the 6 transformation parameters yielded by PEHL and CLARET on the training data for XEF

	$t_x$ (mm)	$t_y$ (mm)	$t_z$ (mm)	$t_\theta$ (°)	$t_\alpha$ (°)	$t_\beta$ (°)
Start	0.86	0.86	8.65	1.71	8.66	8.66
PEHL	0.04	0.04	0.32	0.06	0.18	0.18
CLARET	0.51	0.88	34.85	2.00	19.41	17.52

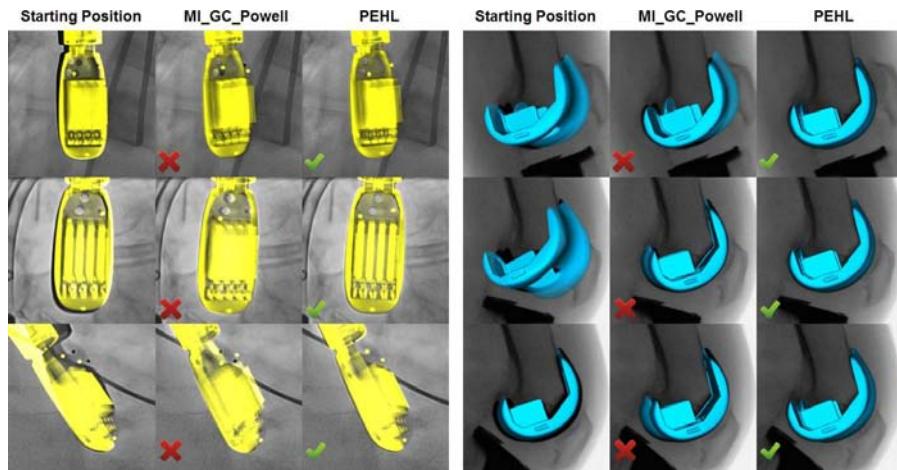
The linear regression resulted in very large errors on the training data (i.e., larger than the perturbation), indicating that the mapping from the global image residual to the underlying transformation parameters is highly nonlinear, and therefore cannot be reliably captured by a linear regressor. In comparison, PEHL employs the 3 algorithmic strategies to simplify the nonlinear relationship and captures it using a CNN with a strong nonlinear modeling capability. As a result, PEHL resulted in a very small error on the synthetic training data. Its ability to generalize the performance to unseen testing data was then assessed on real data from the three clinical applications.

Table 12.3 summarizes the success rate, capture range, percentiles of mTREproj and average running time per registration for PEHL and four intensity-based methods on the three applications. The results show that the four intensity-based methods, MI\_Powell, GC\_Powell, CC\_Powell, and MI\_GC\_Powell, all resulted in relatively small capture ranges and slow speeds that are incapable for real-time registration. The small capture range is due to the limitation of non-convex optimization. Because the intensity-based similarity measures are highly non-convex, the optimizer is likely to get trapped in local maxima if the starting position is not close enough to the global maxima. The relatively slow speed is due to the large number of DRR renderings and similarity measure calculations during the optimization. The fastest intensity-based method is CC\_Powell, which took 0.4–0.9 s per registration, is still significantly slower than typical fluoroscopic frame rates (i.e., 10–15 fps). The success rates for MI\_Powell, GC\_Powell, and CC\_Powell are also very low, mainly due to two different reasons: (i) MI and CC are unable to resolve a small mismatch; (ii) GC is unable to recover a large mismatch. By employing MI\_Powell to recover a large mismatch and GC\_Powell to resolve a small mismatch, MI\_GC\_Powell achieved much higher success rates than using MI or GC alone.

The results show that PEHL achieved the best success rate and capture range among all evaluated methods on all three applications, and is capable for real-time registration. The advantage of PEHL in capture range compared to the second best-performed method, i.e., MI\_GC\_Powell, is significant. In particular, on the three applications, PEHL resulted in 155% (on TKA), 99% (on VIPS), and 306% (on XEF) larger capture range than MI\_GC\_Powell, respectively. The success rates of PEHL are also higher than that of MI\_GC\_Powell by 27.8% (on TKA), 5% (on VIPS), and 5.4% (on XEF). The advantage of PEHL in capture range and robustness is primarily due to the learning of the direct mapping from the LIR to the residual of the transformation parameters, which eliminates the need of optimizing over a highly non-convex similarity measure. PEHL resulted in a running time of ~0.1 s per registration for all

**Table 12.3** Quantitative experiment results of PEHL and baseline methods. Success rate is the percentage of successful registrations in each experiment. Capture range is the initial mTREproj for which 95% of the registrations were successful. The 10th, 25th, 50th, 75th, and 90th percentiles of mTREproj are reported. Running time records the average and standard deviation of the computation time for each registration computed in each experiment. Capture range is only reported for experiments where there are more than 20 samples within the capture range

Application	Method	SR		CR (mm)		mTREproj percentile (mm)					RT (s)
		10th	25th	50th	75th	90th					
TKA	Start	N/A	N/A	3.29	4.63	6.98	10.1	12.7	12.7	N/A	
	MI_Powell	36.2%	N/A	0.44	0.75	1.69	6.24	8.42	8.42	1.37 ± 0.44	
	CC_Powell	43.8%	1.88	0.35	0.64	1.36	6.32	8.40	8.40	0.92 ± 0.27	
	GC_Powell	45.2%	2.14	0.33	0.59	1.31	7.64	9.77	9.77	2.52 ± 1.22	
	MI_GC_Powell	51.8%	2.83	<b>0.30</b>	0.52	1.05	6.41	8.61	8.61	3.11 ± 0.94	
	PEHL	<b>79.6%</b>	<b>7.23</b>	0.33	<b>0.44</b>	<b>0.59</b>	<b>0.90</b>	<b>6.73</b>	<b>6.73</b>	<b>0.11 ± 0.00</b>	
VIPS	Start	N/A	N/A	1.18	1.52	2.00	2.59	3.10	3.10	N/A	
	MI_Powell	75.1%	N/A	0.16	0.23	0.38	0.60	0.92	0.92	1.66 ± 0.60	
	CC_Powell	57.7%	0.89	0.19	0.30	0.54	0.85	1.29	1.29	0.91 ± 0.31	
	GC_Powell	78.7%	1.12	0.12	0.21	0.33	0.54	2.28	2.28	3.91 ± 1.55	
	MI_GC_Powell	92.7%	2.77	<b>0.11</b>	<b>0.17</b>	0.26	<b>0.37</b>	0.54	0.54	4.71 ± 1.59	
	PEHL	<b>99.7%</b>	<b>5.51</b>	0.15	0.18	<b>0.24</b>	0.39	<b>0.45</b>	<b>0.45</b>	<b>0.10 ± 0.00</b>	
XEF	Start	N/A	N/A	1.05	1.37	1.83	2.31	2.79	2.79	N/A	
	MI_Powell	69.7%	N/A	0.17	0.21	0.28	0.40	0.60	0.60	0.79 ± 0.29	
	CC_Powell	54.8%	N/A	0.12	0.17	0.32	0.89	1.17	1.17	0.40 ± 0.10	
	GC_Powell	56.9%	N/A	0.07	0.14	0.28	1.06	3.15	3.15	2.06 ± 1.05	
	MI_GC_Powell	89.1%	0.84	<b>0.05</b>	<b>0.10</b>	0.17	0.27	0.38	0.38	2.03 ± 0.69	
	PEHL	<b>94.5%</b>	<b>3.33</b>	0.08	0.11	<b>0.15</b>	<b>0.20</b>	<b>0.24</b>	<b>0.24</b>	<b>0.10 ± 0.00</b>	

**FIGURE 12.14**

Visual examples of registration results using MI\_GC\_Powell and PEHL.

three applications, which is 20–45 times faster than that of MI\_GC\_Powell and leads to real-time registration at  $\sim 10$  fps. In addition, because the computation involved in PEHL is fixed for each registration, the standard deviation of the running time of PEHL is almost zero, so that PEHL can provide real-time registration at a stable frame rate. In comparison, intensity-based methods require different numbers of iterations for each registration, depending on the starting position, which leads to a relatively large standard deviation of the running time. The mTREproj percentiles show that at lower percentiles (e.g., 10th and 25th), the mTREproj of PEHL is in general larger than that of MI\_GC\_Powell. This is partially due to the fact that the ground truth parameters were generated using GC, which could bear a slight bias toward intensity-based methods using GC as the similarity measure. For higher percentiles (e.g., 75th and 90th), the mTREproj of PEHL becomes smaller than that of MI\_GC\_Powell, showing that PEHL is more robust than MI\_GC\_Powell. Some visual examples of registration results using MI\_GC\_Powell and PEHL are shown in Fig. 12.14.

## 12.8 DISCUSSION

In this chapter, we presented a CNN regression-based method, PEHL, for real-time 2-D/3-D registration. To successfully solve 2-D/3-D registration problems using regression, we introduced 3 novel algorithmic strategies, LIR, MSR, and PSP, to simplify the underlying mapping to be regressed, and designed a CNN regression model with strong nonlinear modeling capability to capture the mapping. We furthermore

validated that all 3 algorithmic strategies and the CNN model are important to the success of PEHL, by disabling them from PEHL and showing the detrimental effect on performance. We empirically found that applying PEHL for 3 iterations is the optimal setting, which leads to close to the optimal success rate and a real-time registration speed of  $\sim 10$  fps. We also demonstrated that PEHL has a strong ability to reproduce the same registration result from different initial positions, by showing that the RMSD<sub>proj</sub> of registered targets approaches to almost zero (i.e., 0.005 mm) as the number of iterations of PEHL increases to 6. In comparison, the RMSE<sub>proj</sub> using the best performed intensity-based method, MI\_GC\_Powell, is 0.52 mm. On three potential clinical applications, we compared PEHL with 4 intensity-based 2-D/3-D registration methods and a linear regression-based method, and showed that PEHL achieved much higher robustness and larger capture range. In particular, PEHL increased the capture range by 99–306% and the success rate by 5–27.8%, compared to MI\_GC\_Powell. We also showed that PEHL achieved significantly higher computational efficiency than intensity-based methods, and is capable of real-time registration.

The significant advantage of PEHL in robustness and computational efficiency over intensity-based methods is mainly due to the fact that CNN regressors are trained to capture the mapping from LIRs to the underlying transformation parameters. In every iteration, PEHL fully exploits the rich information embedded in LIR to make an informed estimation of the transformation parameters, and therefore it is able to achieve highly robust and accurate registration with only a minimum number of iterations. In comparison, intensity-based methods always map the DRR and X-ray images to a scalar-valued merit function, where the information about the transformation parameters embedded in the image intensities is largely lost. The registration problem is then solved by heuristically optimizing this scalar-valued merit function, which leads to an inefficient iterative computation and a high chance of getting trapped into local maxima.

The results also show that PEHL is more accurate and robust than two accelerated intensity-based 2-D/3-D registration methods, Sparse Histogramming MI (SHMI) [6] and Direct Splatting Correlation (DSC) [9], which employ sub-sampled DRR and splatting DRR to quickly compute approximated MI and CC, respectively. Because of the approximation, SHMI and DSC theoretically achieve the same or degraded accuracy compared to using original MI and CC. As shown in Table 12.3, all reported mTRE<sub>proj</sub> percentiles of PEHL are lower than that of MI\_Powell and CC\_Powell, and the differences at mid-range percentiles (i.e., 25th, 50th, and 75th) are quite significant. In particular, at the 50th percentile, the mTRE<sub>proj</sub> of PEHL are 25–65% lower than that of MI\_Powell and CC\_Powell on all three applications. These results suggest that PEHL significantly outperforms SHMI and DSC in terms of robustness and accuracy. In terms of computational efficiency, while all three methods are capable of real-time registration, with an efficient GPU implementation, DSC reported the highest registration speed (i.e., 23.6–92.3 fps) [9].

Like for any machine learning-based method, an important factor for the success of PEHL is the quantity and quality of the training data. For PEHL, it has been a

challenge to obtain sufficient amount of annotated real X-ray images for training, because accurate annotation of 3-D transformation on X-ray projection image is very difficult, especially for those out-of-plane parameters. We have shown that by generating well-simulated synthetic data and training the CNN network on synthetic data only, we could achieve high performance when applying PEHL on real X-ray images. However, it is worth noting that if the object to be registered is a device or implant that is manufactured with a fixed design, it is also possible to have a factory setup to massively acquire real X-ray images with a known ground truth for training PEHL.

---

## DISCLAIMER

This feature is based on research, and is not commercially available. Due to regulatory reasons its future availability cannot be guaranteed.

---

## REFERENCES

1. Rui Liao, Li Zhang, Ying Sun, Shun Miao, Christophe Chefd’Hotel, A review of recent advances in registration techniques applied to minimally invasive therapy, *IEEE Trans. Multimed.* 15 (5) (2013) 983–1000.
2. Primoz Markelj, D. Tomaževič, Bostjan Likar, F. Pernuš, A review of 3d/2d registration methods for image-guided interventions, *Med. Image Anal.* 16 (3) (2012) 642–661.
3. Christelle Gendrin, Hugo Furtado, Christoph Weber, Christoph Bloch, Michael Figl, Supriyanto Ardjo Pawiro, Helmar Bergmann, Markus Stock, Gabor Fichtinger, Dietmar Georg, et al., Monitoring tumor motion by real time 2d/3d registration during radiotherapy, *Radiother. Oncol.* 102 (2) (2012) 274–280.
4. Jérôme Schmid, Christophe Chênes, Segmentation of X-ray images by 3d–2d registration based on multibody physics, in: *Computer Vision – ACCV 2014*, Springer, 2014, pp. 674–687.
5. Shun Miao, Rui Liao, Yefeng Zheng, A hybrid method for 2-d/3-d registration between 3-d volumes and 2-d angiography for trans-catheter aortic valve implantation (TAVI), in: *2011 IEEE International Symposium on Biomedical Imaging: From Nano to Macro*, IEEE, 2011, pp. 1215–1218.
6. Lilla Zöllei, E. Grimson, Alexander Norbash, W. Wells, 2d–3d rigid registration of X-ray fluoroscopy and CT images using mutual information and sparsely sampled histogram estimators, in: *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2, *CVPR 2001*, IEEE, 2001, pp. II-609–II-703.
7. Wolfgang Birkfellner, Markus Stock, Michael Figl, Christelle Gendrin, Johann Hummel, Shuo Dong, Joachim Kettenbach, Dietmar Georg, Helmar Bergmann, Stochastic rank correlation: a robust merit function for 2d/3d registration of image data obtained at different energies, *Med. Phys.* 36 (8) (2009) 3420–3428.
8. Lee Westover, Footprint evaluation for volume rendering, in: *ACM SIGGRAPH Computer Graphics*, vol. 24, ACM, 1990, pp. 367–376.

9. Charles Hatt, Michael Speidel, Amish Raval, Robust 5DOF transesophageal echo probe tracking at fluoroscopic frame rates, in: Medical Image Computing and Computer-Assisted Intervention – MICCAI 2002, Springer, 2015.
10. Jens Kruger, Rüdiger Westermann, Acceleration techniques for GPU-based volume rendering, in: Proceedings of the 14th IEEE Visualization 2003, VIS'03, IEEE Computer Society, 2003, p. 38.
11. Michael M. Bronstein, Alexander M. Bronstein, Fabrice Michel, Nikos Paragios, Data fusion through cross-modality metric learning using similarity-sensitive hashing, in: 2010 IEEE Conference on Computer Vision and Pattern Recognition, CVPR, IEEE, 2010, pp. 3594–3601.
12. Fabrice Michel, Michael Bronstein, Alex Bronstein, Nikos Paragios, Boosted metric learning for 3d multi-modal deformable registration, in: 2011 IEEE International Symposium on Biomedical Imaging: From Nano to Macro, IEEE, 2011, pp. 1209–1214.
13. Ana Rodrigues Gouveia, Coert Metz, Luis Freire, Pedro Almeida, Stefan Klein, Registration-by-regression of coronary CTA and X-ray angiography, Comput. Methods Biomed. Eng. (2015) 1–13.
14. Chen-Rui Chou, Brandon Frederick, Gig Mageras, Sha Chang, Stephen Pizer, 2d/3d image registration using regression learning, Comput. Vis. Image Underst. 117 (9) (2013) 1095–1106.
15. Paul Wohlhart, Vincent Lepetit, Learning descriptors for object recognition and 3d pose estimation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 3109–3118.
16. Piotr Dollár, Peter Welinder, Pietro Perona, Cascaded pose regression, in: 2010 IEEE Conference on Computer Vision and Pattern Recognition, CVPR, IEEE, 2010, pp. 1078–1085.
17. Christopher Zach, Adrian Penate-Sánchez, Minh-Tri Pham, A dynamic programming approach for fast and robust object pose recognition from range images, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 196–203.
18. Roozbeh Mottaghi, Yu Xiang, Silvio Savarese, A coarse-to-fine model for 3d pose estimation and sub-category recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 418–426.
19. Markus Kaiser, Matthias John, Tobias Heimann, Alexander Brost, Thomas Neumuth, Georg Rose, 2d/3d registration of tee probe from two non-orthogonal C-arm directions, in: Medical Image Computing and Computer-Assisted Intervention – MICCAI 2014, Springer, 2014, pp. 283–290.
20. Dejan Tomažević, Bostjan Likar, Tomaž Slivnik, Franjo Pernuš, 3-d/2-d registration of CT and MR to X-ray images, IEEE Trans. Med. Imaging 22 (11) (2003) 1407–1416.
21. Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton, ImageNet classification with deep convolutional neural networks, in: Advances in Neural Information Processing Systems, 2012, pp. 1097–1105.
22. Xavier Glorot, Yoshua Bengio, Understanding the difficulty of training deep feedforward neural networks, in: International Conference on Artificial Intelligence and Statistics, 2010, pp. 249–256.
23. Z. Zhu, S. Ji, M. Yang, H. Ding, G. Wang, An application of the automatic 2d–3d image matching technique to study the in-vivo knee joint kinematics before and after TKA, in: World Congress on Medical Physics and Biomedical Engineering, May 26–31, 2012, Beijing, China, Springer, 2013, pp. 230–233.

24. S.Y. Vetter, I. Mühlhäuser, J. von Recum, P.-A. Grützner, J. Franke, Validation of a virtual implant planning system (VIPS) in distal radius fractures, *Bone Joint J. Orthop. Proc. Suppl.* 96 (SUPP 16) (2014) 50.
25. Gang Gao, Graeme Penney, Yingliang Ma, Nicolas Gogin, Pascal Cathier, Aruna Arujuna, Geraint Morton, Dennis Caulfield, Jaswinder Gill, C. Aldo Rinaldi, et al., Registration of 3d trans-esophageal echocardiography to X-ray fluoroscopy using image-based probe tracking, *Med. Image Anal.* 16 (1) (2012) 38–49.
26. Everine B. De Kraats, Graeme P. Penney, Dejan Tomažević, Theo Van Walsum, Wiro J. Niessen, Standardized evaluation methodology for 2-d–3-d registration, *IEEE Trans. Med. Imaging* 24 (9) (2005) 1177–1189.
27. Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, Trevor Darrell, Caffe: convolutional architecture for fast feature embedding, *arXiv:1408.5093*, 2014.
28. Markus Kaiser, Matthias John, Tobias Heimann, Thomas Neumuth, Georg Rose, Comparison of optimizers for 2d/3d registration for fusion of ultrasound and X-ray, in: *Bildverarbeitung für die Medizin 2014: Algorithmen–Systeme–Anwendungen*, Proceedings des Workshops vom 16. bis 18. März 2014 in Aachen, Springer, 2014, p. 312.

PART

# Computer-Aided Diagnosis and Disease Quantification

5

This page intentionally left blank

# Chest Radiograph Pathology Categorization via Transfer Learning

# 13

**Idit Diamant<sup>\*,1</sup>, Yaniv Bar<sup>\*,1</sup>, Ofer Geva<sup>\*</sup>, Lior Wolf<sup>\*</sup>, Gali Zimmerman<sup>\*</sup>,  
Sivan Lieberman<sup>†</sup>, Eli Konen<sup>†</sup>, Hayit Greenspan<sup>\*</sup>**

*Tel-Aviv University, Ramat-Aviv, Israel<sup>\*</sup> Sheba Medical Center, Tel-Hashomer, Israel<sup>†</sup>*

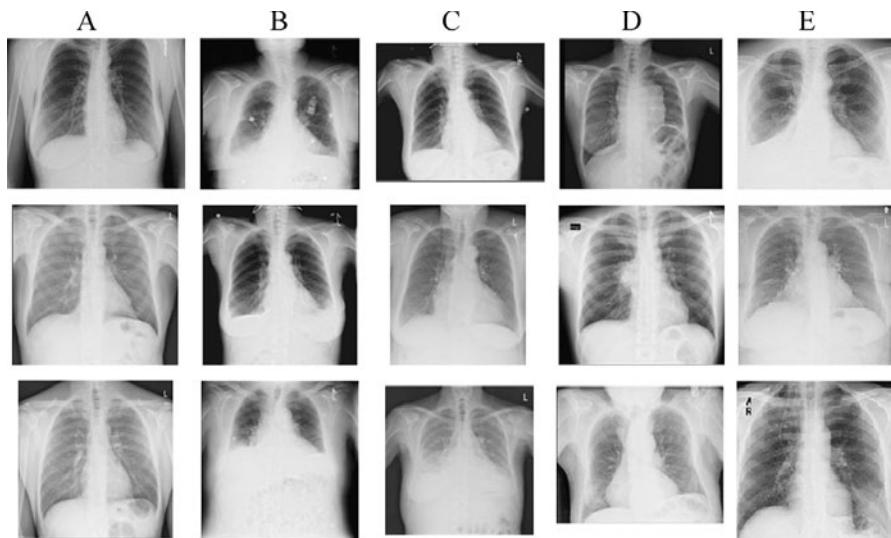
## CHAPTER OUTLINE

<b>13.1</b>	<b>Introduction</b>	300
<b>13.2</b>	<b>Image Representation Schemes with Classical (Non-Deep) Features</b>	303
13.2.1	Classical Filtering	304
13.2.2	Bag-of-Visual-Words Model	305
<b>13.3</b>	<b>Extracting Deep Features from a Pre-Trained CNN Model</b>	306
<b>13.4</b>	<b>Extending the Representation Using Feature Fusion and Selection</b>	309
<b>13.5</b>	<b>Experiments and Results</b>	309
13.5.1	Data	309
13.5.2	Experimental Setup	310
13.5.3	Experimental Results	310
13.5.3.1	Feature Selection Analysis	313
<b>13.6</b>	<b>Conclusion</b>	315
<b>Acknowledgements</b>		317
<b>References</b>		318

## CHAPTER POINTS

- Overview of X-ray analysis: from BoW to deep learning
- Deep learning can be used via transfer learning from an existing network
- Medical images can be represented via deep-network signature
- Transfer learning enables image multi-label categorization

<sup>1</sup>Equal contributors.

**FIGURE 13.1**

Chest X-rays categories examples: (A) healthy, (B) left or right pleural effusion, (C) enlarged heart (cardiomegaly), (D) enlarged mediastinum, (E) left or right consolidation.

*Source: Diagnostic Imaging Department, Sheba Medical Center, Tel Hashomer, Israel*

### 13.1 INTRODUCTION

With approximately 2 billion procedures per year, radiographs (X-rays) are the most common imaging examination in radiology. Adult chest radiographs are a major part of these procedures. They are the most commonly ordered screening test for pulmonary disorders. Chest radiographs are performed to evaluate the lungs, heart, and thoracic viscera. They are essential for the management of various diseases associated with high mortality, including pneumonia, heart failure, pleurisy, and lung cancer. Diagnosis of the various chest conditions is a difficult task even to human experts, due to the subtlety of the information.

Examples of pathological chest radiographs are shown in Fig. 13.1. Several examples per pathology are shown: *Pleural effusion* is excess fluid that accumulates in the pleural cavity, the fluid-filled space that surrounds the lungs. This excess of fluid can impair breathing by limiting the expansion of the lungs. Enlarged heart, otherwise termed *Cardiomegaly*, produces an abnormally large shadow (cardiac silhouette) on a chest X-ray film. Detection of cardiomegaly is often conducted using the cardiothoracic ratio which is defined as the ratio of the maximal horizontal cardiac diameter and the maximal horizontal thoracic diameter (inner edge of ribs or edge of pleura). The heart is enlarged if the cardiothoracic ratio is greater than 50%. *Mediastinal abnormalities* are important radiological findings. The causes of mediastinal widening can be divided into traumatic and nontraumatic and include vascular abnormality and



**FIGURE 13.2**

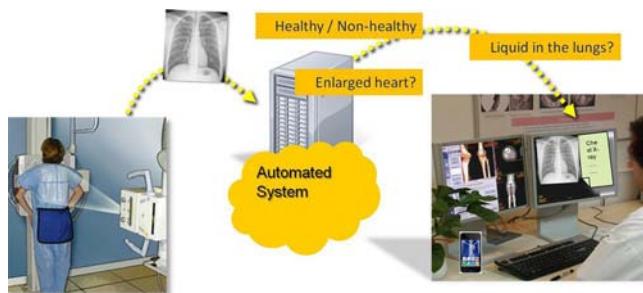
An example of multiple pathologies per patient. The patient is diagnosed with right pleural effusion, enlarged heart, and enlarged mediastinum.

mediastinal mass. *Pulmonary opacification* refers to a visible area of increased attenuation, related to a decrease in the ratio of gas to soft tissue in the lung. This finding often indicates a replacement of air in the alveoli or small airways with dense material (e.g., fluid, blood, cells). Pulmonary opacities may be seen in different kinds of lung diseases such as pneumonia, cancer, and tuberculosis.

Detecting cardiomegaly and pleural effusion in frontal X-rays seems a feasible task since in most cases the heart is either clearly normal in size or clearly abnormally enlarged, and lungs shape and expansion capabilities also stand out from the normal case. Accurate detection of lung consolidation, on the other hand, is a more difficult pathology to detect and characterize correctly. In a clinical setting, it is often the case that *multiple* pathologies are present in a given chest radiograph. An example of such a case is highlighted in Fig. 13.2, illustrating a patient clinically diagnosed with right pleural effusion, enlarged heart, and enlarged mediastinum.

The increasingly growing workload on the radiology staff in radiology departments around the world leads to the reality that radiographs may not be read by the radiologists staff or the read is following a long, sometimes life endangering delay. Therefore, there is an interest in developing automated computer analysis systems to assist the radiologists in the reading task. An automated software which can discriminate between healthy and non-healthy cases is of substantial need, to support initial screening of the examined cases. Moreover, an automated system that shows high sensitivity and specificity in categorizing a set of pathologies can be of great value in real clinical settings. It is expected that such systems can improve accuracy and efficiency in the radiology departments of the Western world, and can be critical to healthcare in third world countries (e.g., China), where no substantial radiology service is available.

A topic of great interest for analysis in chest X-rays is the detection of lung nodules, which are important as a precursor for cancer. Although the target of most research attention, lung nodules are a relatively rare finding in the lungs. The most common findings in chest X-rays include lung infiltrates, catheters, and abnormalities

**FIGURE 13.3**

Automated system for multi-label pathology identification.

of the size or contour of the heart [1]. In our work we focus on the more frequently appearing set of pathologies that need to be written in the radiology report.

Most works found in the literature are *single task* focused. A set of segmentation and landmark localization tools are used to address the specific task at hand. Methods used are often algorithmically and computationally challenging, requiring, for example, segmentation of the lungs, suppression of ribs, and localization of typical lung textures (detection of emphysema [2], diagnosis of interstitial lung diseases [3]).

Very few studies can be found in the literature that focus on chest pathology identification and classification as an *image-level* labeling task (e.g., [4,5]). In [4] healthy versus pathological identification was explored using Local Binary Patterns (LBP). Avni et al. [5] used the Bag-of-Visual-Words (BoVW) model [6] to discriminate between healthy and four pathological cases. The BoVW methodology was proven to lead the ImageClef competitions (<http://www.imageclef.org>) in categorizing X-rays on the organ level (2008, 2009). The success of the BoVW framework on the organ level led to its extension to pathology level categorization of the chest radiographs [5].

Our objective in the current work is to explore the role of a Deep Learning approach as an automated image-level system for pathology categorization of chest-related diseases and in the screening of healthy versus non-healthy cases. See Fig. 13.3 for illustration.

Deep Learning is a class of machine learning techniques, where many layers of information processing stages in hierarchical supervised architectures are exploited for feature learning and for pattern analysis/classification. The essence of deep learning is to compute hierarchical features or representations of the observed data, where the higher-level features or factors are defined from lower-level ones [7]. Deep (i.e., many-layered) convolutional neural networks (CNNs) for machine object recognition, are advancing the limits of performance in domains as varied as computer vision, speech, and text, and are considered as one of the leading technologies for recognition [8,9]. Recent results indicate that the generic descriptors extracted from CNNs are extremely effective in object recognition and provide better results than

systems relying on carefully engineered representations, such as SIFT or HOG features [10,11].

Deep learning methods are most effective when applied to networks with large number of training data to train the deep neural network. In the computer-vision domain such large image sets exist and enable the training of popular CNNs in many image recognition tasks, such as the large scale visual recognition challenge of ImageNet [12–16]. Other domains exist in which there is less data for training. Recently, works have come out in the general computer-vision literature that use *Transfer Learning* – an efficient way to utilize image representations learned with CNNs on large-scale annotated datasets, defined as the source, to domains in which limited data exists, termed the target domain. Such tools are also entering the medical imaging domain, as can be seen by the emerging set of works in the field. For a general overview see [17].

In this chapter we provide an overview of our exploration into two main issues:

- Can we provide a system to automatically analyze an input radiograph to detect multiple pathologies and to provide *multiple labels* per case? Note that our goal is to screen for healthy or non-healthy, and to categorize all pathologies present in the image, in an efficient and robust framework that can adapt to a real clinical setting.
- Can we utilize the deep learning methodology for this medical task? We assume a scenario in which medical data is limited and no network training can be done. We explore if the general-image trained deep network features, trained on the ImageNet data (source domain), provide a robust representation which we can use to categorize the chest radiograph data (target domain).

We show classification results of six different pathologies, on several hundreds of cases from a real clinical setting. Using the deep learning framework, we show categorization results for all pathologies, while exploring further the deep network features using fusion and selection schemes.

---

## 13.2 IMAGE REPRESENTATION SCHEMES WITH CLASSICAL (NON-DEEP) FEATURES

Developing an image recognition solution that is based on a new set of features must be compared against a strong baseline of well established feature extraction techniques. As a benchmark we apply a set of classical descriptors that are known in the literature to perform well in image classification/categorization tasks. These include GIST [18] features, Pyramid Histogram of Oriented Gradients (PHOG) [19], Gabor [20], and Gray-Level Co-occurrence Matrix (GLCM) statistics [21]. A brief overview of these filters is described in Section 13.2.1. We also compare the deep feature representation to the more recent state-of-the-art representation of the Bag-of-Visual-Words (BoVW) model [6]. A brief review of the BoVW is provided in Section 13.2.2.

### 13.2.1 CLASSICAL FILTERING

The *GIST descriptor* proposed in [18] provides a low dimensional representation of a scene which does not require any form of segmentation. The descriptor focuses on the shape of the scene, on the relationship between the outlines of the surfaces and their properties, and ignores the local objects in the scene and their relationships. It is derived by resizing an image to  $128 \times 128$  and iterating over different scales (4 scales in our case) where for each scale the image is divided into  $8 \times 8$  cells. For each cell, orientation (every 45 degrees), color, and intensity histograms are extracted. The descriptor is a concatenation of all histograms, for all scales and cells. The GIST descriptor is similar in spirit to the local SIFT descriptor [24]. It was found to be helpful in scene recognition, e.g., in [22,23].

The *PHOG descriptor*, proposed by Bosch et al. [25], represents an image by its local shape and the spatial layout of the shape. Local shape is captured by the distribution over edge orientations within a region, and spatial layout by tiling the image into regions at multiple resolutions. The descriptor is derived by iterating over different scales (2 scales in our case) where for each scale, the image is first divided into cells and for each cell a histogram of gradient directions or edge orientations is extracted. The descriptor is a concatenation of all histograms, for all scales and cells. PHOG has been successfully applied to object classification in recent years [19,26].

Both GIST and PHOG descriptors are known for capturing the statistical information of a scene, by capturing local shapes based on the distribution of data within the regions of interest, and capturing spatial layout based on the tiling of the image into regions of multiple resolutions.

A closely related descriptor is the Gabor filter set [20]. Gabor descriptors can be considered as edge detectors with adjustable orientations and scales. The descriptors are extracted by creating a Gabor filter bank that comprises a set of Gaussian filters that cover the frequency domain with different radial frequencies and orientations (5 wavelengths and 8 orientations in our case), these filters are convolved with the input image to produce corresponding set of response matrices (40 in our case) of the image. In this work, we use two Gabor-derived representations: A *Gabor-raw descriptor*, which is the result of vectorization of all the response matrices downsampled by a factor of 4, and a *Gabor-statistical-based descriptor* that is generated by combining measurements of mean, standard deviation, energy, and entropy for each response matrix from each scale (2, 4, 8, 16, 32) and orientation (0, 30, 45, 60, 90, 120, 135, 150). Gabor filters have been found to be very effective in texture representation and discrimination [20,27].

The gray-level co-occurrence matrix (GLCM) [21] is a statistical method that considers the spatial relationship of pixels in order to extract texture information. A single GLCM matrix represents how often pairs of pixel with specific values at a specified spatial relationship, occur in an image. Element  $(i, j)$  of the GLCM matrix is generated by counting the number of times a pixel with value  $i$  is adjacent to a pixel with value  $j$  and then dividing the entire matrix by the total number of comparisons made. Each entry is therefore considered to be the probability that a pixel with value  $i$  will be found adjacent to a pixel of value  $j$ . GLCM matrices with differ-

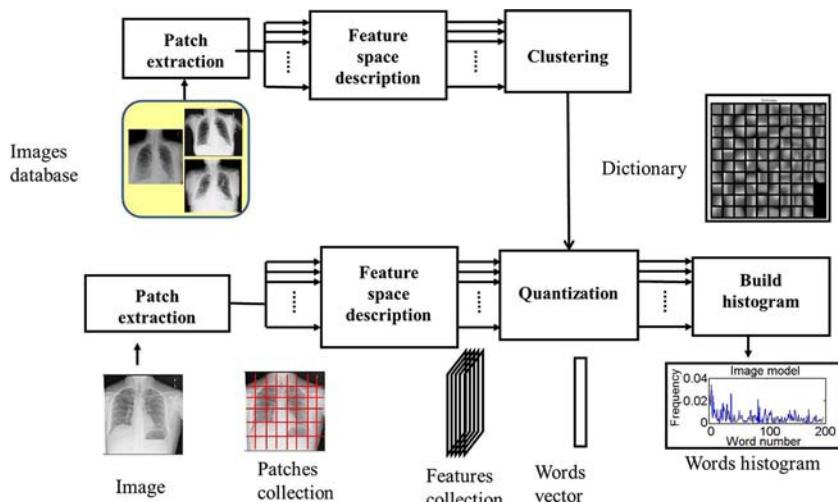
ent horizontal, vertical, and diagonal offsets are generated. Because GLCM matrices are typically large and sparse, they are often represented by a set of features, known as Haralick features [21], which are extracted to form a unique image descriptor. The GLCM image descriptor is very common in the medical image analysis domain [28,29].

### 13.2.2 BAG-OF-VISUAL-WORDS MODEL

The BoVW [6] image representation provided in recent years the state-of-the-art image classification method. Adapted from the bag-of-words (BoW) representation of text documents, the BoVW method includes the following steps (Fig. 13.4): (i) Patch extraction – to extract uniform size patches from the image ROIs. Each small patch shows a localized view of the image content. These patches are considered as candidates for basic elements, or visual-words. The patch size needs to be larger than a few pixels across in order to capture higher-level semantics such as edges or corners. At the same time, the patch size should not be too large if it is aimed to serve as a common building block for many images. (ii) Feature description – feature representation involves representing the patches using feature descriptors. Raw image data, normalized raw data, or other descriptors can be used. To reduce the computational complexity of the algorithm and the level of noise, a principal component analysis procedure (PCA) can be applied to this patch collection. The first few components of the PCA, which are the components with the largest eigenvalues, serve as a basis for the information description. (iii) Quantization and clustering – the final step of the bag-of-words model is to convert vector-represented patches into visual words and generate a representative dictionary. A frequently-used method is to perform K-means clustering over the vectors of the initial collection. The vectors are then clustered into groups in the feature space. The resultant cluster centers serve as a vocabulary of visual words. A given image can now be represented by a unique distribution over the generated dictionary of words, which is a representative image histogram.

The BoVW method has been successful in medical classification tasks, such as medical image retrieval [30], retrieval of similar-appearing liver lesions [31], the classification of breast tissue [32], and retrieval of brain tumors in MRI images [33]. Our group participated in the 2009 imageCLEF X-ray categorization competition using the BoVW model, and was ranked first [5]. The model used dense sampling of simple features with spatial content, and a nonlinear kernel-based SVM classifier. Motivated by the success in the competition on the organ-level categorization, we extended the system to pathology-level categorization of chest X-ray data. Additionally, we developed BoVW variants which were used in different tasks such as classification of breast tissue [34], liver lesion classification (dual dictionary model [35]) and multi-phase liver lesion classification (relevant words representation [36]).

As a benchmark for the current study, we use the same implementation described in [5]: We add the patch center coordinates to the feature vector to introduce spatial

**FIGURE 13.4**

A schematic illustration of BoVW model.

information to the image representation, without the need to explicitly model the spatial dependency between patches. Empirically, we found that using a 9-dimensional patch representation which comprises 7 PCA components of variance-normalized raw patches along with 2 patch coordinates and using a dictionary size of 1000 words results in best classification performance.

### 13.3 EXTRACTING DEEP FEATURES FROM A PRE-TRAINED CNN MODEL

CNNs constitute a feed-forward family of deep networks where intermediate layers receive as input the features generated by the former layer and pass their outputs to the next layer. The initial layers of the CNN are locally-connected, alternating convolution and pooling layers, and the final layers of a network are fully-connected. This hierarchy of layers enables different levels of abstraction in the input representation: For visual data, the low levels of abstraction describe the different orientated edges and lines in the image; middle levels describe parts of an object such as corners, angles and surface boundaries, while high layers refer to larger object parts and even complete objects.

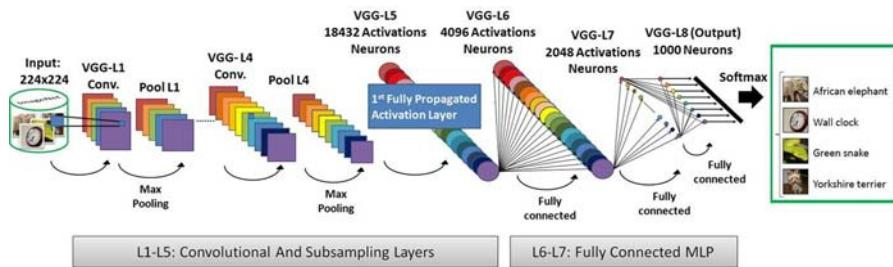
In recent years, CNNs have gained considerable interest due to the development of new variants of networks and the advent of efficient parallel solvers optimized for modern GPUs. Advanced computing resources have allowed deeper and more com-

plex convolutional networks to be trained. Numerous CNN architectures that improve the previous state-of-the-art classification/analysis results obtained using shallow representations have been proposed.

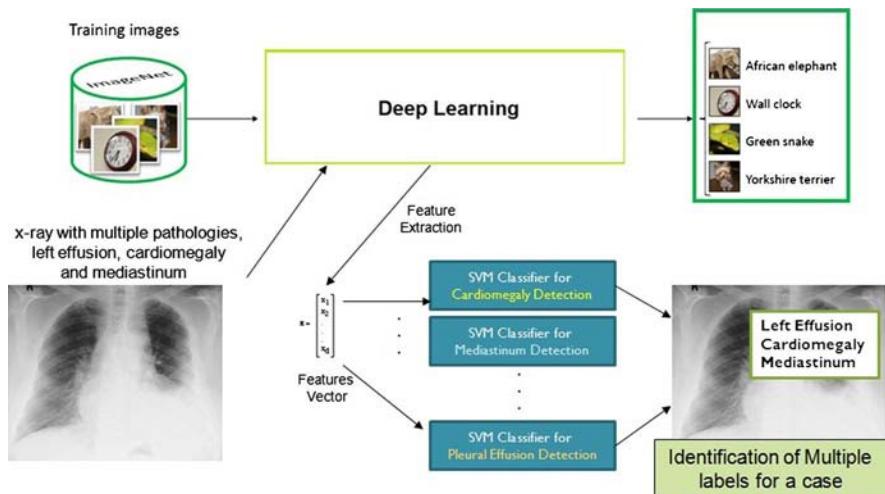
A well known model is the Krizhevsky et al. CNN representation [37] which was learned over a subset of images from ImageNet [12], a comprehensive real-life large scale image database ( $>20M$ ) that is arranged into more than 15K non-medical concepts/categories. A follow up network architecture which was inspired by the Krizhevsky network, was the Zeiler and Fergus [14] CNN model. Zeiler and Fergus network is constructed from five convolutional layers, interleaved with nonlinear and pooling operations, followed by two fully connected classification layers and the fully connected output layer. In their network, Zeiler and Fergus attempted to correct problems in Krizhevsky et al. model without changing its learning framework. Zeiler and Fergus main claim was that the first convolutional layer filters are a mix of extremely high and low frequency information, with little coverage of the mid frequencies. It was also claimed that due to the first convolutional layer large stride, the second layer visualization show aliasing artifacts. Zeiler and Fergus therefore reduced the first layer kernel size and made a decreased stride size, allowing the new architecture to retain much more feature information in the first two layers. It was shown that the architecture slightly outperforms Krizhevsky et al. architecture in the ImageNet classification challenge.

Obtaining datasets in the medical imaging domain that are as comprehensively annotated, such as the ImageNet data, remains a challenge. Transfer learning and fine tuning are key components in the use of deep CNNs in medical imaging applications [17]. In transfer learning, CNN models (supervised) pre-trained from natural image dataset or from a different medical domain are used for a new medical task at hand. In one scheme, a pre-trained CNN is applied to an input image and then the outputs are extracted from layers of the network. The extracted outputs are considered features and are used to train a separate pattern classifier. For instance, in [38–40], pre-trained CNNs were used as a feature generator for chest pathology identification. In [41] integration of CNN-based features with handcrafted features enabled improved performance in a nodule detection system. Fine tuning is relevant when a medium sized dataset does exist for the task at hand. One suggested scheme is to use a pre-trained CNN as initialization of the network, following which further supervised training is conducted, of several (or all) the network layers, using the new data for the task at hand. In this chapter we focus on the use of transfer learning, with no additional network learning.

We show our initial set of results [39] using the Decaf CNN representation which closely follows the network of Krizhevsky et al. [15]. We then proceed to use the VGG-M-2048 pre-trained CNN model [13] which has a similar architecture to the one used by Zeiler and Fergus with the exception of a very minor change in the first two convolutional layers stride size which was introduced to keep the computation time reasonable, along with a reduction in the number of filters in the fourth convolutional layer. A schematic illustration is provided in Fig. 13.5.

**FIGURE 13.5**

A schematic illustration of VGG-M-2048 CNN architecture and training process [13].

**FIGURE 13.6**

Identification of multiple X-ray pathologies via transfer learning.

Features from all layers are extracted (excluding the output layer), as the network representation for the identification task. We denote by *VGG-L1* up to *VGG-L5* the convolutional layers, where *VGG-L5* is the final convolutional layer and is the first set of activations that has been fully propagated through the convolutional layers of the network. *VGG-L6* denotes the first fully-connected layer. We use the notation in [15] to denote the 5th–7th activation layer features of the Decaf network as the network representation.

In the current study, we compare across the different representation schemes, across varying networks. Each selected feature set is input to the SVM classifier, and results are compared. A schematic overview of the full X-ray categorization process using transfer learning is provided in Fig. 13.6.

---

## 13.4 EXTENDING THE REPRESENTATION USING FEATURE FUSION AND SELECTION

In further analysis we also investigated variations, such as feature fusion and selection, that can be applied to the features with an attempt to augment results and increase robustness.

Feature selection, a process of selecting a subset of original features according to certain criteria, is an important and frequently used dimensionality reduction technique for data mining. In the past few decades, researchers have developed many feature selection algorithms. These algorithms are designed to serve different purposes, are based on different models, and all have their own advantages and disadvantages. A repository of feature selection algorithms can be found in [42].

In the current work we use the Information Gain feature selection method [43]. Information Gain is a measure of dependence between the feature and the class label. It is one of the most popular feature selection techniques as it is easy to compute and simple to interpret. The Information Gain (IG) of a feature  $X$  and the class labels  $Y$  is defined as

$$IG(X, Y) = H(X) - H(X|Y), \quad (13.1)$$

the entropy of  $X$  and the entropy of  $X$  after observing  $Y$ , respectively:

$$H(X) = - \sum_i P(x_i) \log_2(P(x_i)), \quad (13.2)$$

$$H(X|Y) = - \sum_j P(y_j) \sum_i P(x_i|y_j) \log_2(P(x_i|y_j)). \quad (13.3)$$

A feature with high information gain is more relevant for a given task (with max IG equal to 1). Information gain is evaluated independently for each feature and the features with the top- $k$  values are selected as the relevant features.

The rationale for applying the feature selection scheme is derived from the fact that it is hard to predict in advance which pre-trained CNN layer or layers will be the most powerful feature representation, given a specific task. Indeed, it is known from [14] and [15] that the deeper layers of the network can be used as a powerful image descriptor applicable to other datasets, however, Zeiler and Fergus [14] point out the 7th layer as the most significant layer, Donahue et al. [15] point out the 6th layer, and in our earlier work [39], we have pointed out the 5th layer.

---

## 13.5 EXPERIMENTS AND RESULTS

### 13.5.1 DATA

Our dataset consists of 637 frontal chest X-ray images in DICOM format. The images are of variable size. They are cropped, centered, and contain several artifacts such

as reading directives (e.g., arrows, left/right indicators) and medical equipment, but otherwise were not preprocessed (e.g., equalization, scaling). We have replicated the Intensity channel to support the CNN 3-channel RGB input data expectations. The pertained CNN resizes the images into specific accepted resolution automatically.

The images were collected from the Diagnostic Imaging Department of Sheba Medical Center, Tel Hashomer, Israel. Gold standard was achieved using image interpretation done by two expert radiologists. The radiologists examined all of the images independently and then reached a decision regarding the label of every image. For each image, a positive or negative label was assigned. In cases of disagreement, the image was removed from the dataset.

The images depict 6 chest pathology conditions: Right Pleural Effusion (RPE) – 73 images; Left Pleural Effusion (LPE) – 74 images; Right Consolidation (RCN) – 58 images; Left Consolidation (LCN) – 45 images; Cardiomegaly/Enlarged Heart (Cardio) – 154 images, and Abnormal (Enlarged) Mediastinum (MED) – 145 images. Overall, the dataset contains 325 images with at least one pathological condition.

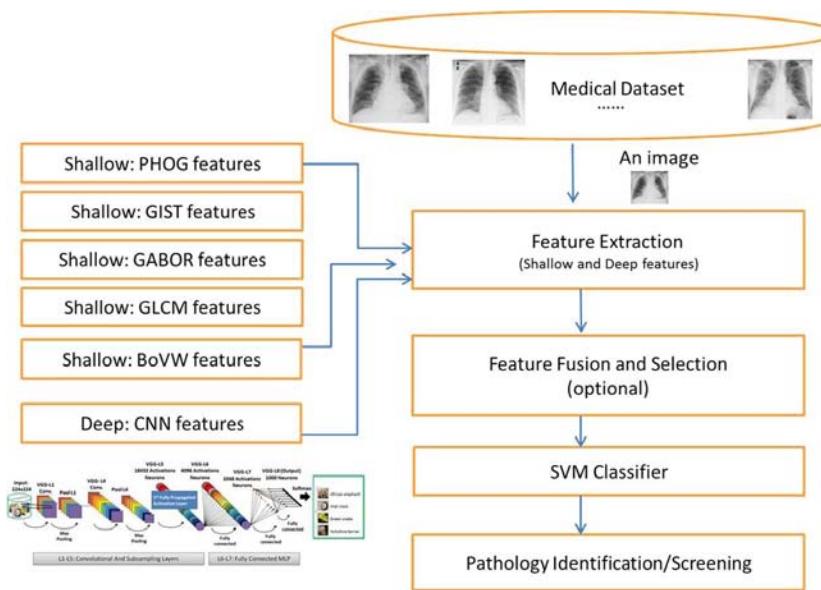
### 13.5.2 EXPERIMENTAL SETUP

We performed a binary categorization task for each pathology. Classification was performed using a Support Vector Machine (SVM) classifier with a nonlinear intersection kernel. We investigated the different linear and nonlinear kernels using standard grid-search technique, and empirically selected the nonlinear intersection kernel, which gave slightly better results. For each binary categorization task, cases diagnosed with an examined pathology were labeled as positive cases, and cases diagnosed with the absence of the examined pathology were labeled as negative cases. In the screening scenario, cases diagnosed with the absence of any pathology were labeled as positive cases, whereas cases diagnosed with the presence of at least one of the examined pathologies were labeled as negative cases. We used  $k$ -folded cross-validation with  $k = 4$ . All features used in our work were normalized: each feature across all images had its mean subtracted and was divided by its standard deviation. The algorithm flowchart is shown in Fig. 13.7.

### 13.5.3 EXPERIMENTAL RESULTS

We start with a comparison across varying representation schemes: from shallow features (PHOG, GLCM, GABOR, and GIST), to medium level representation (BoVW) and deep features of the deep VGG-M-2048 CNN layer features (for layers  $VGG-L1$ , ...,  $VGG-L7$ ). Table 13.1 presents accuracy results, measured as the area under the ROC curve (AUC).

We also examine the likelihood ratio measurement which is based on the Sensitivity and Specificity metrics [44,45]. In evidence-based medicine, likelihood ratios are used for assessing the value of performing a diagnostic test. Positive likelihood ratio metric is defined as the ratio between the probability of a positive test result given the presence of the disease and the probability of a positive test result given the absence

**FIGURE 13.7**

Algorithm flowchart.

of the disease. This is calculated as

$$LR+ = \text{Sensitivity}/(1 - \text{Specificity}).$$

The negative ratio metric is defined as the ratio between the probability of a negative test result given the presence of the disease and the probability of a negative test result given the absence of the disease. This is calculated as

$$LR- = (1 - \text{Sensitivity})/\text{Specificity}.$$

A likelihood ratio greater than 1 indicates the test result is associated with the disease, a likelihood ratio less than 1 indicates that the result is associated with absence of the disease, and likelihood ratios that lie close to 1 have little practical significance.

From [Table 13.1](#) we observe that for all pathology identification cases, the deep architecture descriptors outperformed the shallow descriptors, with a relatively large performance gap in comparison to statistical based representation such as GLCM and Gabor, and to a lesser extent, in comparison to more sophisticated representation such as BoVW and GIST. CNN intermediate layer features (*VGG-L4*) obtained an average increase of 2–4% AUC over GIST and BoVW features. In terms of  $LR+$ , *VGG-L4* scored an average of 4.89 against 3.61 and 3.98 for GIST and BoVW features, respectively, reflecting an increase of more than 20%. Similarly, *VGG-L4*

**Table 13.1** AUC classification results for classical and network features. The descriptor dimensionality appears in parentheses

Descriptor	RPE	LPE	RCN	LCN	Cardio	MED	Healthy	Avg.
VGG-L4 (86,528)	0.93	0.92	0.77	0.74	0.95	0.85	0.89	0.86
VGG-L5 (18,432)	0.92	0.92	0.77	0.72	0.95	0.86	0.88	0.86
GIST (512)	0.88	0.84	0.82	0.64	0.93	0.80	0.87	0.82
BoVW (1000)	0.90	0.93	0.75	0.70	0.93	0.83	0.87	0.84
PHOG (336)	0.82	0.70	0.71	0.70	0.88	0.71	0.79	0.76
GLCM (440)	0.74	0.68	0.59	0.55	0.80	0.64	0.75	0.68
GABOR-Stat. (160)	0.71	0.69	0.56	0.48	0.82	0.68	0.82	0.68
GABOR-Raw (163,840)	0.89	0.88	0.81	0.65	0.95	0.85	0.89	0.85

**Table 13.2** AUC classification results for VGG-M-2048 CNN deep representations. The descriptor dimensionality appears in parentheses

Descriptor	RPE	LPE	RCN	LCN	Cardio	MED	Healthy	Avg.
VGG-L1 (279,936)	0.88	0.88	0.75	0.66	0.95	0.86	0.88	0.84
VGG-L2 (43,264)	0.91	0.91	0.77	0.73	0.96	0.86	0.89	0.86
VGG-L3 (86,528)	0.91	0.92	0.78	0.75	0.95	0.86	0.89	0.87
VGG-L4 (86,528)	0.93	0.92	0.77	0.74	0.95	0.85	0.89	0.86
VGG-L5 (18,432)	0.92	0.92	0.77	0.72	0.95	0.86	0.88	0.86
VGG-L6 (4096)	0.89	0.92	0.79	0.76	0.94	0.86	0.88	0.86
VGG-L7 (2048)	0.89	0.91	0.76	0.79	0.92	0.85	0.87	0.85

scored an average  $LR$  – of 0.26, against 0.31 and 0.27 for GIST and BoVW features, respectively, reflecting an improvement as well.

In Table 13.2 we show the AUC classification results for each one of the deep network layers. We can observe that intermediate deep layers, excluding the penultimate layer, provide the strongest representation. In particular, the *VGG-L4* and *VGG-L5* layers of the network. Another observation is that the first VGG layer results in the lowest performance. A strong resemblance in performance can be seen between the *VGG-L1* and the GABOR raw representation performance, for the different tasks. The first convolutional layer applies many filters to its natural image input. This extracts different features of the input which are normally tuned to edges of different orientations, frequency, and color. Due to the fact that our data comprised a replicated intensity channel, the features that are extracted by the network resemble features that are extracted using Gabor filters. This is a behavior that was exhibited in many modern deep neural networks: when trained on images, they all tend to learn first-layer features that resemble either Gabor filters or color blobs [46]. This also provides a biologically-inspired explanation that links the CNN to the human visual system.

We next compared two well known network schemes: the Decaf network and the VGG. In Table 13.3 we present results of layers 5 through 7 of the Decaf CNN (*Decaf-L5*, *Decaf-L6*, *Decaf-L7*). These results are slightly worse than when using

**Table 13.3** AUC classification results: comparing between deep networks. The descriptor dimensionality appears in parentheses

Descriptor	RPE	LPE	RCN	LCN	Cardio	MED	Healthy	Avg.
DECAF-L5 (9216)	0.93	0.90	0.77	0.77	0.94	0.85	0.89	0.86
DECAF-L6 (4096)	0.92	0.89	0.76	0.80	0.93	0.86	0.89	0.86
DECAF-L7 (4096)	0.90	0.87	0.73	0.79	0.91	0.84	0.87	0.84

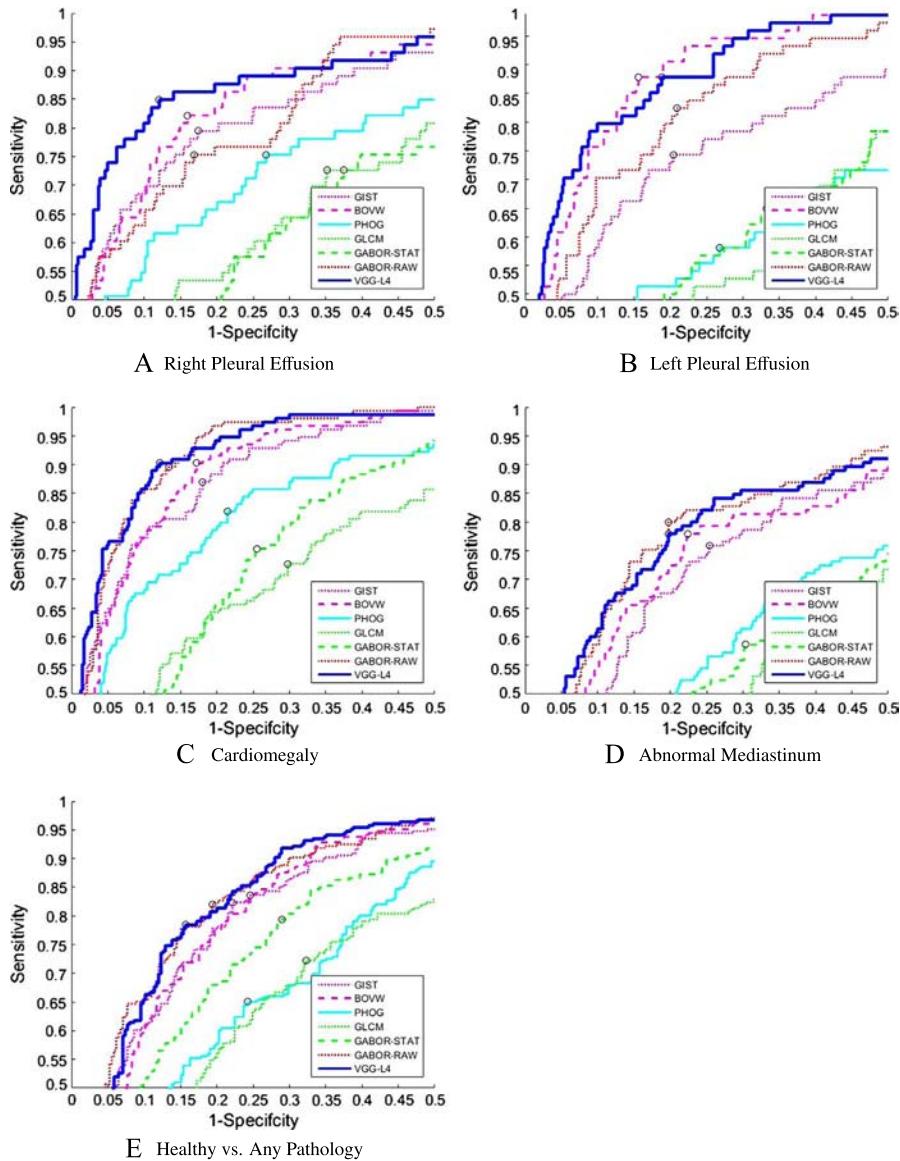
the VGG-M-2048 CNN features: an average  $LR+$  of 4.89 was obtained for *VGG-L4* with 4.60 for *DECAF-L5* features. This reflects an increase of approximately 6% with the same  $LR-$  measurement of 0.26; At the same average AUC of 0.86, VGG intermediate layers perform better in the identification of the Left Pleural Effusion pathology than *Decaf*, with a 2% AUC increase. As the VGG network consistently held the higher scores, we hereon focus on results using the VGG network.

Examples of ROC curves are shown in Fig. 13.8 for the right and left effusion, cardiomegaly, and abnormal mediastinum pathology identification tasks. In all cases the network features provide either the strongest or a match to the strongest representation.

### 13.5.3.1 Feature Selection Analysis

So far we observed the strength of deep *VGG* layers features for the various identification tasks. We also note fluctuations between the different layers, with an average AUC gap of 3% (ranging from 1 to 6%) across all pathologies. Although *VGG-L4* and *VGG-L5* present strong results, no single layer presents the best score consistently across all pathologies. We next wish to explore feature selection as a means for increasing robustness to the labeling tasks. We use the following steps: (i) *VGG* features from all layers, excluding the first convolutional layer, are fused together, in what is known as fusion of features. The features across all layers (L2–L7) are concatenated into an extended feature-vector. (ii) Feature selection is performed on the joint feature vector in order to remove redundant features. And (iii) the selected set of features are input to the SVM classifier. Table 13.4 shows the AUC classification results of using the top 2500 extracted features, as well as when using the optimally extracted feature set of up to 25,000 extracted features. The optimal number of features, out of the entire augmented 240K+ deep *VGG* feature set, were selected using a standard grid-search technique.

In Table 13.4 we see that using a feature selection scheme can increase performance slightly, by an average AUC gain of 2%. For most of the pathologies, the fixed feature selection representation achieves almost the maximum AUC result, ranking either as the first or second score. In terms of likelihood measurements, the average  $LR+$  of the fixed feature selection method is 5.00, as opposed to 4.89 and 4.78 for *VGG-L4* and *VGG-L5* features, respectively. The average  $LR-$  of the fixed feature selection method is 0.23, as compared with 0.26 and 0.25. This indicates an improvement over *VGG-L4* and *VGG-L5* as well.

**FIGURE 13.8**

ROCs of different examined pathologies.

Fig. 13.9 shows a comparative ROC curve analysis using a selected optimal feature set vs the individual layer of *VGG-L4*, for several pathologies. We note that in all examined cases, for most points on the curve, the feature selection curve is on top.

**Table 13.4** VGG-M-2048 CNN top ranked deep features representations: AUC metric classification performance. The descriptor dimensionality appears in parentheses

Descriptor	RPE	LPE	RCN	LCN	Cardio	MED	Healthy	Avg.
Selection (2500)	0.92	0.94	0.80	0.78	0.95	0.87	0.89	0.88
Selection (optimal) (22,000)	0.93	0.94	0.83	0.81	0.96	0.88	0.89	0.89

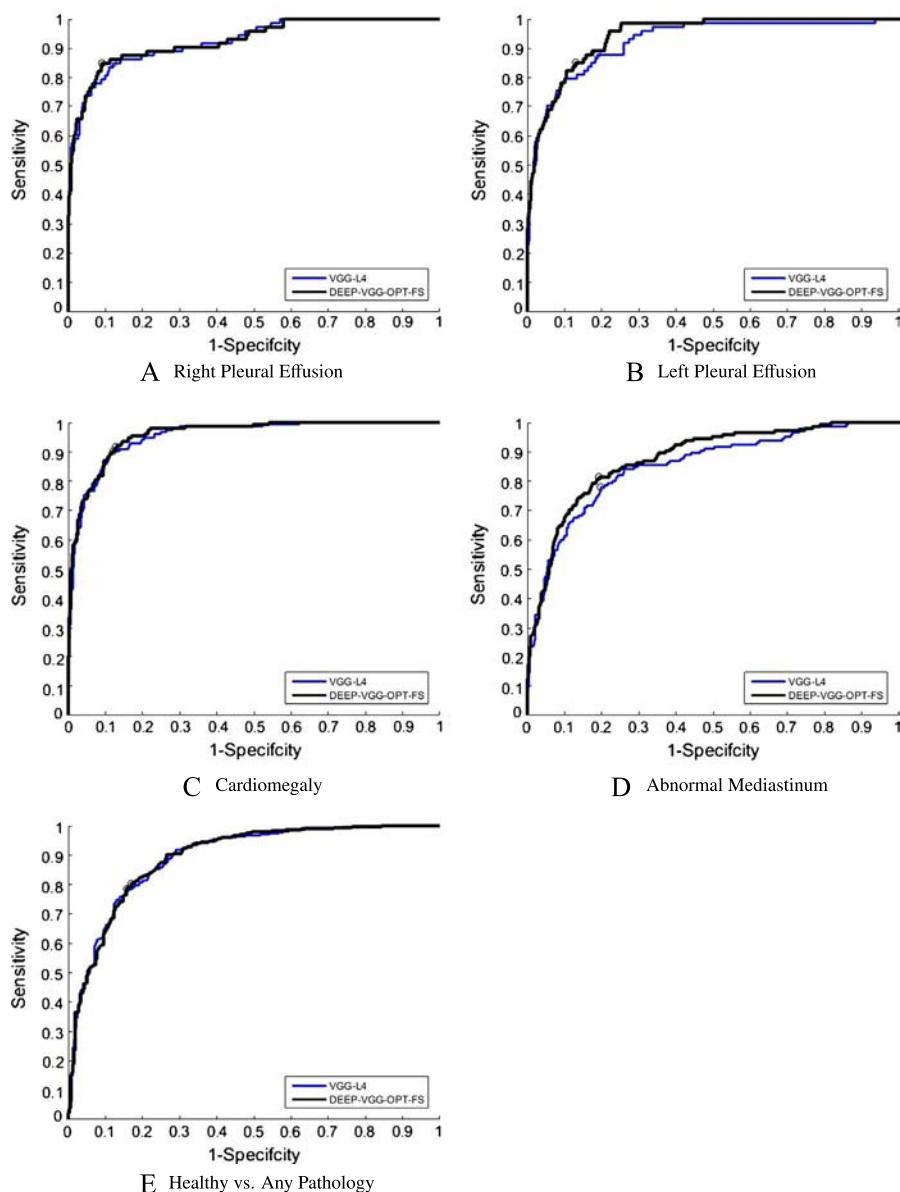
We obtain similar results, excluding the cardiomegaly case, using the fixed (2500) number of features.

[Fig. 13.10](#) presents a case study analysis for the *VGG-L4* feature representation using the SVM classifier. Four subjects are shown. For each one we compare the radiologist ground truth prediction with the classifier normalized score. The 4 examples shown represent a case of True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN), respectively. As this figure shows, it is often the case that images that are evident to the human observer are classified with high probability to the correct label, whereas cases that are difficult (noisy) for the human observer are the ones for which the automated system may misclassify as well.

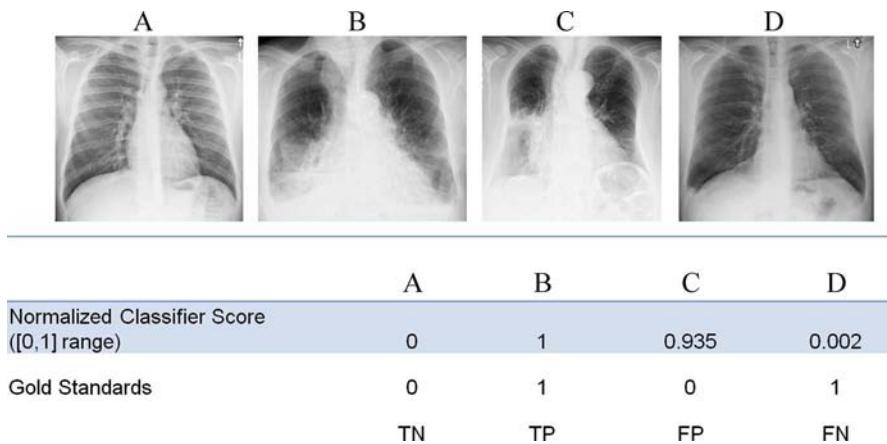
## 13.6 CONCLUSION

In this chapter we showed a collection of results that demonstrate the robustness and strengths of using the CNN representation for medical categorization tasks. We achieve strong results for medical image classification of various pathologies, which surpass relatively recent state-of-the-art results using BoVW. We explore features for the chest pathology categorization task. We use both classical as well as CNN-based features. We show results for categorization based on each feature set independently, as well as the result of using a feature selection stage prior to the categorization task.

How does the deep learning approach compare with the more classical approaches for medical image analysis? A classical approach for medical processing involves preprocessing of the image, segmentation, and landmark extraction, prior to feature extraction and an optional classification stage. As an example, we refer to a recently published work on the detection of pleural effusion [47]. The solution proposed is based on the use of expert-knowledge for the specific pathology. Specifically, segmentation and landmark localization are used based on which localized features are extracted and later classified. A numerical comparison is difficult as different datasets are used. Still, it is interesting to note that for a similar size dataset, with overall similar clinical conditions, the more classical approach in [47] achieves AUC of 0.9 and 0.85 for the correct categorization of RPE and LPE, and our presented deep learning

**FIGURE 13.9**

ROCs of different examined pathologies for the optimal number of selected features.

**FIGURE 13.10**

Right pleural effusion case study analysis using *VGG-L4* feature representation.

based representation scheme achieves similar results of AUC of 0.92 and 0.94, respectively. In the method presented here, the pleural-effusion pathology is just one of several tasks which are *simultaneously* categorized. Also, the use of segmentation and localization are computationally intensive and prone to error, whereas in the framework presented herein, the full image is input to the system with no segmentation necessary.

We conclude that the most informative feature set consists of a selection of features from the CNN layers. Using this selected set of features gives higher AUC values across all pathologies as well as for screening (healthy vs. pathological). Intuitively one could argue that the learned weights which constitute the deep feature layers are optimized to the images of the CNN training dataset and the task it is trained for, thus, one could imagine the optimal representation for each problem lies at an intermediate layer of the CNN but without knowing which layer in advance. Feature selection algorithms can assist us in this problem by picking the most significant deep features from the different layers, in an automated and robust process, while preserving and augmenting the classification performance. The presented approach is general and can be applied to many additional medical classification tasks.

---

## ACKNOWLEDGEMENTS

Part of this work was funded by the Ministry of Industry, Science and Development (Kamin program). Partial support was also given by INTEL Collaborative Research Institute for Computational Intelligence (ICRI-CI).

---

## REFERENCES

1. B. van Ginneken, L. Hogeweg, M. Prokop, Computer-aided diagnosis in chest radiography: beyond nodules, *Eur. J. Radiol.* 72 (2) (2009) 226–230.
2. G. Coppini, M. Miniati, M. Paterni, S. Monti, E.M. Ferdeghini, Computer-aided diagnosis of emphysema in COPD patients: neural-network-based analysis of lung shape in digital chest radiographs, *Med. Eng. Phys.* 29 (2) (2007) 76–86.
3. S. Katsuragawa, K. Doi, Computer-aided diagnosis in chest radiography, *Comput. Med. Imaging Graph.* 31 (4) (2007) 212–223.
4. J.M. Carrillo de Gea, G. Garcia-Mateos, Detection of normality/pathology on chest radiographs using LBP, in: Proceedings of BioInformatics, Valencia, Spain, January 20–23, 2010, pp. 167–172.
5. U. Avni, H. Greenspan, E. Konen, M. Sharon, J. Goldberger, X-ray categorization and retrieval on the organ and pathology level, using patch-based visual words, *IEEE Trans. Med. Imaging* 30 (3) (2011) 733–746.
6. G. Csurka, C.R. Dance, L. Fan, J. Willamowski, C. Bray, Visual categorization with bags of keypoints. Workshop on statistical learning in computer vision, in: ECCV, vol. 1, Prague, 2004, pp. 1–2.
7. Y. Lecun, Y. Bengio, G. Hinton, Deep learning, *Nature* 521 (2015) 436–444.
8. J. Dean, Large scale distributed deep networks, in: Advances in Neural Information Processing Systems, 2012, pp. 1223–1231.
9. K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, arXiv:1409.1556, 2014.
10. A.S. Razavian, A. Hosseini, J. Sullivan, S. Carlsson, CNN features off-the-shelf: an astounding baseline for recognition, in: IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2014, pp. 512–519.
11. M. Oquab, et al., Learning and transferring mid-level image representations using convolutional neural networks, in: 2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014, pp. 1717–1724.
12. J. Deng, et al., ImageNet: a large-scale hierarchical image database, in: 2009 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2009, pp. 248–255.
13. K. Chatfield, K. Simonyan, A. Vedaldi, A. Zisserman, Return of the devil in the details: delving deep into convolutional nets, arXiv:1405.3531, 2014.
14. M.D. Zeiler, R. Fergus, Visualizing and understanding convolutional networks, in: Computer Vision – ECCV 2014, Springer, 2014, pp. 818–833.
15. J. Donahue, et al., Decaf: a deep convolutional activation feature for generic visual recognition, arXiv:1310.1531, 2013.
16. P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, Y. LeCun, Overfeat: integrated recognition, localization and detection using convolutional networks, arXiv:1312.6229, 2013.
17. H. Greenspan, B. van-Ginneken, R.M. Summers, Deep learning in medical imaging: overview and future promise of an exciting new technique, *IEEE Trans. Med. Imaging* 35 (5) (2016) 1153–1159.
18. A. Oliva, A. Torralba, Modeling the shape of the scene: a holistic representation of the spatial envelope, *Int. J. Comput. Vis.* 42 (3) (2001) 145–175.
19. S. Lazebnik, C. Schmid, J. Ponce, Beyond bags of features: spatial pyramid matching for recognizing natural scene categories, *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.* 2 (2006) 2169–2178.

20. I. Fogel, D. Sagi, Gabor filters as texture discriminator, *Biol. Cybern.* 61 (2) (1989) 103–113.
21. R.M. Haralick, K. Shanmugam, I. Dinstein, Textural features for image classification, *IEEE Trans. Syst. Man Cybern.* 6 (10) (1973) 610–621.
22. M. Douze, H. Jégou, H. Sandhawalia, L. Amsaleg, C. Schmid, Evaluation of GIST descriptors for web-scale image search, in: Proceedings of the ACM International Conference on Image and Video Retrieval, Springer, 2009, pp. 19:1–19:8.
23. R. Raguram, C. Wu, J.M. Frahm, S. Lazebnik, Modeling and recognition of landmark image collections using iconic scene graphs, *Int. J. Comput. Vis.* 95 (3) (2011) 213–239.
24. D.G. Lowe, Distinctive image features from scale-invariant keypoints, *Int. J. Comput. Vis.* 60 (2) (2004) 91–110.
25. A. Bosch, A. Zisserman, X. Munoz, Representing shape with a spatial pyramid kernel, in: Proceedings of the 6th ACM International Conference on Image and Video Retrieval, Springer, 2007, pp. 401–408.
26. Y. Bai, L. Guo, L. Jin, Q. Huang, A novel feature extraction method using pyramid histogram of orientation gradients for smile recognition, in: Proceedings of the 16th IEEE International Conference on Image Processing, Springer, 2009, pp. 3269–3272.
27. B. Ioan, G. Alexandru, Directional features for automatic tumor classification of mammogram images, *Biomed. Signal Process. Control* 6 (4) (2011) 370–378.
28. A. Karahaliou, S. Skiadopoulos, I. Boniatis, P. Sakellaropoulos, E. Likaki, G. Panayiotakis, L. Costaridou, Texture analysis of tissue surrounding microcalcifications on mammograms for breast cancer diagnosis, *Br. J. Radiol.* 80 (956) (2007) 648–656.
29. Y. Chi, J. Zhou, S.K. Venkatesh, Q. Tian, J. Liu, Content-based image retrieval of multi-phase CT images for focal liver lesion characterization, *Med. Phys.* 40 (10) (2013).
30. Sebastian Haas, René Donner, Andreas Burner, Markus Holzer, Georg Langs, Superpixel-based interest points for effective bags of visual words medical image retrieval, in: Proceedings of the Second MICCAI International Conference on Medical Content-Based Retrieval for Clinical Decision Support, 2012, pp. 58–68.
31. W. Yang, Z. Lu, M. Yu, M. Huang, Q. Feng, W. Chen, Content-based retrieval of focal liver lesions using bag-of-visual-words representations of single- and multiphase contrast-enhanced CT images, *J. Digit. Imaging* 25 (6) (2012) 708–719.
32. J. Wang, Y. Li, Y. Zhang, H. Xie, C. Wang, Bag-of-features based classification of breast parenchymal tissue in the mammogram via jointly selecting and weighting visual words, in: Proceedings of the International Conference on Image and Graphics, Hefei, P.R. China, 2011, pp. 622–627.
33. M. Huang, W. Yang, M. Yu, Q. Feng, W. Chen, Retrieval of brain tumors with region-specific bag-of-visual-words representations in contrast-enhanced MRI images, *Comput. Math. Methods Med.* (2012) 280538.
34. I. Diamant, J. Goldberger, H. Greenspan, Visual words based approach for tissue classification in mammograms, *Proc. SPIE Med. Imaging* 8670 (2013) 867021.
35. I. Diamant, A. Hoogi, C.F. Beaulieu, M. Safdari, E. Klang, M. Amitai, H. Greenspan, D.L. Rubin, Improved patch based automated liver lesion classification by separate analysis of the interior and boundary regions, *J. Biomed. Health Inf.* (2015), <http://dx.doi.org/10.1109/JBHI.2015.2478255>.
36. I. Diamant, E. Klang, M. Amitai, J. Goldberger, H. Greenspan, Multi-phase liver lesions classification using relevant visual words based on mutual information, in: IEEE Int. Symposium on Biomedical Imaging (ISBI), Brooklyn, NY, USA, 2015, pp. 407–410.

37. A. Krizhevsky, I. Sutskever, G. Hinton, ImageNet classification with deep convolutional neural networks, in: Advances in Neural Information Processing Systems, 2012, pp. 1097–1105.
38. Y. Bar, I. Diamant, L. Wolf, H. Greenspan, Deep learning with non-medical training used for chest pathology identification, Proc. SPIE Med. Imaging (2015) 94140V.
39. Y. Bar, I. Diamant, L. Wolf, S. Lieberman, E. Konen, H. Greenspan, Chest pathology detection using deep learning with non-medical training, in: Proceedings of IEEE International Symposium on Biomedical Imaging (ISBI), 2015.
40. Y. Bar, I. Diamant, L. Wolf, S. Lieberman, E. Konen, H. Greenspan, Chest pathology identification using deep feature selection with non-medical training, Comput. Methods Biomed. Eng. (2016) 1–5.
41. B. van Ginneken, A.A. Setio, C. Jacobs, F. Ciompi, Off-the-shelf convolutional neural network features for pulmonary nodule detection in computed tomography scans, in: Proceedings of IEEE International Symposium on Biomedical Imaging (ISBI), 2015, pp. 286–289.
42. Z. Zhao, F. Morstatter, S. Sharma, S. Alelyani, A. Anand, H. Liu, Advancing feature selection research, ASU Feature Selection Repository, 2010, pp. 1–28.
43. T.M. Cover, J.A. Thomas, Elements of Information Theory, John Wiley & Sons, 2012.
44. J.R. Thornbury, D.G. Fryback, W. Edwards, Likelihood ratios as a measure of the diagnostic usefulness of excretory urogram information 1, Radiology 114 (3) (1975) 561–565.
45. S.G. Pauker, J.P. Kassirer, Therapeutic decision making: a cost–benefit analysis, N. Engl. J. Med. 293 (5) (1975) 229–234.
46. J. Yosinski, J. Clune, Y. Bengio, H. Lipson, How transferable are features in deep neural networks?, in: Advances in Neural Information Processing Systems, 2014, pp. 3320–3328.
47. P. Maduskar, R. Philipsen, J. Melendez, E. Scholten, D. Chanda, H. Ayles, C.I. Sánchez, B. van Ginneken, Automatic detection of pleural effusion in chest radiographs, Med. Image Anal. 28 (2016) 22–32.

# Deep Learning Models for Classifying Mammogram Exams Containing Unregistered Multi-View Images and Segmentation Maps of Lesions<sup>1</sup>

# 14

Gustavo Carneiro\*, Jacinto Nascimento<sup>†</sup>, Andrew P. Bradley<sup>‡</sup>

University of Adelaide, Adelaide, SA, Australia\* Instituto Superior Técnico, Lisbon, Portugal<sup>†</sup>

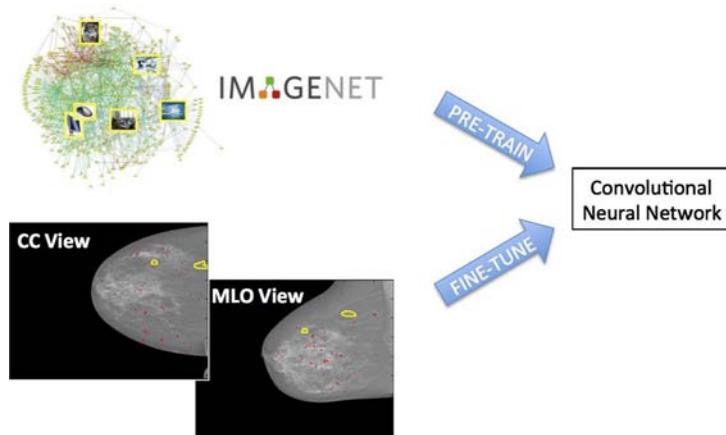
University of Queensland, Brisbane, QLD, Australia<sup>‡</sup>

## CHAPTER OUTLINE

14.1	Introduction .....	322
14.2	Literature Review .....	324
14.3	Methodology .....	326
14.3.1	Deep Learning Model .....	326
14.4	Materials and Methods .....	329
14.5	Results .....	331
14.6	Discussion .....	334
14.7	Conclusion .....	334
	Acknowledgements .....	335
	References .....	335
	Note .....	339

## CHAPTER POINTS

- Shows how to fine-tune deep learning models, pre-trained with computer vision databases, for the analysis of mammograms
- Demonstrates how high-level deep learning model features can be used in multi-view mammogram classification problems without pre-registering the mammograms

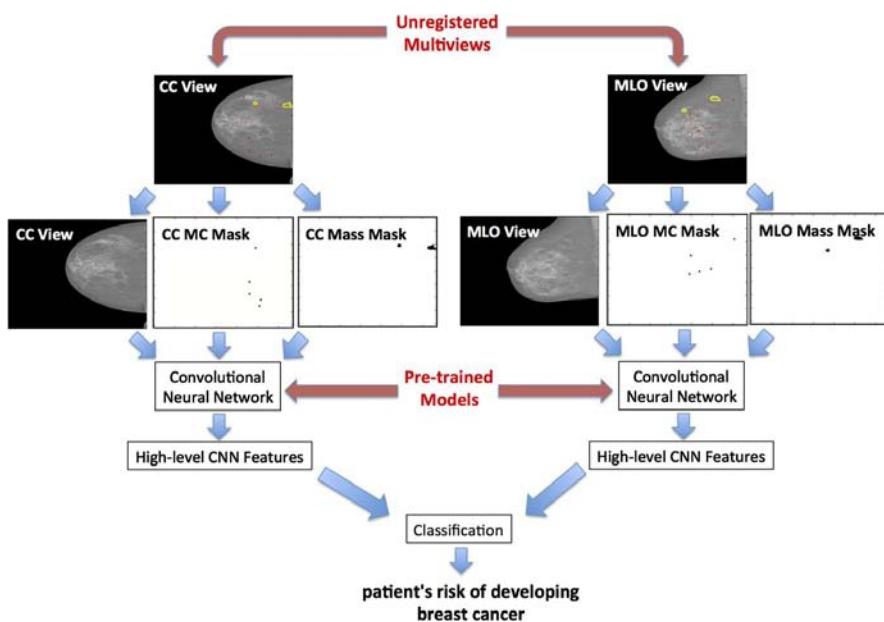
**FIGURE 14.1**

The first goal of the paper is to show how to fine-tune deep learning models, pre-trained with the computer vision database ImageNet [2], for the joint analysis of the cranio-caudal (CC) and medio-lateral oblique (MLO) mammography views. Note that the yellow annotations denote breast masses and red annotations represent micro-calcifications.

## 14.1 INTRODUCTION

According to recent statistical data published by the World Health Organisation (WHO), breast cancer accounts for 23% of all cancer related cases and 14% of all cancer related deaths among women worldwide [3]. The early detection of breast cancer in asymptomatic women using breast screening mammography is currently the most effective tool to reduce the morbidity and mortality associated with breast cancer [4]. A breast screening exam typically consists of two mammography views of each breast: the medio-lateral oblique view (MLO) and the cranio-caudal view (CC); see Figs. 14.1–14.2 for examples of these two views. One of the stages present in the analysis of these mammography views involves the identification and classification of breast lesions, such as breast masses and micro-calcifications (MC) represented by yellow and red annotations, respectively, in Figs. 14.1 and 14.2. This identification and classification is usually performed manually by a radiologist, and a recent assessment of this manual analysis indicates a sensitivity of 84% and a specificity of 91% in the classification of breast cancer [5]. These figures can be improved with the analysis of the mammography views by a second reader: either a radiologist or a computer-aided diagnosis (CAD) system [5]. Therefore, the use of CAD systems as second readers can have a significant impact in breast screening mammography.

Current state-of-the-art CAD systems that can analyze a mammography exam work in delimited stages [5–7]: lesion detection, lesion segmentation, and lesion classification. The main challenges faced by such systems are related to the low signal-to-noise ratio present in the imaging of the lesion, the lack of a consistent

**FIGURE 14.2**

The second goal of the paper is to demonstrate how high-level deep learning model features can be used in multi-view mammogram classification problems without pre-registering the mammograms. The meaning of the yellow and red annotations are the same as in Fig. 14.1.

location, shape and appearance of lesions, and the analysis of each lesion independently of other lesions or the whole mammogram. The detection of lesions usually follow a two-step process that first identifies a large number of lesion candidates that are then selected with the goal of reducing false positives while keeping the true positives [8–15]. Lesion segmentation methods are generally based on global/local energy minimization models that work on a continuous or discrete space [16–18]. The final stage consists of the classification of the segmented lesions based on typical machine learning classifiers that use as input handcrafted features extracted from the image region containing the detected lesion and the respective segmentation map [19–21]. The state-of-the-art binary classification of breast lesions into benign or malignant [22, 23] produces an area under the receiver operating characteristic (ROC) curve between [0.9, 0.95]. More similar to our approach, the multi-modal analysis that takes lesions imaged from several modalities (e.g., mammograms and sonograms) have been shown to improve the average performance of radiologists [24]. Note that these handcrafted features do not guarantee optimality with respect to the classification goal, and the isolated processing of each lesion without looking at the other parts of the exam may ignore crucial contextual information that could help the classification of the whole exam.

In this chapter, we propose a methodology that can analyze a mammography exam in a holistic manner. Specifically, we introduce the design and implementation of a deep learning model [25,26] that takes as input the two mammography views (CC and MLO) and all detected breast masses and micro-calcifications, and produce an output consisting of a three-class classification of the whole exam: negative (or normal), benign or malignant findings. The challenges present in the development of such deep learning model are: (i) the high capacity of such model can only be robustly estimated with the use of large annotated training sets, and (ii) the holistic analysis may require the CC and MLO views to be registered in order to allow the alignment of lesions between these two views. Given that publicly available mammogram datasets do not contain enough annotated samples to robustly train deep learning models, we propose the use of transfer learning [27], where a model is pre-trained with a large annotated computer vision dataset [2], containing typical pictures taken from digital cameras, and fine-tuned with the relatively smaller mammogram datasets. Furthermore, the registration of the CC and MLO views of a mammography exam is a challenging task given the difficulty in estimating the non-rigid deformations that can align these two views, so we propose the classification from the deep learning features, where the hypothesis is that the high-level nature of these features will reduce the need for a low-level matching of the input data [28]. Finally, compared to the previous state-of-the-art in the field, deep learning model can extract features that are automatically learned (as opposed to the previously proposed handcrafted features) using objective functions formulated based on the classification problem. We test our approach on two publicly available datasets (InBreast [29] and DDSM [30]), and results show that our approach produces a volume under ROC surface of over 0.9 and an area under ROC curve (for a two-class problem: benign and malignant) of over 0.9. The results provide evidence that our method can produce state-of-the-art classification results using a new holistic way of addressing medical image analysis problems.

---

## 14.2 LITERATURE REVIEW

Deep learning has been one of the most studied topics in the fields of computer vision and machine learning [25] for at least three decades. The recent availability of large annotated training sets [2] combined with a competent use of graphics processing units (allowing fast training processes) has enabled the development of classification systems that are significantly more accurate than more traditional machine learning methods [26,31–33]. The impact in medical image analysis has been relatively smaller, but also significant [34,35]. Deep learning has several advantages, compared with traditional machine learning methods [36], such as: features of different abstraction levels are automatically learned using high-level classification objective functions [28]; and methodologies can be designed “end-to-end”, where the system can learn how to extract image features, detect and segment visual objects of interest and classify the scene using a unified classification model [37]. The major

challenge present in deep learning models is the extremely high number of parameters to estimate during the training process, which requires an equally large annotated training set to enable a robust training process. This challenge is particularly critical in medical image analysis (MIA) applications due to the limited availability of large annotated training set. In fact, the largest MIA datasets typically have in the order of a few thousands of samples, which is generally considered to be not enough for a robust training of a deep learning model. The initial successful MIA applications have been achieved exactly with problems that contain large annotated training sets, such as the mitosis detection [34] and lymph node detection [35]. However, MIA problems that have limited training sets have been generally tackled with the help of regularization methods, such as unsupervised training [38–41]. The use of registered multi-view input data has also been tested with deep auto-encoders [42,43], which is similar to our methodology, except that our multi-view data is not aligned.

Recently, there has been considerable interest in the development of deep learning methods for the analysis of mammograms, where this analysis can be divided into three stages [5]: (i) detection of lesions (i.e., masses and micro-calcifications), (ii) segmentation of the detected lesions from the first stage, and (iii) classification of the lesions based on texture and shape features extracted from the segmented lesions. The problem of mass detection has been traditionally addressed by classical image processing methods for initial candidate generation, followed by a cascade of machine learning techniques to eliminate false positives [5]. The use of a cascade of deep learning models for mass detection essentially follows the same approach, with the exception that it does not rely on classical image processing methods to generate initial candidates [44]. The use of deep learning methods for lesion segmentation has been explored in different ways. For instance, a straightforward deep learning model receives the image at the input and produces a binary segmentation map at the output [45,46], which only works if the annotated training set is relatively large [30]. When the annotated training set is small [21], Dhungel et al. [47–49] have proposed a method that combines a conditional random field with several deep learning potential functions, where the idea is that this combination will compensate for the small annotated sets used in the training process. Finally, the last stage in the analysis, i.e., lesion classification, has also been addressed with deep learning methods, with a direct classification of the detected and segmented lesions from the first and second stages of the analysis [50–52].

Deep learning is also allowing the development of methods that can analyze mammograms in a more holistic manner, like the work proposed in this chapter, which represents a clear divergence from the traditional 3-stage analysis process [5] mentioned above. For example, the risk of developing breast cancer can be assessed with deep learning classifiers that score breast density and texture [53,54]. Finally, Qiu et al. [55] propose a method that estimates the risk of developing breast cancer from a normal mammogram. We expect that such deep learning-based methods that receive a mammography exam at the input and produce either a diagnosis or prognosis result will become the mainstream of future methodologies.

## 14.3 METHODOLOGY

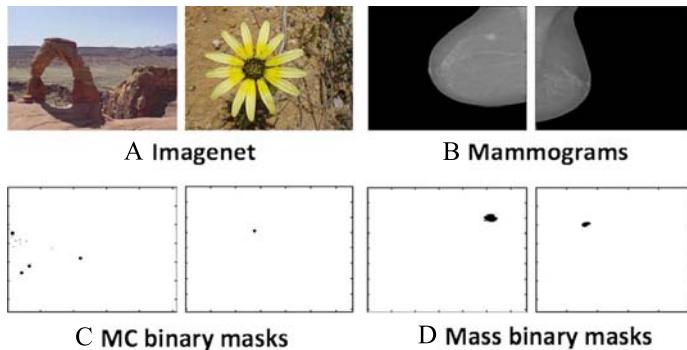
For the training and testing of our methodology, we have the following dataset:  $\mathcal{D} = \{(\mathbf{x}^{(p,b)}, \mathbf{c}^{(p,b)}, \mathbf{m}^{(p,b)}, \mathbf{y}^{(p,b)})\}_{p \in \{1, \dots, P\}, b \in \{\text{left, right}\}}$ , with  $\mathbf{x} = \{\mathbf{x}_{\text{CC}}, \mathbf{x}_{\text{MLO}}\}$  denoting the mammography views CC and MLO, where  $\mathbf{x}_{\text{CC}}, \mathbf{x}_{\text{MLO}} : \Omega \rightarrow \mathbb{R}$  and  $\Omega$  denotes the image lattice,  $\mathbf{c} = \{\mathbf{c}_{\text{CC}}, \mathbf{c}_{\text{MLO}}\}$  representing the micro-calcifications (MC) segmentation in each view with  $\mathbf{c}_{\text{CC}}, \mathbf{c}_{\text{MLO}} : \Omega \rightarrow \{0, 1\}$ ,  $\mathbf{m} = \{\mathbf{m}_{\text{CC}}, \mathbf{m}_{\text{MLO}}\}$  denoting the mass segmentation in each view with  $\mathbf{m}_{\text{CC}}, \mathbf{m}_{\text{MLO}} : \Omega \rightarrow \{0, 1\}$ ,  $\mathbf{y} \in \mathcal{Y} = \{0, 1\}^C$  being the BI-RADS classification with  $C$  classes,  $p \in \{1, \dots, P\}$  indexing the patients, and  $b \in \{\text{left, right}\}$  indexing the patient's left and right breasts (each patient's breast is denoted as a case because they can have different BI-RADS scores). There are six possible BI-RADS classes: 1, negative; 2, benign finding(s); 3, probably benign; 4, suspicious abnormality; 5, highly suggestive of malignancy; and 6, proven malignancy. However, the datasets available for this research only contain limited amounts of training data per class, as shown in Fig. 14.6, so we propose the following three-class division of the original classes: negative, denoted by  $\mathbf{y} = [1, 0, 0]^\top$ , when BI-RADS = 1; benign, represented by  $\mathbf{y} = [0, 1, 0]^\top$ , with BI-RADS  $\in \{2, 3\}$ ; and malignant, denoted by  $\mathbf{y} = [0, 0, 1]^\top$ , when BI-RADS  $\in \{4, 5, 6\}$ . The dataset of non-mammography images, used for pre-training the deep learning model, is represented by  $\tilde{\mathcal{D}} = \{(\tilde{\mathbf{x}}^{(n)}, \tilde{\mathbf{y}}^{(n)})\}_n$ , with  $\tilde{\mathbf{x}} : \Omega \rightarrow \mathbb{R}$  and  $\tilde{\mathbf{y}} \in \tilde{\mathcal{Y}} = \{0, 1\}^{\tilde{C}}$ , where  $\tilde{C}$  represents the cardinality of the set of classes in the dataset  $\tilde{\mathcal{D}}$ . Fig. 14.3 shows examples of the non-mammographic images in  $\tilde{\mathcal{D}}$ , and also the mammography views plus their respective binary MC and mass segmentation masks in  $\mathcal{D}$ .

### 14.3.1 DEEP LEARNING MODEL

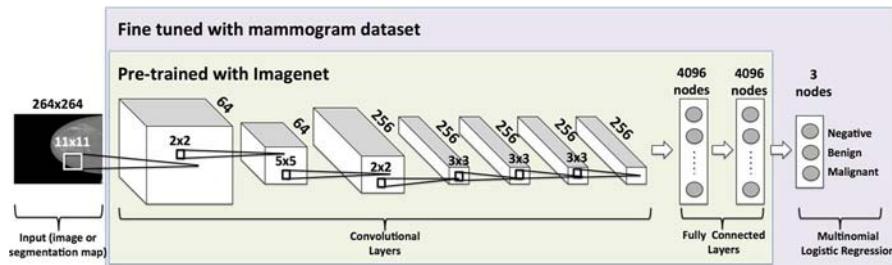
The deep learning model explored in this work consists of the convolutional neural network (CNN), which is represented by  $f : \mathcal{X} \rightarrow \mathcal{Y}$  ( $\mathcal{X}$  denotes the image or binary segmentation map spaces while  $\mathcal{Y}$  represents the space of classification vectors):

$$f(\mathbf{x}, \theta) = f_{\text{out}} \circ f_{fc} \circ h_L \circ g_L \circ f_L \circ \dots \circ h_1 \circ g_1 \circ f_1(\mathbf{x}), \quad (14.1)$$

where  $\circ$  represents the composition operator,  $\{f_i(\cdot)\}_{i=1}^L$  denotes the convolutional layers,  $\theta$  represents the model parameters formed by the input weight matrices  $\mathbf{W}_l \in \mathbb{R}^{k_l \times k_l \times n_l \times n_{l-1}}$  and bias vectors  $\mathbf{b}_l \in \mathbb{R}^{n_l}$  for each layer  $l \in \{1, \dots, L\}$ , with  $k_l \times k_l$  representing the filter size of the  $n_l$  filters in layer  $l$  that has  $n_{l-1}$  input channels,  $g_l(\cdot)$  is a nonlinear activation layer,  $h_l(\cdot)$  is a sub-sampling layer,  $f_{fc}$  denotes the set of fully-connected layers  $\{\mathbf{W}_{fc,k}\}_{k=1}^K$  (with  $\mathbf{W}_{fc,k} \in \mathbb{R}^{n_{fc,k-1} \times n_{fc,k}}$  representing the connections from fully connected layer  $k-1$  to  $k$ ) and biases  $\{\mathbf{b}_{fc,k}\}_{k=1}^K$  (with  $\mathbf{b} \in \mathbb{R}^{n_{fc,k}}$ ) that are also part of the model parameters  $\theta$ , and  $f_{\text{out}}$  is a multinomial logistic regression layer [26] that contains weights  $\mathbf{W}_{\text{out}} \in \mathbb{R}^{n_{fc,K} \times C}$  and bias  $\mathbf{b}_{\text{out}} \in \mathbb{R}^C$ , which also belong to  $\theta$  (Fig. 14.4 shows a visual description of this model). The convolutional layer is defined by

**FIGURE 14.3**

The images in (A) represent samples from the dataset  $\tilde{\mathcal{D}}$ , used for pre-training the deep learning model, while (B)–(D) display training images and binary maps of micro-calcifications (MC) and masses from dataset  $\mathcal{D}$ .

**FIGURE 14.4**

Visualization of the single view CNN model used in this work, containing  $L = 5$  stages of convolutional layers,  $K = 2$  stages of fully connected layers and one final layer containing the softmax layer.

$$\mathbf{F}_l = f_l(\mathbf{x}_{l-1}) = \mathbf{W}_l \star \mathbf{X}_{l-1}, \quad (14.2)$$

where the bias term  $\mathbf{b}_l$  is excluded to simplify the equation and we are abusing the notation by representing the convolution of  $n_{l-1}$  channels of input  $\mathbf{X}_{l-1} = [\mathbf{x}_{l-1,1}, \dots, \mathbf{x}_{l-1,n_{l-1}}]$  with the  $n_l$  filters of matrix  $\mathbf{W}_l$ , with  $\star$  denoting the convolution operator. The input  $\mathbf{X}_{l-1}$  of (14.2) is obtained from the activation (e.g., logistic or rectified linear [26]) and sub-sampling (e.g., the mean or max pooling functions [26]) of the preceding layer by  $\mathbf{X}_{l-1} = h_{l-1}(g_{l-1}(f_{l-1}(\mathbf{X}_{l-2})))$ , where  $\mathbf{X}_0$  represents the input mammogram  $\mathbf{x}$  or segmentation maps  $\mathbf{c}$  or  $\mathbf{m}$ . The output from (14.2) is  $\mathbf{F}_l = [\mathbf{f}_{l,1}, \dots, \mathbf{f}_{l,n_l}]$ , which is a volume containing  $n_l$  pre-activation matrices. The  $L$  convolutional layers are followed by a sequence of fully connected layers that vectorize the input volume  $\mathbf{X}_L$  into  $\mathbf{x}_L \in \mathbb{R}^{|\mathbf{x}_L|}$  (where  $|\mathbf{x}_L|$  denotes the length of the

vector  $\mathbf{x}_L$ ) and apply a couple of linear transforms [26]:

$$\mathbf{f}_{fc} = f_{fc}(\mathbf{X}_L) = (\mathbf{W}_{fc,2} (\mathbf{W}_{fc,1} \mathbf{x}_L + \mathbf{b}_{fc,1}) + \mathbf{b}_{fc,2}), \quad (14.3)$$

where the output is a vector  $\mathbf{f}_{fc} \in \mathbb{R}^{n_{fc,2}}$ . Finally, these fully connected layers are followed by a classification layer defined by a softmax function over a linearly transformed input, as follows [26]:

$$\mathbf{f}_{out} = f_{out}(\mathbf{f}_{fc}) = \text{softmax}(\mathbf{W}_{out} \mathbf{f}_{fc} + \mathbf{b}_{out}), \quad (14.4)$$

where  $\text{softmax}(\mathbf{z}) = \frac{e^{\mathbf{z}}}{\sum_j e^{\mathbf{z}(j)}}$ , and  $\mathbf{f}_{out} \in [0, 1]^C$  represent the output from the inference process that takes  $\mathbf{x}$  as the input (recall that the input can be either a mammogram or a segmentation map of a micro-calcification or a mass), with  $C$  representing the number of output classes.

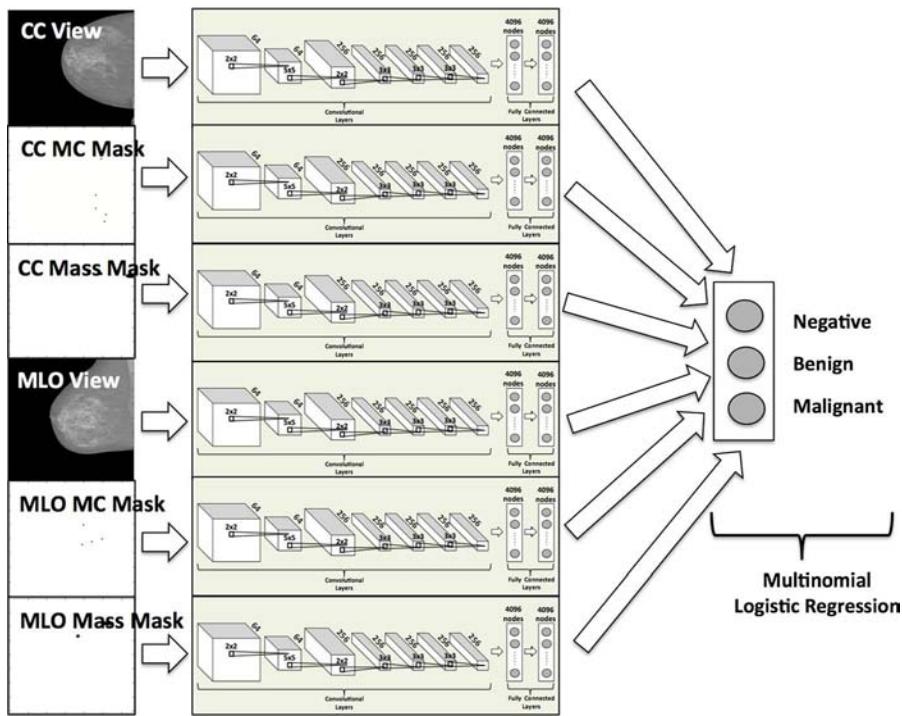
Finally, estimating  $\theta$  in (14.1) involves a training process that is carried out with stochastic gradient descent to minimize the cross-entropy loss [26] over the training set, as follows [26]:

$$\ell(\theta) = \frac{1}{N} \sum_{i=1}^N \mathbf{y}_i \log \mathbf{f}_{out,i}^\top, \quad (14.5)$$

where  $N$  denotes the number of cases available for training, which are indexed by  $i$ .

The training of the model in (14.1) comprises two steps: a pre-training stage using the dataset of non-mammography images  $\tilde{\mathcal{D}}$  and a fine-tuning stage that relies on the dataset of mammography images and segmentation maps  $\mathcal{D}$ . The pre-training process produces the model  $\tilde{\mathbf{y}}^* = f(\tilde{\mathbf{x}}; \tilde{\theta})$ , where  $\tilde{\theta} = [\tilde{\theta}_1, \dots, \tilde{\theta}_L, \tilde{\theta}_{fc,1}, \tilde{\theta}_{fc,2}, \tilde{\theta}_{out}]$  with  $\tilde{\theta}_l = [\tilde{\mathbf{W}}_l, \tilde{\mathbf{b}}_l]$  denoting the parameters of the convolutional layer  $l \in \{1, \dots, L\}$ ,  $\tilde{\theta}_{fc,k} = [\tilde{\mathbf{W}}_{fc,k}, \tilde{\mathbf{b}}_{fc,k}]$  representing the parameters of the fully connected layer  $k \in \{1, 2\}$ , and  $\tilde{\theta}_{out} = [\tilde{\mathbf{W}}_{out}, \tilde{\mathbf{b}}_{out}]$  denoting the parameters of the softmax layer. This pre-training is carried out by minimizing the cross-entropy loss in (14.5) with the  $\tilde{C}$  classes present in the dataset  $\tilde{\mathcal{D}}$ . The fine-tuning process takes a subset of  $\tilde{\theta}$  comprising  $[\tilde{\theta}_1, \dots, \tilde{\theta}_L, \tilde{\theta}_{fc,1}, \tilde{\theta}_{fc,2}]$  (i.e., all parameters except for  $\tilde{\theta}_{out}$ ) to initialize the new training parameters  $\theta = [\theta_1, \dots, \theta_L, \theta_{fc,1}, \theta_{fc,2}, \theta_{out}]$ , where  $\theta_{out}$  is initialized with random values, and all parameters in  $\theta$  are re-trained to minimize the cross-entropy loss in (14.5) with the  $C$  classes in  $\mathcal{D}$  (see Fig. 14.4). Recently published results [27] have shown that such fine-tuning process depends on the use of a large number of pre-trained layers, which explains why we initialize almost all parameters (except for  $\theta_{out}$ ) with the values estimated from the pre-training process. This fine-tuning shall produce six models: (i) MLO image model  $\mathbf{y} = f(\mathbf{x}_{MLO}; \theta_{MLO,im})$ , (ii) CC image model  $\mathbf{y} = f(\mathbf{x}_{CC}; \theta_{CC,im})$ , (iii) MLO MC segmentation map model  $\mathbf{y} = f(\mathbf{c}_{MLO}; \theta_{MLO,mc})$ , (iv) CC MC segmentation map model  $\mathbf{y} = f(\mathbf{c}_{CC}; \theta_{CC,mc})$ , (v) MLO mass segmentation map model  $\mathbf{y} = f(\mathbf{m}_{MLO}; \theta_{MLO,ma})$ , and (vi) CC mass segmentation map model  $\mathbf{y} = f(\mathbf{m}_{CC}; \theta_{CC,ma})$ .

Finally, the multi-view model combines the six models produced from the fine-tuning process by concatenating the features from the last fully connected layer of

**FIGURE 14.5**

Visualization of the multi-view model with the responses from the single view CNN models (see Fig. 14.4) that are connected to a classification layer.

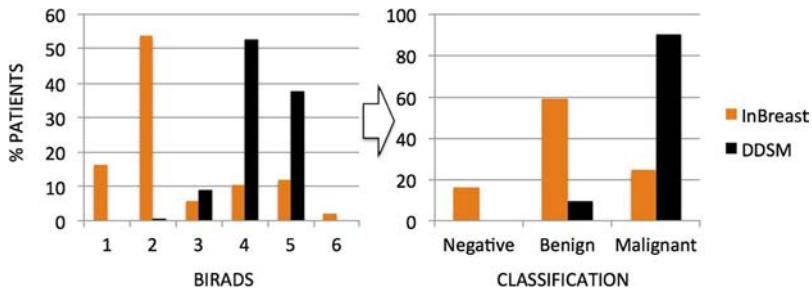
all six models, represented by  $\{\mathbf{f}_{fc,i}\}_{i \in \{MLO,im,CC,MLO,im,MLO,mc,CC,mc,MLO,ma,CC,ma\}}$ , and training a single multinomial logistic regression layer using those inputs (Fig. 14.5). This multi-view model is represented by:

$$\mathbf{f}_{out,mv} = softmax(\mathbf{W}_{out,mv}[\mathbf{f}_{fc,i}]_{i \in \{MLO,im,CC,MLO,im,MLO,mc,CC,mc,MLO,ma,CC,ma\}} + \mathbf{b}_{out,mv}), \quad (14.6)$$

and trained by minimizing the cross-entropy loss in (14.5) with the  $C$  classes in  $\mathcal{D}$ , where  $\theta_{mv} = [\mathbf{W}_{out,mv}, \mathbf{b}_{out,mv}]$  is randomly initialized in this multi-view training.

## 14.4 MATERIALS AND METHODS

For the experiments below, we use two mammogram datasets that are publicly available: InBreast [29] and DDSM [30]. The InBreast [29] dataset contains 115 patients, where there are around four images per patients, amounting to 410 images. InBreast

**FIGURE 14.6**

Distribution of BI-RADS (left) and negative, benign, and malignant classes (right) for the cases in InBreast (orange) and DDSM (black).

does not come with a suggested division of training and testing sets, so our experimental results are based on a five-fold cross-validation, where each fold uses a division of 90 patients for training and 25 patients for testing. The DDSM [30] dataset contains 172 patients, each having around four images, which results in 680 images. This dataset is formed by merging the original micro-calcification and mass datasets, but removing the overlapping cases that are available from the training set of mass and testing set of micro-calcification, and vice versa. We use the suggested division of training and testing sets for DDSM [30], containing 86 patients for training and 86 for testing. It is important to notice that the distributions of BI-RADS and, consequently the negative, benign, and malignant classes in InBreast and DDSM are quite different, as shown in Fig. 14.6. In particular, InBreast tries to keep the percentage of negative (i.e., normal) and benign cases at a higher level than the malignant cases, while DDSM has a much larger percentage of malignant cases, compared to benign and negative cases.

The CC and MLO mammography views are pre-processed with local contrast normalization, which is followed by Otsu's segmentation [56] that crops out the image region containing the background. The remaining image is scaled (using bi-cubic interpolation) to have size  $264 \times 264$ . Furthermore, a simple algorithm is run in order to flip the mammograms such that the pectoral muscle always lies on the right-hand side of the image. The manual binary segmentation maps representing the micro-calcification and mass present in a mammography view uses the same geometric transformations applied to their respective views (i.e., the cropping, scaling and flipping). If no micro-calcification and mass is present in a particular view, then we use a  $264 \times 264$  image filled with zeros (i.e., a blank image). Fig. 14.9 shows some samples of the pre-processed mammography views and their respective segmentation maps.

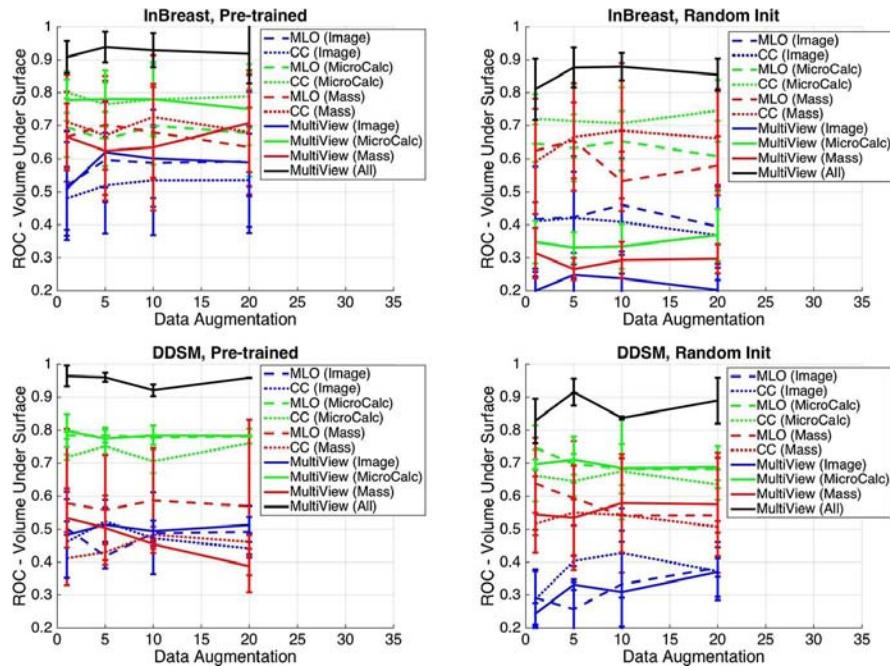
The base CNN model is based on Chatfield et al.'s CNN-F model [57], which is a simplified version of AlexNet [26], containing fewer filters. Fig. 14.4 shows the details of the CNN-F model, where the input image has size  $264 \times 264$ , the first convolutional stage as 64  $11 \times 11$  filters and a max-pooling that sub-samples the

input by 2, the second convolutional stage has 256  $5 \times 5$  filters and a max-pooling that sub-samples the input by 2, the third, fourth, and fifth convolutional stages have 256  $3 \times 3$  filters (each) with no sub-sampling, the first and second fully connected stages have 4096 nodes (each), and the multinomial logistic regression stage contains softmax layer with three nodes. We use the CNN-F model that Chatfield et al. [57] have pre-trained with ImageNet [2] (1K visual classes, 1.2M training, 50K validation, and 100K test images). The fine-tuning process consists of replacing the multinomial logistic regression stage at the end by a new layer that has three classes (negative, benign, and malignant) and train it for the CC and MLO views, the micro-calcification segmentation maps of the CC and MLO views, and the mass segmentation maps of the CC and MLO views (see Fig. 14.4). The multi-view model is built by concatenating the 4096-dimensional feature vectors available from the second fully connected stages of the six models (forming a 24,576-dimensional feature vector) and training a single multinomial logistic regression with three nodes (see Fig. 14.5). This two-stage training, comprising pre-training and fine-tuning, can be seen as a regularization that helps the generalization ability of the model. As a result, we can compare such two-stage training to other forms of regularization, such as data augmentation [26], which is obtained by applying random geometric transformations to the original training images in order to generate new artificial training samples. We compare our proposed two-stage training with data augmentation with an experiment that uses the CNN-F structure defined above without pre-training, which means that the training for the parameter  $\theta$  in (14.1) is randomly initialized using an unbiased Gaussian with standard deviation 0.001, and run with data augmentation by adding 5, 10, and 20 new samples per training image. In this data augmentation training, each original training image is randomly cropped from the top-left and bottom-right corners within a range of [1, 10] pixels from the original corners. This data augmentation is also used in the two-stage training process in order to verify if the combination of two regularization methods can improve the generalization of the CNN-F model. In all training processes, the learning rate is fixed at 0.001 and momentum is 0.9.

Classification accuracy is measured in two ways. For a three-class problem, with classes negative, benign, and malignant, the accuracy is measured with the volume under ROC surface (VUS) [58]. The two-class problem, with classes benign and malignant, is assessed by the area under ROC curve (AUC), where it is assumed that all cases contain at least one finding (a micro-calcification or a mass).

## 14.5 RESULTS

The VUS as a function of the data augmentation (varying in the range {0, 5, 10, 20}) for the test sets of InBreast and DDSM are shown in Fig. 14.7. For InBreast, the results are shown with the average and standard deviation of 5-fold cross-validation and the two breasts, and for DDSM, results are based on the average and standard deviation of the two breasts. Also note that in Fig. 14.7 we show the results for the MLO and CC views, and with each isolate input (image and segmentation maps) in

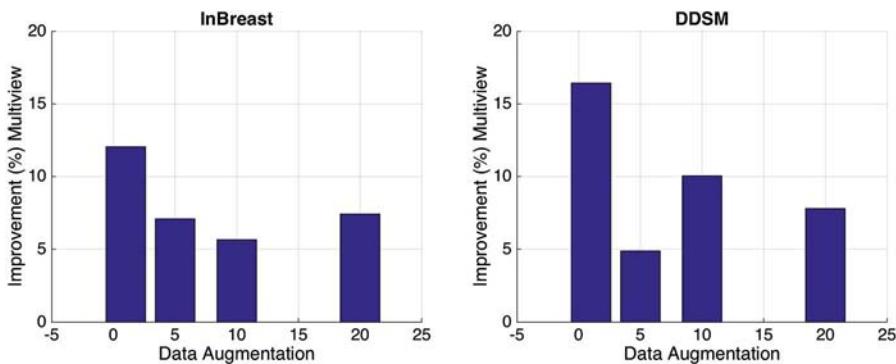
**FIGURE 14.7**

VUS in terms of data augmentation on InBreast (top) DDSM (bottom) for the MLO and CC views, and with each isolate input (image, micro-calcification and mass segmentation maps), all inputs (All) and both views (Multiview) together. The first column shows the results with the ImageNet pre-trained model, and the second column shows the randomly initialized models.

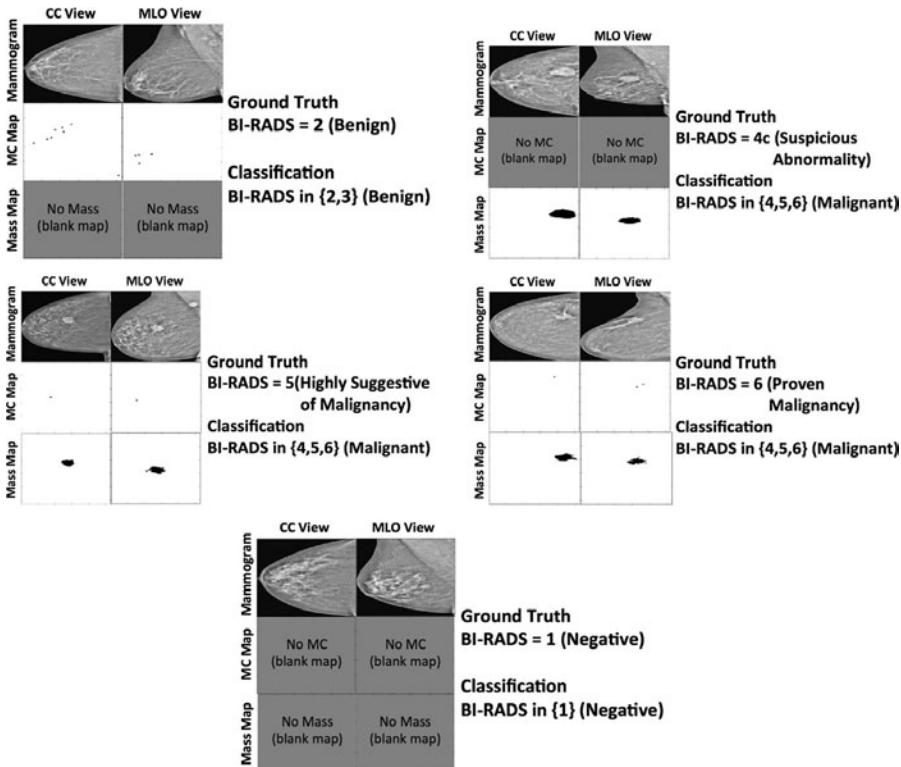
addition to all inputs and both views together. The improvement achieved with the use of the ImageNet pre-trained model, compared with the randomly initialized model, is shown in Fig. 14.8. We also show five examples from the classification process using the pre-trained model (without data augmentation) on InBreast test cases in Fig. 14.9.

Focusing on the cases where at least one lesion (mass or micro-calcification) exists allows us to compute the AUC for a two-class problem (benign or malignant). Using the model that is pre-trained with ImageNet and fine-tuned with InBreast without data augmentation, produces an AUC of  $0.91 (\pm 0.05)$ , and fine-tuned with DDSM results in an AUC of  $0.97 (\pm 0.03)$ .

Finally, running MatConvNet [57] on a standard desktop (2.3 GHz Intel Core i7 with 8 GB, and graphics card NVIDIA GeForce GT 650M 1024 MB), the training time for six models and the multiview model (without data augmentation) is one hour. With the addition of 10 artificial training samples per original training sample, the training time increases to four hours, and with 20 artificial training samples per original training sample, the training time increases to seven hours.

**FIGURE 14.8**

Mean improvement of the VUS results for InBreast (left) and DDSM (right) of the pre-trained models compared to the randomly initialized models.

**FIGURE 14.9**

InBreast test case results using ImageNet pre-trained model with no data augmentation, where the ground truth and the automatic classifications are shown.

## 14.6 DISCUSSION

The graphs in Fig. 14.7 show that the multiview results that use all inputs (images and segmentation maps), represented by the solid black curve, present the best performance among all models considered in this work. This shows evidence that the high-level features provided by each model are indeed useful for the classification of the whole exam, even though the input images and segmentation maps are not registered. Another interesting result shown in Figs. 14.7 and 14.8 is the improvement of 5% to 16% observed with the use of ImageNet pre-trained models, particularly when the training process does not involve data augmentation. One final point shown by Fig. 14.7 is that the results, with respect to data augmentation, saturates rather quickly with the use of five or more artificially generated training samples (per each original training sample). This point needs further investigation. For instance, it may be the case that geometric transformations may not be the most appropriate way of augmenting medical data. The visual results in Fig. 14.9 show that the system is likely to classify cases as malignant when micro-calcifications and masses are found, and as negative when no lesions are found. However, when either masses or micro-calcifications (but not both) are found, then the system can classify the case either as benign or malignant.

The results in Section 14.5 also show poor performance of the single/multi-view classifications containing only one of the inputs (image or segmentation maps). This may happen due to several reasons, such as that cases where BI-RADS > 1 may contain annotations for either micro-calcification or mass, but not for both lesions. Also, the mammogram images alone may not have sufficient information for a robust classification, particularly considering the fact that they are down-sampled around ten-fold to an input of size  $264 \times 264$ . It is also interesting to note the consistency of the results in the sense that micro-calcification segmentation maps produce better classification results than mass, which in turn is better than the image classification.

The comparison of our proposed methodology to previously proposed methods in the field (see Section 14.2) is difficult because most of these previous methods use datasets that are not publicly available and they also focus on the classification of individual lesions, as opposed to the classification of the whole exam that we propose. In any case, it is possible to compare the AUC results produced by our method to the AUC results of individual mass/micro-calcification classification of the current state of the art, which are between [0.9, 0.95] for MCs and mass classification [22,23]. Therefore, we can conclude that our proposed method is comparable (on InBreast) or superior (on DDSM) than the current state-of-the-art.

---

## 14.7 CONCLUSION

In this chapter, we show that the high level features produced by deep learning models are effective for classification tasks that use un-registered inputs. This is particularly important in mammograms, where registration is challenging due to non-rigid de-

formations. Moreover, the use of pre-trained models appears to be advantageous, compared to the randomly initialized models. This is somewhat an expected result given that the randomly initialized model is more likely to overfit the training data. We would like to emphasize that the results shown in Section 14.5 can serve as a baseline for the field because the data used is publicly available, which allows for a fair comparison with future works that will be published in the field [5]. Our proposal has the potential to open two research fronts that can be applied to other medical image analysis problems: (i) the use of deep learning models pre-trained with non-medical imaging datasets, and (ii) the holistic analysis of un-registered multi-view medical images.

---

## ACKNOWLEDGEMENTS

This work was partially supported by the Australian Research Council's Discovery Projects funding scheme (project DP140102794). Prof. Bradley is the recipient of an Australian Research Council Future Fellowship (FT110100623).

---

## REFERENCES

1. Gustavo Carneiro, Jacinto Nascimento, Andrew P. Bradley, Unregistered multiview mammogram analysis with pre-trained deep learning models, in: Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015, Springer, 2015, pp. 652–660.
2. Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhi-heng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, Li Fei-Fei, ImageNet Large Scale Visual Recognition Challenge, arXiv:1409.0575, 2014.
3. Ahmedin Jemal, Rebecca Siegel, Elizabeth Ward, Yongping Hao, Jiaquan Xu, Taylor Murray, Michael J. Thun, in: Cancer statistics, 2008, CA Cancer J. Clin. 58 (2) (2008) 71–96.
4. Béatrice Lauby-Sercretan, Chiara Scoccianti, Dana Loomis, Lamia Benbrahim-Tallaa, Véronique Bouvard, Franca Bianchini, Kurt Straif, Breast-cancer screening – viewpoint of the IARC Working Group, N. Engl. J. Med. 372 (24) (2015) 2353–2358.
5. Maryellen L. Giger, Nico Karssemeijer, Julia A. Schnabel, Breast image analysis for risk assessment, detection, diagnosis, and treatment of cancer, Annu. Rev. Biomed. Eng. 15 (2013) 327–357.
6. Arnau Oliver, Jordi Freixenet, Joan Martí, Elsa Perez, Josep Pont, Erika R.E. Denton, Reyer Zwiggelaar, A review of automatic mass detection and segmentation in mammographic images, Med. Image Anal. 14 (2) (2010) 87–110.
7. Jinshan Tang, Rangaraj M. Rangayyan, Jun Xu, Issam El Naqa, Yongyi Yang, Computer-aided detection and diagnosis of breast cancer with mammography: recent advances, IEEE Trans. Inf. Technol. Biomed. 13 (2) (2009) 236–251.
8. Ehsan Kozegar, Mohsen Soryani, Behrouz Minaei, Inês Domingues, et al., Assessment of a novel mass detection algorithm in mammograms, J. Cancer Res. Ther. 9 (4) (2013) 592.
9. Michael Beller, Rainer Stotzka, Tim Oliver Müller, Hartmut Gemmeke, An example-based system to support the segmentation of stellate lesions, in: Bildverarbeitung für die Medizin 2005, Springer, 2005, pp. 475–479.

10. Guido M. te Brake, Nico Karssemeijer, Jan H.C.L. Hendriks, An automatic method to discriminate malignant masses from normal tissue in digital mammograms, *Phys. Med. Biol.* 45 (10) (2000) 2843.
11. Renato Campanini, Danilo Dongiovanni, Emiro Iampieri, Nico Lanconelli, Matteo Masiotti, Giuseppe Palermo, Alessandro Riccardi, Matteo Roffilli, A novel featureless approach to mass detection in digital mammograms based on support vector machines, *Phys. Med. Biol.* 49 (6) (2004) 961.
12. Nevine H. Eltonsy, Georgia D. Tourassi, Adel Said Elmaghhraby, A concentric morphology model for the detection of masses in mammography, *IEEE Trans. Med. Imaging* 26 (6) (2007) 880–889.
13. Mehul P. Sampat, Alan C. Bovik, Gary J. Whitman, Mia K. Markey, A model-based framework for the detection of spiculated masses on mammography, *Med. Phys.* 35 (5) (2008) 2110–2123.
14. Roberto Bellotti, Francesco De Carlo, Sonia Tangaro, Gianfranco Gargano, Giuseppe Maggiapinto, Marcello Castellano, Raffaella Massafra, Donato Cascio, Francesco Fauci, Rosario Magro, et al., A completely automated CAD system for mass detection in a large mammographic database, *Med. Phys.* 33 (8) (2006) 3066–3075.
15. Jun Wei, Berkman Sahiner, Lubomir M. Hadjiiski, Heang-Ping Chan, Nicholas Petrick, Mark A. Helvie, Marilyn A. Roubidoux, Jun Ge, Chuan Zhou, Computer-aided detection of breast masses on full field digital mammograms, *Med. Phys.* 32 (9) (2005) 2827–2838.
16. John E. Ball, Lori Mann Bruce, Digital mammographic computer aided diagnosis (CAD) using adaptive level set segmentation, in: Engineering in Medicine and Biology Society, EMBS 2007, 29th Annual International Conference of the IEEE, IEEE, 2007, pp. 4973–4978.
17. Peyman Rahmati, Andy Adler, Ghassan Hamarneh, Mammography segmentation with maximum likelihood active contours, *Med. Image Anal.* 16 (6) (2012) 1167–1186.
18. Jaime S. Cardoso, Inês Domingues, Hélder P. Oliveira, Closed shortest path in the original coordinates with an application to breast cancer, *Int. J. Pattern Recognit. Artif. Intell.* 29 (01) (2015) 1555002.
19. C. Varela, S. Timp, N. Karssemeijer, Use of border information in the classification of mammographic masses, *Phys. Med. Biol.* 51 (2) (2006) 425.
20. Jiazheng Shi, Berkman Sahiner, Heang-Ping Chan, Jun Ge, Lubomir Hadjiiski, Mark A. Helvie, Alexis Nees, Yi-Ta Wu, Jun Wei, Chuan Zhou, et al., Characterization of mammographic masses based on level set segmentation with new image features and patient information, *Med. Phys.* 35 (1) (2008) 280–290.
21. I. Domingues, E. Sales, J.S. Cardoso, W.C.A. Pereira, InBreast-Database masses characterization, in: XXIII CBEB, 2012.
22. H.D. Cheng, X.J. Shi, Rui Min, L.M. Hu, X.P. Cai, H.N. Du, Approaches for automated detection and classification of masses in mammograms, *Pattern Recognit.* 39 (4) (2006) 646–668.
23. Liyang Wei, Yongyi Yang, Robert M. Nishikawa, Yulei Jiang, A study on several machine-learning methods for classification of malignant and benign clustered microcalcifications, *IEEE Trans. Med. Imaging* 24 (3) (2005) 371–380.
24. Karla Horsch, Maryellen L. Giger, Carl J. Vyborny, Li Lan, Ellen B. Mendelson, R. Edward Hendrick, Classification of breast lesions with multimodality computer-aided diagnosis: observer study results on an independent clinical data set, *Radiology* 240 (2) (2006) 357–368.

25. Yann LeCun, Yoshua Bengio, Convolutional networks for images, speech, and time series, in: *The Handbook of Brain Theory and Neural Networks*, vol. 3361, 1995.
26. Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton, ImageNet classification with deep convolutional neural networks, in: *NIPS*, vol. 1, 2012, p. 4.
27. Jason Yosinski, Jeff Clune, Yoshua Bengio, Hod Lipson, How transferable are features in deep neural networks?, in: *Advances in Neural Information Processing Systems*, 2014, pp. 3320–3328.
28. Yoshua Bengio, Learning deep architectures for AI, *Found. Trends Mach. Learn.* 2 (1) (2009) 1–127.
29. Inês C. Moreira, Igor Amaral, Inês Domingues, António Cardoso, Maria João Cardoso, Jaime S. Cardoso, INbreast: toward a full-field digital mammographic database, *Acad. Radiol.* 19 (2) (2012) 236–248.
30. Michael Heath, Kevin Bowyer, Daniel Kopans, Richard Moore, P. Kegelmeyer, The digital database for screening mammography, in: *Proceedings of the 5th International Workshop on Digital Mammography*, 2000, pp. 212–218.
31. Clément Farabet, Camille Couprie, Laurent Najman, Yann LeCun, Learning hierarchical features for scene labeling, *IEEE Trans. Pattern Anal. Mach. Intell.* 35 (8) (2013) 1915–1929.
32. Ross Girshick, Jeff Donahue, Trevor Darrell, Jitendra Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, in: *2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2014, pp. 580–587.
33. Yuting Zhang, Kihyuk Sohn, Ruben Villegas, Gang Pan, Honglak Lee, Improving object detection with deep convolutional networks via Bayesian optimization and structured prediction, in: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 249–258.
34. Dan C. Cireşan, Alessandro Giusti, Luca M. Gambardella, Jürgen Schmidhuber, Mitis-sis detection in breast cancer histology images with deep neural networks, in: *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2013*, Springer, 2013, pp. 411–418.
35. Holger R. Roth, Le Lu, Ari Seff, Kevin M. Cherry, Joanne Hoffman, Shijun Wang, Jiamin Liu, Evrim Turkbey, Ronald M. Summers, A new 2.5 D representation for lymph node detection using random sets of deep convolutional neural network observations, in: *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2014*, Springer, 2014, pp. 520–527.
36. Christopher M. Bishop, *Pattern Recognition and Machine Learning*, 2006.
37. Jifeng Dai, Kaiming He, Jian Sun, Instance-aware semantic segmentation via multi-task network cascades, *arXiv:1512.04412*, 2015.
38. Rasool Fakoor, Faisal Ladakh, Azadeh Nazi, Manfred Huber, Using deep learning to enhance cancer diagnosis and classification, in: *Proceedings of the ICML Workshop on the Role of Machine Learning in Transforming Healthcare (WHEALTH)*, Atlanta, GA, 2013.
39. Gustavo Carneiro, Jacinto C. Nascimento, Combining multiple dynamic models and deep learning architectures for tracking the left ventricle endocardium in ultrasound data, *IEEE Trans. Pattern Anal. Mach. Intell.* 35 (11) (2013) 2592–2607.
40. Angel Alfonso Cruz-Roa, John Edison Arevalo Ovalle, Anant Madabhushi, Fabio Augusto González Osorio, A deep learning architecture for image representation, visual interpretability and automated basal-cell carcinoma cancer detection, in: *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2013*, Springer, 2013, pp. 403–410.

41. Rongjian Li, Wenlu Zhang, Heung-Il Suk, Li Wang, Jiang Li, Dinggang Shen, Shuiwang Ji, Deep learning based imaging data completion for improved brain disease diagnosis, in: Medical Image Computing and Computer-Assisted Intervention – MICCAI 2014, Springer, 2014, pp. 305–312.
42. Tom Brosch, Youngjin Yoo, David K.B. Li, Anthony Traboulsee, Roger Tam, Modeling the variability in brain morphology and lesion distribution in multiple sclerosis by deep learning, in: Medical Image Computing and Computer-Assisted Intervention – MICCAI 2014, Springer, 2014, pp. 462–469.
43. Yanrong Guo, Guorong Wu, Leah A. Commander, Stephanie Szary, Valerie Jewells, Weili Lin, Dinggang Shen, Segmenting hippocampus from infant brains by sparse patch matching with deep-learned features, in: Medical Image Computing and Computer-Assisted Intervention – MICCAI 2014, Springer, 2014, pp. 308–315.
44. N. Dhungel, G. Carneiro, A.P. Bradley, Automated mass detection in mammograms using cascaded deep learning and random forests, in: 2015 International Conference on Digital Image Computing: Techniques and Applications (DICTA), 2015, pp. 1–8.
45. Anastasia Dubrovina, Pavel Kisilev, Boris Ginsburg, Sharbell Hashoul, Ron Kimmel, Computational mammography using deep neural networks, in: Workshop on Deep Learning in Medical Image Analysis (DLMIA), 2016.
46. Mehmet Gunhan Ertosun, Daniel L. Rubin, Probabilistic visual search for masses within mammography images using deep learning, in: 2015 IEEE International Conference on Bioinformatics and Biomedicine (BIBM), IEEE, 2015, pp. 1310–1315.
47. Neeraj Dhungel, Gustavo Carneiro, Andrew P. Bradley, Deep learning and structured prediction for the segmentation of mass in mammograms, in: Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015, Springer, 2015, pp. 605–612.
48. N. Dhungel, G. Carneiro, A.P. Bradley, Tree RE-weighted belief propagation using deep learning potentials for mass segmentation from mammograms, in: 2015 IEEE 12th International Symposium on Biomedical Imaging (ISBI), 2015, pp. 760–763.
49. N. Dhungel, G. Carneiro, A.P. Bradley, Deep structured learning for mass segmentation from mammograms, in: 2015 IEEE International Conference on Image Processing (ICIP), 2015, pp. 2950–2954.
50. John Arevalo, Fabio A. González, Raúl Ramos-Pollán, Jose L. Oliveira, Miguel Angel Guevara Lopez, Representation learning for mammography mass lesion classification with convolutional neural networks, *Comput. Methods Programs Biomed.* (2016) 248–257.
51. Yuchen Qiu, Shiju Yan, Maxine Tan, Samuel Cheng, Hong Liu, Bin Zheng, Computer-aided classification of mammographic masses using the deep learning technology: a preliminary study, in: SPIE Medical Imaging, International Society for Optics and Photonics, 2016, 978520.
52. Zhicheng Jiao, Xinbo Gao, Ying Wang, Jie Li, A deep feature based framework for breast masses classification, *Neurocomputing* 197 (2016) 221–231.
53. Michiel Kallenberg, Kersten Petersen, Mads Nielsen, Andrew Ng, Pengfei Diao, Christian Igel, Celine Vachon, Katharina Holland, Nico Karssemeijer, Martin Lillholm, Unsupervised deep learning applied to breast density segmentation and mammographic risk scoring, *IEEE Trans. Med. Imaging* 35 (5) (2016) 1322–1331.
54. Kersten Petersen, Mads Nielsen, Pengfei Diao, Nico Karssemeijer, Martin Lillholm, Breast tissue segmentation and mammographic risk scoring using deep learning, in: *Breast Imaging*, Springer, 2014, pp. 88–94.
55. Yuchen Qiu, Yunzhi Wang, Shiju Yan, Maxine Tan, Samuel Cheng, Hong Liu, Bin Zheng, An initial investigation on developing a new method to predict short-term breast cancer

- risk based on deep learning technology, in: SPIE Medical Imaging, International Society for Optics and Photonics, 2016, 978521.
- 56. Nobuyuki Otsu, A threshold selection method from gray-level histograms, *Automatica* 11 (285–296) (1975) 23–27.
  - 57. Ken Chatfield, Karen Simonyan, Andrea Vedaldi, Andrew Zisserman, Return of the devil in the details: delving deep into convolutional nets, arXiv:1405.3531, 2014.
  - 58. Thomas C.W. Landgrebe, Robert P.W. Duin, Efficient multiclass ROC approximation by decomposition via confusion matrix perturbation analysis, *IEEE Trans. Pattern Anal. Mach. Intell.* 30 (5) (2008) 810–822.

---

**NOTE**

- 1. This work is an extension of the paper published by the same authors at the Medical Image Computing and Computer-Assisted Intervention (MICCAI 2015) [1].

This page intentionally left blank

## 15

# Randomized Deep Learning Methods for Clinical Trial Enrichment and Design in Alzheimer's Disease

Vamsi K. Ithapu\*, Vikas Singh\*, Sterling C. Johnson†,\*

University of Wisconsin–Madison, Madison, WI, United States\* William S. Middleton Memorial Hospital, Madison, WI, United States†

## CHAPTER OUTLINE

---

15.1	Introduction .....	342
15.2	Background .....	344
15.2.1	Clinical Trials and Sample Enrichment .....	344
15.2.2	Neural Networks .....	345
15.2.3	Backpropagation and Deep Learning .....	346
15.2.3.1	Denoising Autoencoders (DA) and Stacked DA (SDA) ..	348
15.2.3.2	Dropout Networks .....	349
15.3	Optimal Enrichment Criterion .....	350
15.3.1	Ensemble Learning and Randomization .....	351
15.4	Randomized Deep Networks .....	352
15.4.1	Small Sample Regime and Multiple Modalities .....	353
15.4.2	Roadmap .....	354
15.4.3	rDA and rDr Training .....	356
15.4.3.1	Hyperparameters .....	358
15.4.4	The Disease Markers – rDAm and rDrm .....	358
15.5	Experiments .....	360
15.5.1	Participant Data and Preprocessing .....	360
15.5.2	Evaluations Setup .....	360
15.5.3	Results .....	362
15.6	Discussion .....	368
	Acknowledgements .....	374
	References .....	374
	Notes .....	377

---

## 15.1 INTRODUCTION

Alzheimer’s disease (AD) affects over 20 million people worldwide [1], and in the last decade, efforts to identify AD biomarkers have intensified. There is now broad consensus that the disease pathology manifests in the brain images years before the onset of AD. Various groups have adapted sophisticated machine learning methods, to *learn* patterns of pathology by classifying healthy controls from AD subjects. The success of these methods (which obtain over 90% accuracy [2]) has led to attempts at more fine grained classification tasks, such as separating controls from Mild Cognitively Impaired (MCI) subjects and even identifying which MCI subjects will go on to develop AD [3,4]. Even in this difficult setting, multiple current methods have reported over 75% accuracy. While accurate classifiers are certainly desirable, one may ask if they address a real practical need – if no treatments for AD are currently available, is AD diagnosis meaningful? To this end, [5,6] showed the utility of computational methods beyond diagnosis/prognosis; they may, in fact, be leveraged for designing *efficient* clinical trials for AD. There are, however, several issues with such procedures, and this work is in this context of designing methods for AD trials which are *deployable in practice* and *cost-effective*.

Recent clinical trials designed to evaluate new treatments and interventions for AD at the mild to moderate dementia stage have largely been unsuccessful and there is growing consensus that trials should focus on the earlier stages of AD including MCI or even the presymptomatic stage [7,8], if such stages can be accurately identified in individual subjects [9–11]. However, MCI is a clinical syndrome with heterogeneous underlying etymology that may not be readily apparent from a clinical work-up, posing a major challenge in reliably identifying the most probable beneficiaries of a putative effective treatment [12]. For instance, MCIs may have clinical but not biomarker evidence of incipient AD, may have biomarker evidence in some modalities, or, despite biomarker presence, may not show symptomatic progression during the trial time-period. An efficient MCI trial would ideally include only those patients most likely to benefit from treatment; who possess AD pathology based on a constellation of amyloid, tau and neural injury biomarker assessments, and who are most likely to progress clinically to symptomatic AD. The typical annual conversion rate to dementia among MCI due to AD is 3–20% across several studies [13]. Hence, over a two year trial, at best only 40% of participants would have naturally progressed, and the ability to detect the true efficacy of the trial is perhaps diminished.

Several ongoing AD trials “enrich” their population using one or more disease markers as inclusion criteria [8,14]. The general framework here is to effectively screen out subjects who are weak decliners (i.e., MCI who may not convert to AD) [15]. Unless there is a natural phase change (i.e., an elbow) in the distribution for distinguishing the at-risk and not-at-risk subjects, a fixed fraction of the total cohort are filtered out based on the study design. Imaging-based markers (e.g., fluorodeoxyglucose (FDG), hippocampal and ventricular volume) and cerebrospinal fluid (CSF) profiles have been shown to be effective in screening out low-risk subjects, due to the fact that disease manifests much earlier in imaging data compared to cognition [7,8].

However, these markers are uni-modal while several studies have shown the efficacy of multi-modal data [4,2]. Furthermore, CSF cannot be used in practice as a screening instrument because assays typically need to be performed in a single batch and are highly lab specific [16]. Several recent studies have used *computational* multi-modal markers derived from support vector machines (SVMs) and other machine learning models [14,5,6,17,18]. The strategy here uses imaging data from two time points (i.e., TBM or hippocampus volume change) and derives a machine learning based biomarker. Based on this marker, say, the top (strongest decliners) one-third quantile subjects may be selected to be included in the trial. Using this enriched cohort, the drug effect can then be detected with higher statistical power, making the trial more cost effective and far easier to setup/conduct. Most such approaches use longitudinal data, however, a practical enrichment criterion should only use baseline (trial start-point) data. We argue that existing approaches to enrichment, including state-of-the-art computational techniques, *cannot* guarantee this optimal enrichment behavior – optimally correlate with dementia spectrum with high confidence having access only to the baseline data, while simultaneously ensuring small intra-stage variance.

We approach the optimal enrichment design from basic principles. Specifically, consider a trial where participants are randomly assigned to treatment (intervened) and placebo (non-intervened) groups, and the goal is to quantify any drug effect. Traditionally, this effect is quantified based on a “primary” outcome, like cognitive measure or brain atrophy. If the distributions of this outcome for the two groups are statistically different, we conclude that the drug is effective. When the effects are subtle, the number of subjects required to see statistically meaningful differences can be huge, making the trial infeasible. Instead, one may derive a “customized outcome” from a statistical machine learning model that assigns predictions based on probabilities of class membership (no enrichment is used). If these customized predictions are statistically separated (classification is a special case), it directly implies that potential improvements in power and the efficiency of the trial are possible. This paper is focused on designing specialized learning architectures toward this final objective. In principle, *any* machine learning method should be appropriate for the above task. But it turns out that high statistical power in these experiments is not merely a function of the classification accuracy of the model, rather the conditional entropy of the outputs (prediction variables) from the classifier at test time. An increase in classifier accuracy does not directly reduce the variance in the predictor (from the learned estimator). Therefore, SVM type methods *are* applicable, but significant improvements are possible by deriving a learning model with the *concurrent* goals of classifying the stages of dementia *as well as* ensuring small conditional entropy of the outcomes.

We achieve the above goals by proposing a novel machine learning model based on ideas from deep learning [19]. Deep architectures are nonparametric learning models [20–22] that have received much interest in machine learning, computer vision and natural language processing recently [23–25]. They are effective and robust in learning complex concepts, and recently several authors extensively studied their success from both empirical and theoretical view-points [26,27]. Although powerful, it

is well known that they require very large amounts of data (unsupervised or labeled) [20], which is infeasible in medical applications including neuroimaging, bioinformatics, etc., where data dimensionality ( $d$ ) is always much larger than the number of instances ( $n$ ). A naïve use of off-the-shelf deep networks expectedly yields poor performance. Nevertheless, independent of our work, deep learning was used in structural and functional neuroimaging, where [28] use a region of interest approach while [29,30] sub-samples each data instance to increase  $n$ . Our work provides a mechanism where no such adjustments are necessary, and, in fact, the framework developed here is more generally applicable for learning deep networks in small sample regime (i.e., learning problems where number of data instances is much smaller than the data dimensionality like whole-brain voxel-wise analysis).

**Our contributions.** (a) We propose a novel, scalable, and a general, deep learning framework that is applicable for learning problems in the small sample regime, and provide certain guarantees for their performance; (b) Using our proposed models, we design novel predictive multi-modal imaging-based disease markers, based only on the trial start-time (baseline) acquisitions, that correlates very strongly with future AD decline; and (c) We show via extensive analyses using imaging, cognitive and other clinical data that the new computational markers result in cost-efficient clinical trials with moderate sample sizes when used as trial inclusion criteria. Section 15.2 briefly introduces clinical trials and deep networks, and Section 15.3 presents the optimal enrichment design. Section 15.4 presents our proposed models, referred to as *randomized deep networks*. Sections 15.5 and 15.6 extensively evaluate these models and discuss their efficacy in enrichment.

---

## 15.2 BACKGROUND

The design and learning of randomized deep networks is directly motivated by the optimal enrichment design. Hence, we first present some background on the sample size estimation for conducting clinical trials [31] discussing the necessity of a *computational enrichment criterion*. Although there are many classes/types of deep architectures in the literature, we focus our presentation using two of the most widely used (and well studied) architectures – stacked denoising autoencoders [32] and fully-supervised dropout [33]. The ideas presented here are applicable for any such architectures used for learning problems in the small-sample regime.

### 15.2.1 CLINICAL TRIALS AND SAMPLE ENRICHMENT

Consider a randomized clinical trial (RCT) designed to test the efficacy of some treatment for an underlying disease condition [31,34]. The population under study are randomly assigned to either of the treatment (or intervention) and non-treatment (or placebo) groups. If the drug indeed offers an improvement, then the two groups should show this change when measured using some outcome measure summarizing

the disease status. Such change would generally correspond to *reducing* the disease progression to a certain extent, referred to as the effect size [34], in the treatment group compared to the placebos. The outcome measure is, in general, a reliable disease marker. Given such an outcome, the trial efficacy is measured by estimating the Type-II error between the two groups, after inducing the drug or intervention. Clearly, this Type-II error (and the effect size) would be influenced by the choice of the outcome and the size (and demographics) of the trial population. Hence, in practice, one would want to “estimate” the trial’s efficacy ahead of time to ensure that the effect size is good enough, the population is reasonably large and diverse, and the outcome is appropriately chosen – basically ensuring that the trial makes sense. In such a hypothetical RCT, the drug is induced by *fixing* the effect size ahead of time, and computing the resulting Type-II error for the given outcome and population.

Let  $\delta$  denote the difference of mean outcome (the standard change) between the trial start and end points (e.g., 2 years) in the placebos. Let  $\sigma$  be the standard deviation of the outcome, and the effect size be  $\eta$  (e.g., 0.25, a 25% reduction in the disease is desired).  $\delta$  and  $\sigma$  are known *a priori* (reported in alternate studies on the disease). The treatment group is then *expected* to have the change in outcome decreased to  $(1 - \eta)\delta$ , which will correspond to a hypothetical improvement of  $\eta$  induced by the drug/treatment. Within this setting, the number of samples  $s$  required per arm (treatment and placebo) is given by [31]

$$s = \frac{2(Z_\alpha - Z_{1-\beta})^2 \sigma^2}{(1 - \eta)^2 \delta^2} \quad (15.1)$$

where  $(1 - \beta)$  denotes the desired statistical power at a significance level of  $\alpha$ . This expression directly follows from applying a difference of means *t*-test, where the means are computed from the two distributions of interest – outcome change in treatment and placebo groups [31]. The null hypothesis is that mean change in the outcome is same for the treatment and placebo groups. The necessity of sample enrichment can be directly seen from (15.1). If the population under study has less standard change in the outcome  $\delta$ , then the required sample  $s$  for achieving a given power  $1 - \beta$  will be very large. Alternatively, the power can be maximized only when the incipient change  $\delta$  is as large as possible in the trial population. Hence, the participants need to be *enriched* so as to include only those subjects with large change in the disease *during* the trial time-line. We define an *optimal enrichment criterion*, in Section 15.3, based on these ideas and (15.1).

### 15.2.2 NEURAL NETWORKS

Artificial neural networks (ANN) are representation learning machines introduced in the late 1950s for learning complex concepts [35]. An ANN transforms a given (input) instance/example (a  $d$ -dimensional vector of features/covariates) into a new representation that may be used within the context of classification or regression or other learning paradigms. These transformations are nonlinear, and possibly non-convex,

and calculated by first computing an affine projection of the input, followed by applying a monotonic nonlinear ‘‘activation’’ function over these projections (which may not be necessarily point-wise) [20]. The resulting features correspond to some high-level and abstract representations of the inputs. Given a set of examples  $(\mathbf{x}_i, \mathbf{y}_i)_{i=1}^n$  from  $n$  different instances (or subjects in the case of medical imaging), the new representations are given by  $\mathbf{h}_i = \sigma(\mathbf{W}\mathbf{x}_i + b)$ .  $\mathbf{W} \in \mathbb{R}^{d_1 \times d}$  and  $b \in \mathbb{R}^{d_1 \times 1}$  are the unknown transformation coefficients (i.e., weights) (assuming  $\mathbf{y}_i \in \mathbb{R}^{d_1}$ ).  $\sigma(\cdot)$  is the point-wise activation, and in general, it is a sigmoid (i.e.,  $\sigma(z) = \frac{1}{1+\exp(-z)}$ ), although other types of functions including hyperbolics or rectified linear units [36] may be used.

This *single-layer* neural network comprises of a visible layer ( $\mathbf{x}_i$ s) and an output layer ( $\mathbf{y}_i$ s). *Multi-layer* neural networks (MLNN) perform  $L > 1$  such transformations sequentially, thereby resulting in  $L$  *hidden* layers ( $\mathbf{h}_i^l$ s,  $l = 1, \dots, L$ ):

$$\mathbf{h}_i^l = \sigma(\mathbf{W}^l \mathbf{h}_i^{l-1} + b^l), \quad \mathbf{h}_i^0 = \mathbf{x}_i, \quad l = 1, \dots, L, \quad i = 1, \dots, n. \quad (15.2)$$

The highest level hidden layer ( $\mathbf{h}_i^L$ s) may then be used for a given learning task. Specifically, a classification or regression model can be trained using  $(\mathbf{h}_i^L, \mathbf{y}_i)_{i=1}^n$ , or in certain situations,  $\mathbf{h}_i^L$ s may directly correspond to  $\mathbf{y}_i$ s. Once the learning is done, the output layer of MLNN would simply be the predictions  $\hat{\mathbf{y}}_i$ s. The hidden layer lengths are denoted by  $d_l$ ,  $l = 1, \dots, L$ , and whenever  $d_l > d \forall l$ , the network learns over-representations of the inputs that are invariant and abstract in some sense. Fig. 15.1A shows the architecture of a typical  $L$ -layered MLNN. Layer-to-layer connections shown in the figure correspond to applying the corresponding transformations  $((\mathbf{W}^l, b^l))$ , followed by point-wise sigmoid nonlinearity  $\sigma(\cdot)$ .

### 15.2.3 BACKPROPAGATION AND DEEP LEARNING

To learn the unknown transformations  $(\mathbf{W}^l, b^l)$  one compares the MLNN predictions  $\mathbf{y}_i$ s to the desired outputs using some appropriate loss function  $\ell(\cdot)$  (e.g., squared loss, entropy, or divergences), chosen entirely based on the problem at hand. This objective is non-convex because of the presence of multiple (nested) composition of nonlinearities, and hence, the estimation proceeds via stochastic gradients on the loss function objective [37]. This procedure is referred to as backpropagation [38], because the errors in the final/output layer need to be ‘propagated’ back to the inputs for estimating the coefficients. Several variants of backpropagation have been proposed over the last few decades. The reader should note that it is impractical to review (or even refer to) the vast literature on exhaustive empirical analysis of the various backpropagation strategies – and there have been very many. We only point out the main bottlenecks of backpropagation, which eventually made the neural network learning obsolete (while kernel machines and decision trees received more attention because of their ease of learning and implementability). An interested reader can refer to [19,39] and others for further details.

Although MLNNs have attractive theoretical properties (e.g., a 3-layer network can represent any polynomial of arbitrarily high degree, see Chapter 5 in [39]), training them involves a difficult optimization task due to the composition of nonlinear (and possibly non-convex) functions like sigmoids. The parameter space is highly non-convex with many local minima and saddle points [37] (see Chapter 6 in [19]). Stochastic minimization methods, nevertheless, compute a local minima, but these may be sensitive to initialization [20,38]. The *goodness* of such solutions – in terms of stability to noise, saddle point behavior and sensitivity to perturbations – depend on the number of stochastic iterations, and in turn, on the number of training samples available to exhaustively (and empirically) search through the solution space. Further, gradient based methods are prone to exponential decay of errors for large MLNNs, since the errors computed at the last layer “die-out”, in some sense, by the time they reaches the inputs because of the presence of nonlinearities. Hence, even large errors from the objective may not lead to reasonably good gradient paths for the bottom layer transformations. Lastly, MLNN learning involves critical modeling/design choices about the network structure (number and lengths of layers). Although several studies have shown the necessity of over-represented networks, there was no consensus on which variant of classical backpropagation would best suite a given choice of network, and if there was any standardized way of choosing the network structure. These issues make efficient and/or robust learning of MLNNs difficult, which will eventually lead to poor generalization. Deep learning refers to a suite of algorithms for efficient training of MLNNs by mitigating some of these issues [20, 19]. The *depth* simply refers to the many levels of transformations.

Recall the main issue with MLNNs was that the multiple function compositions makes the search space non-convex with many local minima and saddle points. Deep learning algorithms partly mitigate this issue by disentangling the compositions and only working with one-layer (i.e., one transformation) at a time. Working with each layer individually makes the objective simpler albeit, non-convex but likely with fewer local minima. Once *good* estimates of the transformations are obtained *layer-wise*, the entire network can then be initialized with these estimates. The resulting final layer predictions can then be compared with desired outputs to adjust or *fine-tune* the estimated parameters across all layers. This fine tuning is the same as performing complete backpropagation but using layer-wise estimates as an initialization or warm-start [20]. The basic rationale is that once reasonable layer-wise warm-starts are obtained, the transformation coefficients are already in some good solution bowl in the gradient search space, which may be smoother and better behaved than the ones obtained with random initializations (and/or learning all layers concurrently) or whole-network warm starts [20]. The challenge then is to construct such efficient layer-wise procedures to perform this two stage learning – initialize with layer-wise *pretraining*, followed by global fine-tuning. Several such procedures have been constructed, all of which broadly fall under the two categories – restricted Boltzmann machines [40] and autoencoders [32]. More recently, an interesting learning procedure referred to as “Dropout”, has been proposed to address the over-fitting problem whenever large MLNNs are to be learned [33]. Specifically, [41,24] showed

that one can perform fully supervised dropout with dropout rate of 0.5 (we will discuss more about this shortly) and completely *by-pass* the layer-wise pretraining. As discussed earlier, we present our models using one of the autoencoder schemes, denoising autoencoders, and dropout learning as building blocks.

### 15.2.3.1 Denoising Autoencoders (DA) and Stacked DA (SDA)

An autoencoder is a single-layer network that learns robust *distributed representations* of the input data. Given inputs  $\mathbf{x}_i$ , the autoencoder learns hidden/latent representations  $\mathbf{h}_i = \sigma(\mathbf{W}\mathbf{x}_i + b)$ , such that the reconstructions  $\hat{\mathbf{x}}_i = \sigma(\mathbf{W}^T\mathbf{h}_i + c)$  are as *close* as possible to  $\mathbf{x}_i$ . It minimizes the following input reconstruction error

$$\mathcal{Z}_a(\{\mathbf{x}_i\}_1^n, \theta) := \arg \min_{\mathbf{W}, b, c} \sum_{i=1}^N \ell(\mathbf{x}_i, \sigma(\mathbf{W}^T \sigma(\mathbf{W}\mathbf{x}_i + b) + c)) \quad (15.3)$$

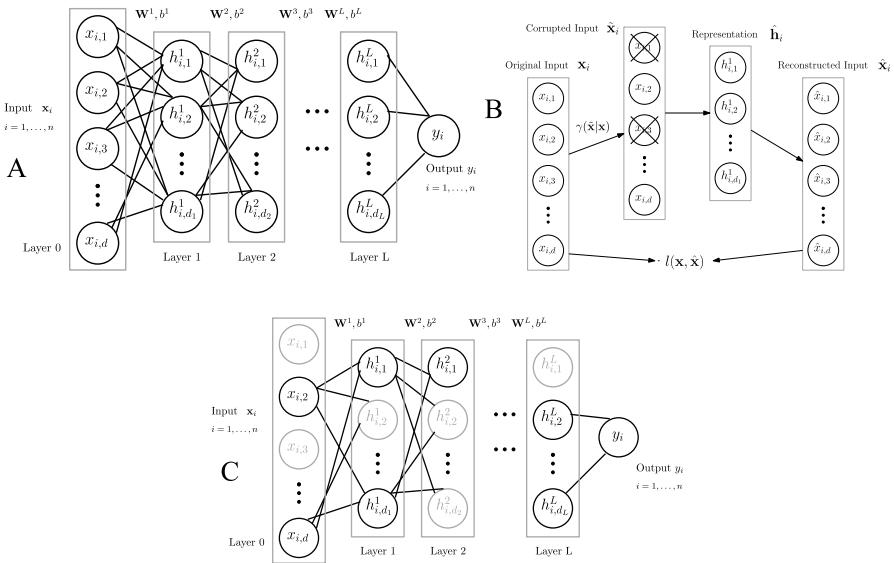
where  $\ell(\cdot, \cdot)$  denotes a suitable loss function, e.g., squared loss. A stochastic gradient scheme [42] can be used to perform this minimization. Observe that with no other constraints on  $(\mathbf{W}, b, c)$ , the above minimization could potentially learn *identity* mappings, i.e.,  $\mathbf{h}_i$ s will be identical to the inputs, making the autoencoding setup useless. Several approaches have been suggested to avoid such identity mappings and instead learn useful representations [20]. We consider the approach where the inputs  $\mathbf{x}_i$  are *corrupted* stochastically and the autoencoder is forced to reconstruct the original (non-corrupted) versions. This is referred to as a *denoising* autoencoder (DA) [32], and the minimization in (15.3) will change to

$$\mathcal{Z}_{da}(\{\mathbf{x}_i\}_1^n, \theta) := \arg \min_{\mathbf{W}, b, c} \sum_{i=1}^n \mathbb{E}_{\tilde{\mathbf{x}} \sim \gamma(\tilde{\mathbf{x}}|\mathbf{x})} \ell(\mathbf{x}_i, \sigma(\mathbf{W}^T \sigma(\mathbf{W}\tilde{\mathbf{x}}_i + b) + c)) \quad (15.4)$$

where  $\gamma(\cdot)$  is a stochastic corruption function, and  $\tilde{\mathbf{x}}_i$  represents the corrupted  $\mathbf{x}_i$ .  $\gamma(x_{ij}) = x_{ij}$  with some (given) probability  $\zeta$  and 0 elsewhere ( $j = 1, \dots, d$  are the data dimensions).

DA is a stochastic autoencoder whose learning procedure seeks to undo the input corruptions (hence the name, denoising). The corruption forces the transformations to correspond to some properties of input data, since the reconstruction error in (15.4) decreases only if the transformations pick out the most informative data dimensions. Hence  $\mathbf{h}_i$ s are abstract enough to *generate* the inputs [32]. Fig. 15.1B summarizes the DA learning. Multiple DAs can then be concatenated to construct a *stacked* DA (SDA), where the hidden representations of  $l$ th DA are the uncorrupted inputs to  $(l+1)$ th DA ( $l = 1, \dots, L-1$ ). The objective of SDA is

$$\begin{aligned} \mathcal{Z}_{sda}(\{\mathbf{x}_i\}_1^n, L, \theta) &:= \sum_{l=0}^{L-1} \mathcal{Z}_{da}(\{\mathbf{h}_i^l\}_1^n, \theta); \\ \mathbf{h}_i^l &= \sigma(\mathbf{W}^l \mathbf{h}_i^{l-1} + p^l); \quad \mathbf{h}_i^0 = \mathbf{x}_i. \end{aligned} \quad (15.5)$$

**FIGURE 15.1**

(A)  $L$ -layered MLNN transforming the input features  $\mathbf{x}_i$  to a desired output  $\mathbf{y}_i$ . The lengths of hidden layers are  $d_1, d_2, \dots, d_L$ . Layer 0 is the visible layer corresponding to the inputs  $\mathbf{x}_i$ . Layers 1 to  $L$  are the  $L$  hidden layers, and  $y_i$ s denote the outputs. Layer-to-layer transformations are represented by  $\mathbf{W}^l$  and  $b^l$ .  $i = 1, \dots, n$  and  $l = 1, \dots, L$ . (B) The learning process of DA where the input  $\mathbf{x}_i$  is corrupted to generate  $\tilde{\mathbf{x}}_i$ , which is then used to reconstruct an approximation of  $\mathbf{x}_i$  denoted by  $\hat{\mathbf{x}}_i$ . The crossed-out elements in  $\tilde{\mathbf{x}}_i$  represent the stochastically corrupted units i.e., for these units  $\tilde{\mathbf{x}}_{i,\cdot} = 0$ . For the rest of the non-crossed-out units  $\tilde{\mathbf{x}}_{i,\cdot} = \mathbf{x}_{i,\cdot}$ . The loss function  $\ell(\cdot, \cdot)$  then compares the original  $\mathbf{x}_i$  to the reconstruction  $\hat{\mathbf{x}}_i$ . (C) The strategy of dropout learning, where, a fraction  $\eta(\cdot)$  of all the units are *dropped* in each layer i.e., all the connections involving these units are dropped (biases are never dropped). Under the given iteration, this smaller network is learned instead of the entire one, and the process repeats stochastically across all iterations.

It is straightforward to see that the structure of SDA is identical to that of an  $L$ -layered MLNN, and the learned parameters ( $\mathbf{W}^l, b^l, c^l$ ) can be used to initialize the MLNN. The final layer  $\mathbf{h}_i^l$  can then be compared to the desired outputs and the errors can be propagated back to fine-tune the network. SDA learning proceeds layer-wise.

### 15.2.3.2 Dropout Networks

Unlike SDAs where the corruption is applied stochastically in a layer-wise fashion, the dropout network simply drops a fraction of the network units across *all* layers. Fig. 15.1C shows the dropout learning procedure. During the training stage, in each iteration of the backpropagation, the transformation parameters of a smaller

sub-network are updated. The size of this sub-network is determined by the dropout rate  $\zeta$  (which, in general, is the same across all layers). The sub-networks across different iterations are chosen randomly, and hence, each transformation parameter is updated approximately  $(1 - \zeta)t$  times (where  $t$  is the number of gradient iterations). During the test time, the learned parameters are scaled up by  $(1 - \zeta)$  corresponding in predicting the final layer representation of the input. One can perform pretraining type initialization in tandem with dropout since the latter does not work layer-wise [33]. In principle, both SDA and dropout are different types of feature corruption based regularization scheme, and the dropout learning procedure has been introduced to address the overfitting issue in MLNNs [33]. However, [41,24] have shown that, in certain cases, pretraining can be ‘by-passed’ completely when using dropout. This is because the dropout dynamics results in smaller networks, there by reducing the effect of the local minima while also resulting in parameter estimates that are generalizable and robust to input perturbations. With this background we now present the rationale behind our proposed deep learning models for the small sample regime followed by their structure and learning procedure.

---

### 15.3 OPTIMAL ENRICHMENT CRITERION

The ideas driving our randomized deep networks are motivated, as discussed earlier, from the design of optimal enrichment criterion. The sample size estimation from (15.1) suggests that, for a given significance  $\alpha$  and power  $1 - \beta$ , the required sample size  $s$  (per arm) *increases* as the standard deviation of the outcome  $\sigma$  *increases*, and/or, as the standard change in the outcome  $\delta$  *decreases*. Recall that the hypothetical RCT presented in Section 15.2.1 asks for the change in outcome to decrease by  $(1 - \eta)$  (or the disease should reduce by  $\eta$ ). Hence, a smaller  $\eta$  directly indicates that a smaller drug induced change in the treatment is desired and can be detected, there by increasing the robustness of the trial. Clearly, for a fixed number of subjects  $s$ , decreasing  $\sigma$  and/or increasing the mean change in the outcome  $\delta$ , will decrease the detectable drug effect  $\eta$ . Hence, (15.1) implies that one can design an efficient clinical trial by selecting the population and the outcome such that, during the span of the trial, there will be large longitudinal changes in the outcome  $\delta$  *and* small outcome variance  $\sigma^2$ . Ideally, the trial should *not* include subjects who remain healthy (and/or do not decline) as time progresses because they will reduce the trial’s sensitivity for detecting any drug effect. Nevertheless, in general, the trial is always diluted by including those subjects who are unlikely to benefit from the drug, indicating that the alternative hypothesis that the drug induces some effect is moot (for this subgroup). Removing such weak decliners from the trial will result in large  $\delta$ , but may not necessarily ensure smaller  $\sigma^2$  for the outcome. To address this, we need to *explicitly* reduce the outcome variance; this is generally not feasible because the outcomes are cognitive and/or blood flow based

measures whose statistical characteristics (range, median, structure, etc.) cannot be altered.

An alternative approach to ensure smaller outcome variance is by designing a *computational* disease marker that predicts the decline in the disease with high confidence. This new marker is explicitly ensured to have smaller variance. If the correlation of this computational marker with the intended trial outcome is strong, then the low variance characteristic of the new marker translates, to a certain extent, to the outcome due to the marker's strong predictability of future decline. Hence, once the new marker is established to have strong correlations to the outcome, one can use it as an “inclusion or enrichment criterion” to filter the trial population – those subjects whose future decline is small according to this enrichment criterion are removed from the trial. An optimal enrichment criterion would then be a computational disease marker with strongest predictability of the disease with smallest possible variance across subjects. Note that the intuition behind this is that reduction of this enricher's variance will indirectly reduce the outcome variance. Using an existing disease marker may not necessarily guarantee this behavior, and hence, one needs to explicitly design such a criterion. In summary, the optimal inclusion criterion should have the following properties:

- strong discrimination power for different stages of the disease – i.e., *no approximation/modeling bias*;
- strong correlation with an existing disease outcomes or other biomarkers – i.e., *strong predictive power* of the disease; and
- small prediction variance – i.e., *small variance* on the intended outcome.

The above requirements can be formulated as a statistical estimation problem. Given the inputs, which may include medical imaging data and/or other relevant types of clinical and/or demographic information, the estimator output is a *new* disease marker satisfying the above requirements. No approximation bias and strong predictive power implies that the estimator needs to be unbiased with respect to the classes/labels. Concurrently with the low prediction variance, the problem of designing the optimal enrichment criterion reduces to constructing a *minimum variance unbiased (MVUB) estimator* of the disease.

### 15.3.1 ENSEMBLE LEARNING AND RANDOMIZATION

The existence of an MVUB estimator is governed by whether the Cramer–Rao lower bound can be achieved, i.e., any unbiased estimator that achieves this lower bound is referred to as an MVUB [43]. However, finding such an unbiased estimator can be difficult, and especially, in the small-sample regime where  $d \gg n$ , computing the lower bound itself may be problematic. Instead, in such settings an alternative approach to designing MVUBs is by first generating sets of unbiased estimators that are approximately *uncorrelated* to each other (in the ideal case, independent), and combine them in some reasonable manner to reduce the variance while retaining unbiasedness. This is the classical bootstrap approach to MVUB design in high-dimensional

statistics [44], and the family of models that adapt this approach are broadly referred to as “ensemble learning” methods [45]. The variance reduction behavior follows from ensuring that the estimators/learners have sufficiently small cross-correlation, and so, their linear combination will have smaller variance compared to that of each individual estimator/learner. To see this, let  $\mathbf{z}_k$  for  $k = 1, \dots, K$  denote  $K$  different random variables (e.g., the outputs from  $K$  different estimators/learners), and  $\bar{\mathbf{z}}$  denote their mean. Assuming, without any loss of generality, that the variance and cross-covariance of  $\mathbf{z}_k$ s are  $\sigma^2$  and  $\rho$ , respectively, we have

$$\text{Var}(\mathbf{z}_k) = \sigma^2, \quad \text{Cov}(\mathbf{z}_k, \mathbf{z}_{k'}) = \rho; \quad \text{then} \quad \text{Var}(\bar{\mathbf{z}}) = \frac{\sigma^2}{K} + \frac{(K-1)\rho\sigma^2}{K}. \quad (15.6)$$

Depending on  $\rho$ , the variance of  $\bar{\mathbf{z}}$  goes from  $\frac{\sigma^2}{K}$  to  $\sigma^2$  (under the assumption that  $\rho > 0$ , i.e., the estimators are not negatively correlated). In the current setting, since all the  $K$  estimators/learners share the same set of input data, they cannot be independent. However, by ensuring that  $\rho$  is as small as possible and increasing  $K$ , one can make sure that the composed estimator  $\bar{\mathbf{z}}$  will have the smallest possible variance. Several approaches may be used to ensure small  $\rho$ , most of which are based on *randomly* dividing the input dimensions, data instances and/or other estimation/learning parameters into  $K$  subsets and constructing one estimator from each of these subsets [45]. The outputs of these estimators can then be considered to be random (or stochastic, in some sense) approximations of the ideal output with zero-bias and small variance. There is extensive empirical evidence for such strategies where the eventual *ensemble learner* will retain the discriminative power of the individual weak learners while reducing the apparent prediction variance [45]. We will use deep networks to construct an efficient weak learner, followed by presenting a systematic strategy to construct the ensemble – this overall learning procedure will correspond to the randomized deep network model.

---

## 15.4 RANDOMIZED DEEP NETWORKS

The ideas from ensemble learning do address the variance reduction requirement for the optimal enrichment criterion. However, the weak learners that go into this ensemble need to be unbiased to begin with. Recall the success of deep learning in learning complex concepts in computer vision, natural language processing and information retrieval [23–25]. It is reasonable to expect that these methods should be translatable to learning problems in medical imaging and neuroimaging with improved performance than the state-of-the-art. This should be plausible, especially because the concepts to be learned in brain imaging might be “less” complex from the perspective of the size of hypotheses spaces to be searched over. Hence, the desired ensemble MVUB could be constructed using deep network weak learners that are trained appropriately to predict the disease. There are a few caveats, however, as described below.

### 15.4.1 SMALL SAMPLE REGIME AND MULTIPLE MODALITIES

Although the pretraining idea in tandem with the dropout learning address the issue of non-convexity to a certain extent, [46,20,47] have shown extensive evidence that one of the main reasons for the success of deep learning is the availability of large number of unsupervised and/or supervised training instances. The few studies that apply deep learning methods in neuroimaging have reported such observations as well [29,30]. Simply put, the non-convexity, together with the stochasticity that comes from the corruption in DA or the dropout process, demand a very large number of gradient search iterations and data instances to effectively search through the solution space in computing generalizable solutions. As the data dimensionality increases, the dataset size required to ensure that sufficiently many combinations of corrupted/dropped dimensions are passed to the objective also increases proportionally. Now, the fundamental difference between vision type domains, and medical imaging and bioinformatics is the lack of such large datasets. In vision, one has access to extremely large datasets (on the orders of millions of images) – including both large number of unlabeled and supervised instances (e.g., in object recognition, document analysis, and so on). On the other hand, a typical voxel wise imaging study, for instance, will have  $n < 500$  subjects while the number of voxels/features ( $d$ ) will exceed a million – the classical small-sample regime.

Further, in contrast to classical machine learning tasks, the problems in bioinformatics involve data from multiple acquisition types/domains (e.g., brain imaging data including Magnetic Resonance images (MRI), Positron Emission Tomographic (PET) images, several types of cognitive and neuropsychological scores, lists of vascular and blood perfusion data, genetic single nucleotide polymorphisms). Most applications arising in these areas would require efficient statistical models for “fusing” such multi-modal data, especially because they provide distinctive information about the underlying disease. Such multi-modal studies have always been shown to result in higher performance than uni-modal ones [48]. Using classical version of deep architectures, including SDA or dropout networks, for these multi-modal problems in tandem with  $d \gg n$  issues will result in unreliable outputs, with no guarantee of generating either stable or generalizable solutions. This is a direct consequence of under sampling issues in statistical learning (like in VC-dimension or Nyquist sampling analyses), where the number of training instances cannot be below a certain pre-specified number for efficient estimation of the underlying concepts [39], and with presence of multi-modal data the concept is far more complex than from the uni-modal setting.

One of the main contributions of our proposed randomized deep networks is to translate the success of deep learning to multi-modal neuroimaging problems in the small sample regime. We will use neuroimaging terminology like voxels, subjects, etc., to present our models mainly because  $d \gg n$  regime is common in brain imaging. The simplest solution to mitigating the  $d \gg n$  issue is by pre-selecting the most influential voxels using some statistical test (e.g.,  $t$ -test based on some known grouping information) and using only these as features within the learning framework downstream. However, this proposal is lossy, in the sense that, non-selected features

are discarded irrespective of how discriminative they are; and so, to ensure least information loss, the processing should be “appropriately” chosen. This not only makes the selection process biased to the task (and not generalizable), but, more importantly, the performance would be entirely driven by the “goodness” of pre-processing. Bourgon et al. [49] discuss this necessity of avoiding bias and influence of data processing on the false positive error rates for the eventual learning task.

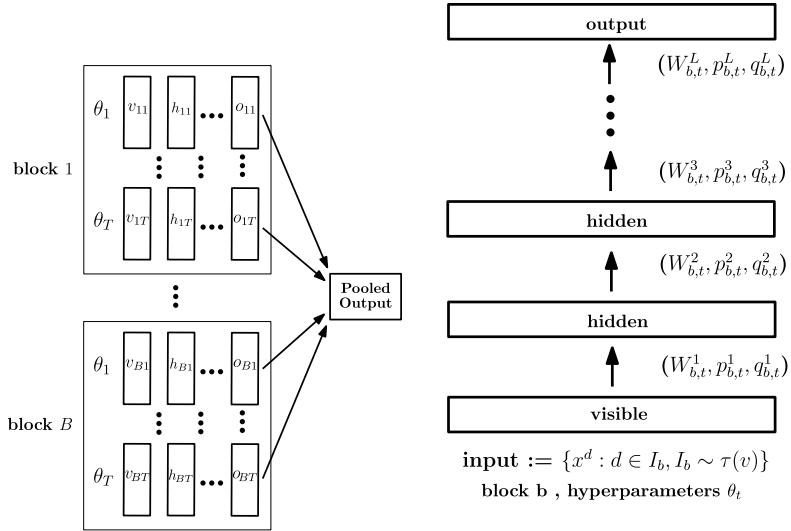
On the other hand, one can avoid the feature screening completely by working with slices of the input data (e.g., 2D slices or smaller resolution images from a 3D image). Although this is lossless, working with one slice at-a-time will restrict interactions of voxels and brain regions from anatomically far apart regions. Clearly, allowing for arbitrary far apart interactions in predicting the output label would be reasonable following the hypotheses that, in general, all brain regions have complex biological interactions in generating the final label (e.g., the disease status). An alternative to these extremes is to *categorize* (or *tessellate*) the entire set of voxels into multiple subsets (e.g., spatially contiguous blocks) and learn a network (e.g., SDA or dropout network) on each block separately while allowing for different blocks to interact with each other in some prescribed manner. The network learned in each block will then serve as a weak learner in the above described ensemble (as in boosting [45]). Later, the individual block-wise networks, which interact with each other, can be combined in a meaningful manner yielding a better fit for the dependent variable  $y$ .

### 15.4.2 ROADMAP

The starting point of our proposed models is the rationale that this above described proposal is viable in some sense. Observe that if the number of voxels in each subset/block is comparable to  $n$ , and much smaller than the input dimensionality allowed in this smaller subset (denoted by  $\tilde{d}$ ), the learning problem at the level of *this* block is well defined. For instance,  $d$  voxels can be divided into  $B$  subsets and an SDA or dropout network can be trained on each block separately. On its own, each block’s output will not be useful since it corresponds to only a small number of features. But our final output will utilize the outputs from all  $B$  blocks, treating each as a stub (i.e., weak learner). This scenario assumes that the tessellation (i.e., the voxel to block mapping) is given or fixed. But in practice, the “correct” tessellation is not known in advance. However, it is possible to marginalize over this as described below. Consider the set of all possible tessellations  $\mathcal{C}$  (an exponentially large set). Constructing a learning model, however simple, for each item in  $\mathcal{C}$  is unrealistic. Instead, we can use the standard trick of marginalizing over  $\mathcal{C}$  by drawing a large number of samples from it to approximate the summand. In other words, by resampling from  $\mathcal{C}$  and deriving many possible  $B$  (sufficiently large) number of subsets (and learning network weights for them), we can obtain partial invariance to the lack of correct tessellation. Note that one realization of this process corresponds to drawing a sample from  $\mathcal{C}$ ; this process provides *randomization* over clusterings where  $B$  different predictions from input instance are combined to generate a single classification/regression output.

The number of blocks  $B$  can be fixed ahead of time. The process of assigning voxels to blocks can leverage domain information like neighborhood interactions and/or consistency of correlations across all  $d$  dimensions or any other randomized procedure. For instance, if it is known that a local neighborhood has strong interactions, then each such neighborhood can constitute a single block. To allow for arbitrary brain regions to interact with each other possibly with in a block, the block construction should group arbitrary voxels together. Hence, to balance this arbitrary voxel selection and domain information driven block generation, we first ‘rank’ all the voxels according to some information criterion (e.g., Kullback–Leibler divergence, entropy). We then sample the voxels (for a given block) without replacement according to the cumulative distribution of these ranks. If these blocks are *sufficiently* independent, then any linear combination of their outputs will have smaller variance resulting in an MVUB as desired. However, the input data to all the blocks comes from the same subject, implying that the  $B$  weak learners will always be correlated. Nevertheless, we can force them to be approximately uncorrelated by adding another level of randomization over the block generation process. To do this, observe that, beyond the block generation, there are other sets of hyper-parameters corresponding to the learning mechanism of individual blocks like denoising rate, dropout rate, gradient stepsizes, etc. Hence, for a given block we can learn  $T$  different number of learning models by randomly generating  $T$  different sets of such learning hyper-parameters.

This two-fold randomization will result in an ensemble of  $B \times T$  number of weak learners with as small correlation among them as possible. Clearly, increasing the number of weak learners by  $T > 1$ , decreases the variance of MVUB, while also mitigating the influence of learning parameters. This is the crux of our randomized deep networks. Specific details about the randomization process will be described in the following section as we present the architecture and training mechanisms for these models. Lastly, observe that it is easy to incorporate multi-modal features like MRI or PET within our setup. The simplest way would be to generate blocks from each of the modalities independently, train them separately, and combine their outputs at a later stage. If  $m$  denotes the number of modalities, then the blocks from each modality can simply be concatenated resulting in  $mB$  number of weak learners. Alternatively, one can construct ‘cross-modal’ blocks where voxels across multiple modalities are sampled and assigned to a single block. This procedure will have to take into account the model specific tessellating distributions and is far more complex than the intra-modal design mentioned earlier. Since the networks are trained locally on each block, once the blocks are fixed, the voxels within a block do not *directly* interact with those from the other. One can nevertheless construct a feedback procedure that reassigns voxels among blocks based on the goodness of predictions from the previous set of blocks. This feedback is computationally expensive and it may not improve the performance whenever the number and size of the blocks are reasonably large. We refer to the two proposed randomized deep network models as – randomized Denoising Autoencoders (rDA) and randomized Dropout Networks (rDr).

**FIGURE 15.2**

The architecture of randomized deep network is an ensemble of  $BT$  weak learners, where each weak learner is an  $L$ -layered MLNN. Each block processes fixed set of voxels  $s_b$  of length  $d_b$ . Within each block there are  $T$  MLNNs.

### 15.4.3 RDA AND RDR TRAINING

Let  $\mathcal{V} = \{v_1, \dots, v_d\}$  denote the set of voxels. Consider a probability distribution  $\tau(v)$  over the voxels  $v \in \mathcal{V}$ . This is the sampling distribution that governs the block construction, i.e., the assignment of voxels to blocks, and in the simplest case, it is a uniform distribution. For each block  $b = 1, \dots, B$ , we sample  $d_b \ll d$  (fixed a priori) number of voxels without replacement using the distribution  $\tau(v)$ . We call this set of voxels,  $s_b$ . Each block is presented with  $T$  different sets of learning parameters (i.e., denoising rate, gradient learning rate and so on) denoted by  $\theta_t \in \Theta$  for  $t = 1, \dots, T$ , where  $\Theta$  is the given hyper-parameter space. This means that each sample from the hyper-parameter space yields one weak learner, i.e., one SDA or Dropout network for one block. Hence, a total of  $B \times T$  number of weak learners are constructed. If  $\tau(\cdot)$  is uniform, then asymptotically we expect to see one voxel in at least one of the  $B$  blocks. As discussed earlier, instead of uniform  $\tau(\cdot)$ , alternative choices may be used depending on prior information about the importance of including a particular voxel in one/more blocks. Depending on  $\tau(\cdot)$  the blocks may be mutually exclusive. The influence of model hyper parameters including  $B, T$ , the number of voxels per block  $d_b$ , the sampling distribution  $\tau(\cdot)$ , and the robustness of the model to these choices are described in Section 15.4.3.1.

**Fig. 15.2** (left) shows the architecture of randomized deep network with  $B$  blocks, each with  $T$  weak learners, where each weak learner corresponds to an  $L$ -layered

MLNN (as shown in Fig. 15.2 (right)). The outputs from the ensemble of  $B \times T$  networks are combined using ridge regression. Algorithm 15.1 summarizes the training procedure. Given training data  $(\mathbf{x}_i, \mathbf{y}_i)$  for  $i \in \{1, \dots, n\}$ , we first learn the unknown transformations –  $(\mathbf{W}_{b,t}^l, p_{b,t}^l, q_{b,t}^l), \forall b \in \{1, \dots, B\}; t \in \{1, \dots, T\}; l \in \{1, \dots, L\}$ . Depending on whether the weak learner is an SDA or a dropout network, the learning process for these  $B \times T \times L$  transformations will follow the minimization of (15.4) and (15.5), or the discussion from Section 15.2.3.2, respectively. The *Reweigh*( $\cdot$ ) operation in Algorithm 15.1 skews the sampling distribution to ensure that the un-sampled voxels (i.e., voxels that are not assigned to any block, yet), will be given priority in the later blocks, while avoiding oversampling of the same set of voxels across multiple blocks.

Concatenating the  $L$ th layer outputs from  $B \times T$  learners, we get  $\mathbf{H}_i = [[\mathbf{h}_{b,t}^L]]_{1,1}^{B,T}$ . The weighted regression pooling then composes these outputs

$$\mathbf{U} \leftarrow (\mathbf{H}^T \mathbf{H} + \lambda \mathbf{I})^{-1} \mathbf{H}^T \mathbf{Y}; \quad \mathbf{H} = [[\mathbf{H}_i]]_1^n; \quad \mathbf{Y} = [[\mathbf{y}_i]]_1^n, \quad (15.7)$$

where  $\mathbf{U}$  are the regression coefficients,  $\lambda$  is the regularization constant and  $\mathbf{Y} = [\mathbf{y}_i]_1^n$ . Since the networks are already capable of learning complex concepts (see discussion from Section 15.2.2), the pooling operations that we used was a simple linear combination of the  $B \times T$  outputs with  $\ell_2$ -loss providing minimum mean squared error. Clearly, instead of regression or any other fancier combinations, a simple mean of the outputs may also suffice to generate the final output because the networks are already ensured to be as uncorrelated as possible, and so, their mean output is a reasonable estimate of the predicted labels. Observe that the training algorithm and the design of randomized deep networks in general are agnostic to the type of architectures that were used as weak learners. Once the randomized network is trained, its prediction on a new test instance/example is given by (the scaling up of transformations in dropout network case are not shown here to avoid notational clutter, see [33] for more details),

$$\hat{\mathbf{y}} = \mathbf{h}\mathbf{U}; \quad \mathbf{h} = [[\mathbf{z}_{b,t} \mathbf{h}_{b,t}^L]]_{1,1}^{B,T}; \quad \mathbf{h}_{b,t}^l = \sigma(\mathbf{W}_{b,t}^l \mathbf{h}_{b,t}^{l-1} + p_{b,t}^l); \quad \mathbf{h}_{b,t}^0 = \mathbf{x}. \quad (15.8)$$

---

**Algorithm 15.1** Randomized deep networks – blocks training

---

**Input:**  $\theta_t \sim \Theta, \mathcal{V}, B, s_B, L, T, \mathcal{D} \sim \{\mathbf{x}_i, \mathbf{y}_i\}_1^n, \lambda$

**Output:**  $(\mathbf{W}_{b,t}^l, p_{b,t}^l, q_{b,t}^l)$

**for**  $b = 1, \dots, B$  **do**

$I_b \sim \tau(\cdot)$

**for**  $t = 1, \dots, T$  **do**

$(\mathbf{W}_{b,t}^l, p_{b,t}^l, q_{b,t}^l) \leftarrow \mathcal{Z}(\mathcal{D}, L, I_b, \theta_t)$

**end for**

$\tau(\mathcal{V}) \leftarrow \text{Reweigh}(\tau(\mathcal{V}), I_b)$

**end for**

---

### 15.4.3.1 Hyperparameters

The hyperparameters of randomized deep network include:

- $B$ , the number of blocks
- $T$ , the number of hyperparameter sets; depth of the network, gradient learning rate, denoising (for rDA) or dropout (for rDr) rates (or other appropriate regularization criteria depending on the weak learning mechanism), number of gradient iterations, etc.
- $\tau(\cdot)$ , the sampling distribution over all the voxels for constructing the blocks,
- $d_b$ , the number of voxels within each block (or length of the input layer for weak learners)
- $\lambda$ , the regularization parameter for ridge regression

Observe that, within each block, the randomization is over the  $T$  sets of learning parameters, and hence if the hyperparameter space is sampled uniformly, the model's outputs will be robust to changes in  $T$ . The simplest choice for the block-wise sampler  $\tau(\cdot)$  assigns uniform probability over all dimensions/voxels as described above. However, we can assign large weights on local neighborhoods which are more sensitive to the disease progression, if desired. We can also setup  $\tau(\cdot)$  based on entropy or the result of a hypothesis test. More precisely, entropic measures (like Kullback–Leibler divergence), t-scores or z-scores can be used to estimate the discrimination power of each voxel (this would correspond to performing  $V$  number of hypothesis tests; uncorrected). The resulting scores (being positive) can then be normalized and used as the sampling distribution  $\tau(\cdot)$ . The *Reweigh(·)* step in [Algorithm 15.1](#) takes care of previously unsampled dimensions/voxels. The simplest such re-weighting includes removing the already sampled voxels from  $\tau(\cdot)$  (which is the same as sampling voxels without replacement). Although there is no analytic way to setup  $B$  and  $d_b$  (for  $b = 1, \dots, B$ ), a reasonably large number of blocks with  $d_b = d/B$  would suffice (refer to discussion in [Section 15.5](#) about the choices made in our experiments). The influence of dropout and denoising rates on the performance of deep networks have been well studied empirically [[32,33](#)], and [[41](#)] analyze the dynamics of the dropout networks as the dropout rate changes. Since such studies already provide ample evidence for setting these rates, we do not explicitly analyze them for our setting.

### 15.4.4 THE DISEASE MARKERS – RDAM AND RDRM

The randomized deep network from [Fig. 15.2](#) and [Algorithm 15.1](#) will now be adapted to the problem of designing an MVUB of disease spectrum (refer back to our discussion from [Section 15.3](#)). Depending on the choice of architectures used, these MVUBs will be referred to as randomized Denoising Autoencoder marker (rDAm) or randomized Dropout network marker (rDrm). The sigmoid nonlinearity ensures that the outputs of individual blocks lie in  $[0, 1]$ , and so the predictions from the rDA and rNr models on new test examples are bounded between 0 and 1. This is clearly advantageous since a bounded predictor would implicitly guarantee bounded

variance (a desirable property from the perspective of MVUB design; refer to (15.6) and Section 15.3.1) while also covering the entire disease spectrum. Hence, we only need to ensure that each individual block is an unbiased estimate of the disease. We then train the weak learners (the blocks) *only* using healthy controls and completely diseased subjects each labeled  $y = 1$  and  $y = 0$ , respectively. Here the diseased subjects would be those with clinical AD, and all subjects in other stages of the disease (like early or late MCI [9–11]) are not going to be used for training. Since the pooling operation corresponds to a regression (see (15.8)), the test time predictions are the desired disease markers rDAm or rDrm. They will correspond to the *confidence* of rDA or rDr in predicting the subject’s decline – closer to 1 if the subject is healthy, or 0 otherwise. Clearly, changing the training setup from regression to classification will modify the interpretation of these predicted markers, but their properties like bounded variance and MVUB would still remain the same.

Recall the discussion in Section 15.2.1 about the sample enrichment procedure where the inclusion criterion decides whether the subject needs to be enrolled in the trial or not. As noted in Section 15.1, from the practical perspective, the inclusion criterion should filter subjects at the trial start point itself. Specifically, the inclusion criterion, either rDAm or rDrm computed at the *baseline* (or trial start point) will then be used to enroll the subjects. Since the markers are bounded between 0 and 1, the enrichment is driven by choosing the “appropriate” *threshold* or cut-off to retain subjects accordingly. This procedure is summarized here:

1. The first check for performing this baseline sample enrichment is to ensure that these markers, computed at the baseline, have strong correlation (or dependence) with other disease markers, some of which would indeed be the intended trial outcomes [50,51]. If the dependencies turn out to be significant, this is evidence of convergent validity, and using baseline rDAm or rDrm as inclusion criteria for enriching the trial population is, at minimum, meaningful.
2. Once this is the case, using the enrichment threshold  $t$  ( $0 < t < 1$ ), the enriched cohort would include only those subjects whose baseline rDAm or rDrm is smaller than  $t$  (closer to being diseased). Alternatively, by avoiding to choose the optimal cut-off  $t$ , one can instead include a fixed fraction (e.g., 1/4th or 1/3rd) of the whole population whose baseline rDAm or rDrm is closest to 0 (fixing a fraction automatically fixes  $t$ ).

Clearly, the choice of  $t$  is vital here, and one way to choose it is by comparing the mean longitudinal change of some disease markers (MMSE, CDR, and so on) for the enriched cohort as  $t$  goes from 0 to 1. The optimal  $t$  would correspond to a discontinuity or a “bump” in the change trends as  $t$  increases. We discuss more on these issues as we present evaluate rDA and rDr.

---

## 15.5 EXPERIMENTS

### 15.5.1 PARTICIPANT DATA AND PREPROCESSING

Imaging data including [<sup>18</sup>F]Florbetapir amyloid PET (AV45) singular uptake value ratios (SUVR), FDG PET SUVRs and gray matter tissue probability maps derived from T1-weighted MRI data, and several neuropsychological measures and CSF values from 516 individuals enrolled in Alzheimer’s Disease Neuroimaging Initiative-II (ADNI2)<sup>1</sup> were used in our evaluations. Of these 516 persons (age  $72.46 \pm 6.8$ , female 38%), 101 were classified as AD (age  $75.5 \pm 5.1$ ), 148 as healthy controls (age  $70.75 \pm 7$ ), and 131 and 136 as early and late MCI (age  $74.3 \pm 7.1$  and  $75.9 \pm 7.7$ ), respectively, at baseline. There was a significant age difference across the four groups with  $F > 10$  and  $p < 0.001$ . Among the MCIs, 174 had positive family history (FH) for dementia, and 141 had at least one Apolipoprotein E (APOE) e4 allele. CSF measures were only available at baseline, and three time point data (baseline, 12 months, and 24 months) was used for the rest. The imaging protocols follow the standards put forth by ADNI. MRI images are MP-RAGE/IR-SPGR from a 3T scanner. PET images are 3D scans consisting of four 5-min frames<sup>2</sup> from 50 to 70 min post-injection for [<sup>18</sup>F]Florbetapir PET, and six 5-min frames from 30 to 60 min post injection for FDG PET. Modulated gray matter tissue probability maps were segmented from the T1-weighted MRI images (other tissue maps are not used in our experiments) using voxel-based morphometry [52]. The segmented map was then normalized to Montreal Neurological Institute (MNI) space, smoothed using 8 mm Gaussian kernel, and the resulting map was thresholded at 0.25 to compute the final gray matter image. All PET images were first co-registered to the corresponding T1 images, and then normalized to the MNI space. Manually constructed masks of pons, vermis, and cerebellum were then used to scale these PET maps by the average intensities in pons and vermis (FDG PET SUVR) and cerebellum (Florbetapir PET SUVR). All preprocessing was done in SPM8 [53].

### 15.5.2 EVALUATIONS SETUP

We train the randomized deep networks using only baseline imaging data from all the three modalities, MRI, FDG PET, and AV45 PET with diseased (AD, labeled 0) and healthy (CN, cognitively normal, labeled 1) subjects. When testing on MCI subjects, these trained models output a multi-modal rDAm and rDrm, which represent the confidence of rDA and rDr that a given MCI subject is (or is not) likely to decline. We only use baseline imaging data for training, thereby making the models deployable in practice, while the predictions can be performed on MCIs at baseline and/or future time-points.  $B = 5000$ ,  $d_b = d/B$  (i.e., each voxel appears in only one block), and  $\lambda = 1$  for both rDA and rDr.  $\tau(\cdot)$  is based on differentiating ADs and CNs using KL divergence (refer to Section 15.4.3.1). Multiple combinations of  $B$ ,  $d_b$  and  $\tau(\cdot)$  (including uniform and t-score based) were also evaluated, however, none of them gave any significant improvements over the above settings. The blocks construction for multi-modal rDA and rDr did not use cross-modal sampling to ensure manage-

able computational burden (see Section 15.4.2). Within this setup, our evaluations are three-fold. Since the test data are MCIs, we first evaluate if baseline rDAm and rDrm differentiate early MCI from late MCI, and parental family history as a contributing risk factor. These evaluations include the baseline markers derived from seven different combinations corresponding to the three imaging modalities available.

After checking the construct that multi-modal markers are superior to unimodal markers, we evaluate the premise whether the multi-modal rDA and rDr markers are good disease progression markers. We demonstrate this by computing the dependence of these multi-modal baseline rDAm and rDrm with well-known outcome measures including, Mini Mental State Examination (MMSE), Alzheimer’s Disease Assessment Scale (ADAS Cognition 13), Montreal Cognitive Assessment (MOCA), Rey Auditory Verbal Learning Test (RAVLT), neuropsychological summary score for Memory (PsyMEM), summary score for Executive Function (PsyEF), hippocampal volume from gray matter images, Clinical Dementia Rating sum of boxes (CDR-SB), a binary marker for conversion from MCI to AD (DxConv), CSF levels including Tau  $\tau$ , Phospho-Tau  $p\tau$ , Amyloid Beta  $A\beta$ 42, ratios of  $\tau$  and  $p\tau$  with  $A\beta$ 42, APOE allele 4 and maternal/paternal family history (FH). Please see [54,10,51] and other appropriate references<sup>3</sup> therein for complete details about these disease and at-risk markers. For continuous markers, we used the Spearman Rank Order Correlation coefficient to assess the dependencies and accepted those statistics as significant whenever the corresponding  $p < 0.05$ . For binary markers the  $t$ -test was used with the same significance value. Observe that we are interested in evaluating the predictive power of baseline rDAm and rDrm. Specifically, we report the correlations of baseline rDAm with these markers at 12 months and 24 months, and also the longitudinal change with this one year, thereby providing evidence that whenever the baseline markers are closer to 0, the subject’s longitudinal changes are, in fact, steeper/stronger.

Once the correlation construct is appropriately validated, we evaluate the use of baseline rDAm and rDrm for sample enrichment. We compute the sample sizes (using (15.1) based on the discussion in Section 15.2.1) required when using the above cognitive, neuropsychological, diagnostic and other imaging-based outcome measures with (and without) rDAm or rDrm based enrichment. The sample size trends are computed across different enrichment thresholds (see Section 15.4.4). For better interpretation of the estimates from the perspective of a practitioner or clinician, we estimate the effect size as a function of the marker enrichment cut-off for a given (fixed) sample size. We also compute the performance improvement from using our markers relative to the alternative imaging-derived enrichers including ROI summaries from FDG and florbetapir images<sup>4</sup> with particular attention to the current state-of-the-art imaging based computational summary measure that we refer to as a Multi-Kernel Learning marker [4]. MKLm is based on a Multi-Kernel support vector machine (MKL) [4] that tries to harmonize contributions from multiple imaging modalities for deriving a maximum margin classifier in the concatenated Hilbert spaces. Unlike traditional support vector machines, MKLm uses a linear combination of kernels and solves for both the weights on the kernels as well as the normal to the

**Table 15.1** rDA and rDr vs. MKLm. A, amyloid; F, FDG; and T, T1GM

Model	Amyloid	FDG	T1GM	A+F	A+T	F+T	A+F+T
(A) Early versus late MCI							
MKL	20.5 <sup>†</sup>	16.8 <sup>†</sup>	16.5 <sup>†</sup>	16.4 <sup>†</sup>	20.4 <sup>†</sup>	23.6*	27.9*
rDA	22.1*	9.7 <sup>†</sup>	20.0 <sup>†</sup>	19.5 <sup>†</sup>	24.1*	21.2*	27.6*
rDr	20.1 <sup>†</sup>	11.5 <sup>†</sup>	20.3 <sup>†</sup>	17.5 <sup>†</sup>	23.0*	21.2*	26.9*
(B) Family history: positive versus negative							
MKL	4.3**	7.5 <sup>†</sup>	5.3**	7.3 <sup>‡</sup>	6.8 <sup>‡</sup>	6.6**	8.3 <sup>‡</sup>
rDA	4.7**	11.8 <sup>†</sup>	11.2 <sup>†</sup>	6.8 <sup>†</sup>	12.4 <sup>†</sup>	13.2 <sup>†</sup>	13.3 <sup>†</sup>
rDr	4.6**	9.9 <sup>†</sup>	12.0 <sup>†</sup>	6.9 <sup>†</sup>	11.7 <sup>†</sup>	13.2 <sup>†</sup>	12.0 <sup>†</sup>

hyper-plane concurrently [4]. Similar to the proposed models, MKL is trained using AD and CN subjects, and the corresponding predictions on MCIs are referred to as MKL markers (MKLm). Note that whenever the labels correspond to continuous predictors instead of class indices (like AD vs. CN), we use  $\epsilon$ -support vector machine version of MKL.

### 15.5.3 RESULTS

**Table 15.1** shows the discrimination power of rDA and rDr for classifying early and late MCI (**Table 15.1A**) and family history (**Table 15.1B**). Both rDA and rDr perform better than the baseline MKL. **Tables 15.2 and 15.3** correspond to the predictive power of baseline rDAm and rDrm, respectively. They show the Spearman correlations and t-statistics of the baseline multi-modal markers with cross-sectional scores (baseline, 12 and 24 months) and longitudinal change (12 and 24 months) in other disease markers. Negative correlations indicate that the corresponding measures (ADAS,  $\tau$ ,  $p\tau$ ,  $\tau/A\beta42$ ,  $p\tau/A\beta42$ ) increase with disease progression. Across both rDAm and rDrm, large correlations with  $r > 0.45$  and/or  $p < 10^{-4}$  (denoted by the superscript \*), were observed with baseline summary measures (see the second column in **Tables 15.2 and 15.3**), specifically with ADAS, neuropsychological (memory and executive function) composite scores, hippocampal volume, and CSF levels involving  $A\beta42$ . For both rDA and rDr, FH had a smaller influence on baseline rDAm compared to APOE. All the cross-sectional correlations (columns 2–4, **Tables 15.2 and 15.3**) were significant ( $p < 10^{-4}$ ). The correlations of baseline markers with longitudinal change (last two columns) were significant with  $r > 0.21$  and  $p < 0.001$  for all the measures (except few cases involving PsyEF and MOCA). rDAm’s correlations are stronger than that of rDrm’s across all the constructs. **Tables 15.1–15.3** provide strong evidence for both rDAm’s and rDrm’s disease predictive power.

**Table 15.4**, **Figs. 15.3 and 15.4** show the relevance of rDA and rDr for enrichment. **Table 15.4** shows the coefficient of variation (CV, the ratio of standard deviation to mean) of rDAm and rDrm for three different populations of interest – all MCIs, late MCIs and MCIs with positive FH. The proposed markers’ CV is smaller than MKLm for all three populations and combinations of modalities – making it a better candi-

**Table 15.2** Correlations of multi-modal baseline rDAm

Biomarker	Baseline	Cross-sectional	Longitudinal change	
MMSE	0.39*	0.49*	0.45*	0.21 <sup>†</sup> 0.33 <sup>‡</sup>
ADAS	-0.56*	-0.58*	-0.53*	0.21 <sup>†</sup> -0.53 <sup>‡</sup>
MOCA	0.48*	0.51*	0.59*	0.06 0.59*
RAVLT	0.49*	0.52*	0.57*	0.13** 0.57 <sup>†</sup>
PsyMEM	0.56*	0.57*	0.59*	0.28* 0.42 <sup>†</sup>
PsyEF	0.52*	0.57*	0.46*	0.15** 0.26**
HippoVol	0.72*	0.74*	0.79*	0.33* 0.47*
CDR-SB	-0.33*	-0.49*	-0.55*	-0.36* -0.53*
DxConv	—	21*	31*	21* 31*
$\tau$	-0.39*	—	—	—
$p\tau$	-0.40*	—	—	—
$A\beta42$	0.55*	—	—	—
$\tau/A\beta42$	-0.52*	—	—	—
$p\tau/A\beta42$	-0.52*	—	—	—
APOE	3.47 <sup>†</sup>	—	—	—
FH	2.16**	—	—	—

**Table 15.3** Correlations of multi-modal baseline rDrm

Biomarker	Baseline	Cross-sectional	Longitudinal change	
MMSE	0.37*	0.41*	0.31*	0.21* 0.25 <sup>‡</sup>
ADAS	-0.44*	-0.42*	-0.34*	-0.22* -0.18**
MOCA	0.41*	0.35*	0.33*	0.09 0.29 <sup>‡</sup>
RAVLT	0.43*	0.34*	0.27 <sup>‡</sup>	0.11 0.20 <sup>‡</sup>
PsyMEM	0.48*	0.39*	0.34 <sup>‡</sup>	0.22* 0.29 <sup>‡</sup>
PsyEF	0.45*	0.38*	0.22	0.13** 0.10
HippoVol	0.62*	0.50*	0.25 <sup>†</sup>	0.22 <sup>†</sup> 0.15**
CDR-SB	-0.33*	-0.35*	-0.35*	-0.30* -0.31*
DxConv	—	17*	17*	11 <sup>†</sup> 10 <sup>†</sup>
$\tau$	-0.34*	—	—	—
$p\tau$	-0.35*	—	—	—
$A\beta42$	0.50*	—	—	—
$\tau/A\beta42$	-0.46*	—	—	—
$p\tau/A\beta42$	-0.46*	—	—	—
APOE	6.9 <sup>‡</sup>	—	—	—
FH	5.01**	—	—	—

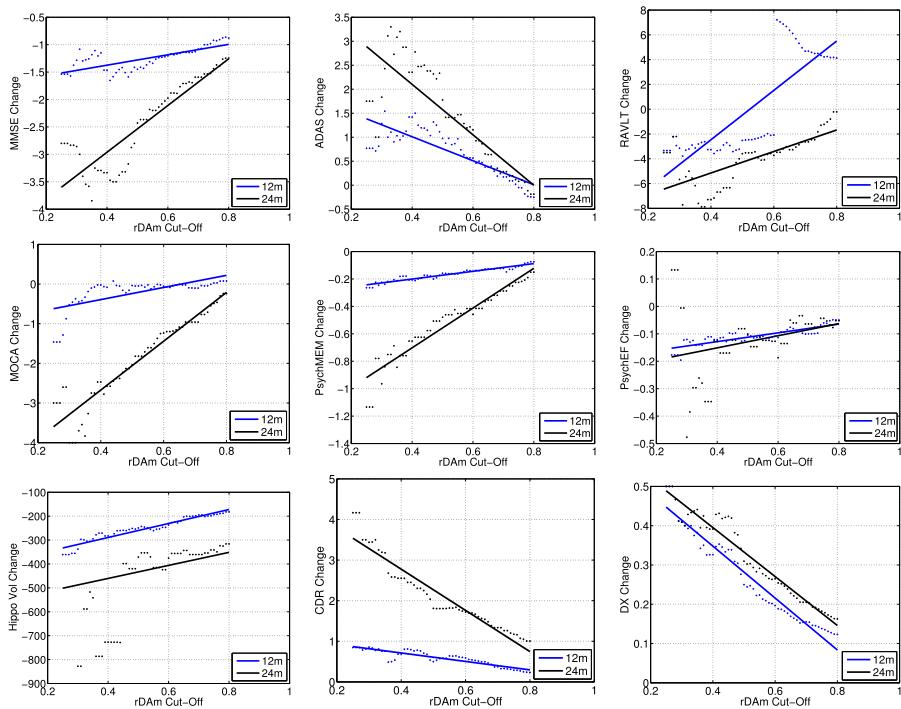
date to be used as a prediction measure as well as an enricher (refer to (15.1), where the right hand side includes terms depending on  $CV^2$ ). CV of rDAm is smaller than rDrm. Each plot in Figs. 15.3 and 15.4 corresponds to the mean longitudinal change of some disease marker, *after* the MCI population (the test set) is enriched by removing the weak decliners (subjects with baseline rDAm or rDrm above a certain

**Table 15.4** CV of rDAm and rDrm vs. MKLm

<b>Modality</b>	<b>Marker</b>	<b>MCIs</b>	<b>LMCIs</b>	<b>FHMCIs</b>
Amyloid	MKLm	0.56	0.70	0.42
	rDAm	0.49	0.57	0.41
	rDrm	0.55	0.60	0.46
FDG	MKLm	0.49	0.53	0.39
	rDAm	0.33	0.36	0.26
	rDrm	0.45	0.44	0.30
T1MRI	MKLm	0.55	0.60	0.48
	rDAm	0.36	0.42	0.26
	rDrm	0.41	0.42	0.26
A+F	MKLm	0.52	0.63	0.39
	rDAm	0.42	0.49	0.33
	rDrm	0.50	0.57	0.39
A+T	MKLm	0.56	0.67	0.42
	rDAm	0.41	0.49	0.29
	rDrm	0.50	0.51	0.29
F+T	MKLm	0.51	0.58	0.41
	rDAm	0.34	0.38	0.25
	rDrm	0.41	0.44	0.31
A+F+T	MKLm	0.54	0.65	0.39
	rDAm	0.41	0.50	0.28
	rDrm	0.44	0.57	0.30

cut-off  $t$ , shown on the  $x$ -axis). For both rDA and rDr, the plots show that MMSE, MOCA, hippocampal volume, CDR-SB, and DxConv have large changes when weak decliners are progressively removed – strong evidence for rDAm’s and rDrm’s predictive power. Specifically, for some measures the changes are much steeper for 24 months than 12 months (black and blue lines in each plot). PsyEF resulted in irregular changes at different time points for both rDA and rDr.

Tables 15.5–15.8 show sample sizes when multi-modal baseline rDAm and rDrm are used as enrichers, at 80% statistical power and 0.05 significance level, with a hypothesized treatment effect of  $\eta = 0.25$  (i.e., 25% decrease in the disease). Tables 15.5 and 15.6 show the sample sizes at four different rDAm and rDrm enrichment cut-offs (third to last columns), respectively. Recall that higher values of the markers at baseline imply closer to being healthy. Hence, enrichment entails to filtering out all subjects whose baseline rDAm or rDrm are above some chosen cut-off. Results show that, compared to the no-enrichment regime (second column), the sample estimates from rDAm or rDrm enrichment are significantly smaller, with more than 5 times reduction when using bottom 20% and 25% percentiles (third and fourth columns).

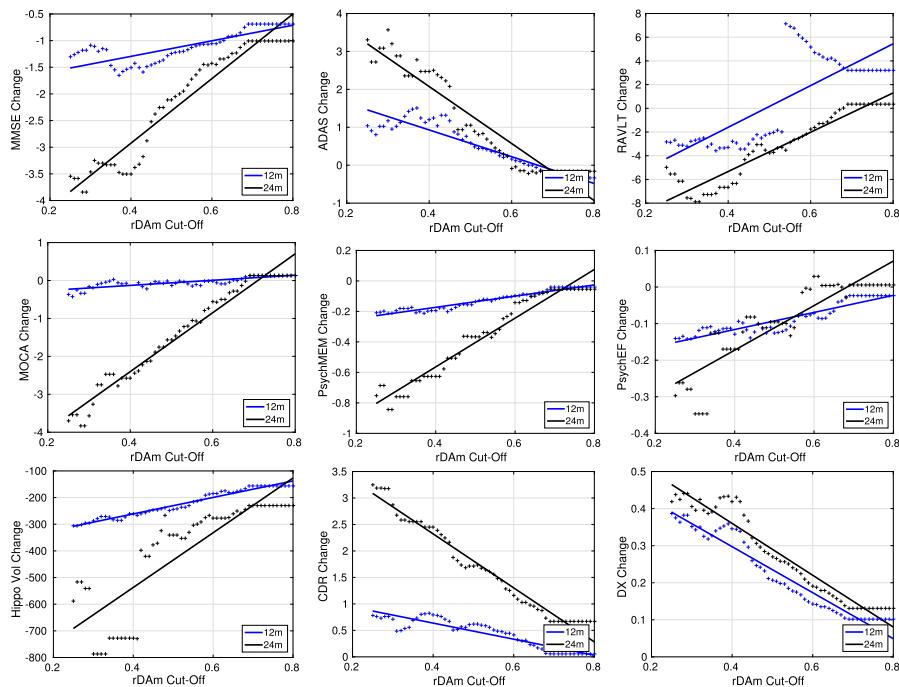
**FIGURE 15.3**

Longitudinal change of measures vs. multi-modal baseline rDAm.

**Table 15.5** Sample sizes with multi-modal baseline rDAm enrichment

Outcome measure	No enrichment	Bottom 20% rDAm $\leq 0.41$	Bottom 25% rDAm $\leq 0.46$	Bottom 33% rDAm $\leq 0.52$	Bottom 50% rDAm $\leq 0.65$
MMSE	1367	200	239	371	566
ADAS	>2000	775	945	>2000	>2000
MOCA	>2000	449	674	960	1919
RAVLT	>2000	591	1211	>2000	>2000
PsyMEM	>2000	420	690	786	1164
PsyEF	>2000	>2000	>2000	>2000	>2000
HippoVol	>2000	543	1504	1560	1675
CDR-SB	1586	281	317	430	433
DxConv	895	230	267	352	448

The sample sizes from rDAm enrichment are smaller than those from rDrm, following the previous observation that the CV's of rDrm are larger (refer to Table 15.4). In particular, with rDAm; MMSE, CDR-SB, and DxConv give consistently smaller

**FIGURE 15.4**

Longitudinal Change of measures vs. multi-modal baseline rDrm.

**Table 15.6** Sample sizes with multi-modal baseline rDrm enrichment

<b>Outcome measure</b>	<b>No enrichment</b>	<b>Bottom 20% rDAm <math>\leq 0.39</math></b>	<b>Bottom 25% rDAm <math>\leq 0.44</math></b>	<b>Bottom 33% rDAm <math>\leq 0.58</math></b>	<b>Bottom 50% rDAm <math>\leq 0.70</math></b>
MMSE	1367	252	341	394	560
ADAS	>2000	930	1770	>2000	>2000
MOCA	>2000	655	795	1106	1866
RAVLT	>2000	1556	>2000	>2000	>2000
PsyMEM	>2000	442	700	799	1215
PsyEF	>2000	>2000	>2000	>2000	>2000
HippoVol	>2000	621	1100	1846	>2000
CDR-SB	1586	307	524	608	618
DxConv	895	232	287	307	429

estimates (200 to 600) across all columns (the four different percentiles). ADAS and PsychEF still required very large sizes (774 and >2000, respectively) even at 20% enrichment percentile. Similar trends are observed for rDrm.

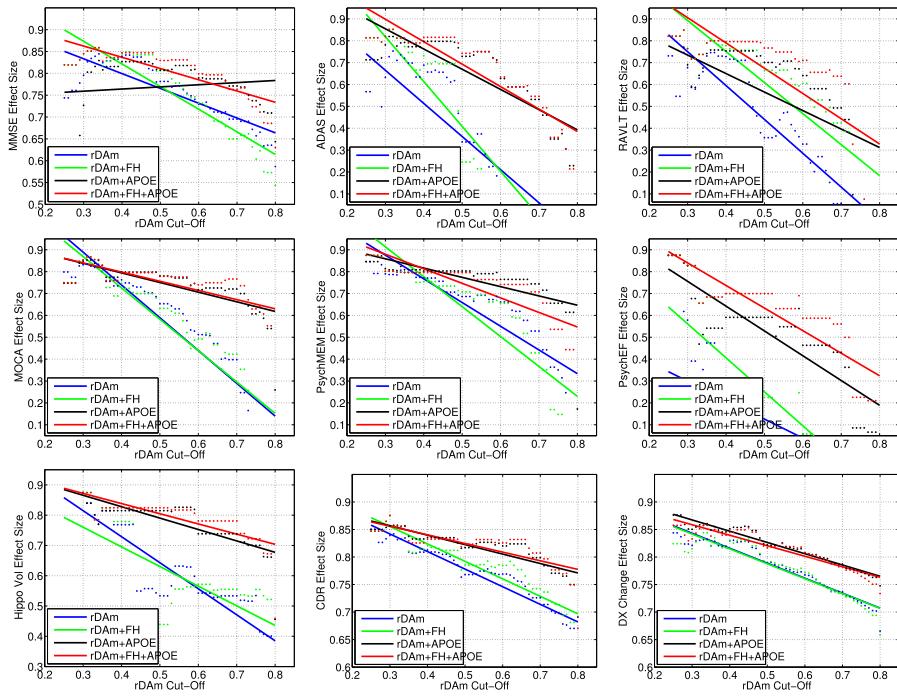
**Table 15.7** Sample sizes with rDAm + FH and/or APOE enrichment

Outcome measure	No enrichment	FH only	APOE only	rDAm only	rDAm + FH	rDAm + APOE	rDAm + both
MMSE	1367	1668	1015	200	182	240	186
ADAS	>2000	>2000	>2000	775	574	328	271
MOCA	>2000	>2000	>2000	449	516	326	334
RAVLT	>2000	>2000	>2000	591	394	484	332
PsyMEM	>2000	>2000	>2000	420	481	310	333
PsyEF	>2000	>2000	>2000	>2000	>2000	1337	721
HippoVol	>2000	>2000	>2000	428	391	274	246
CDR-SB	1586	1787	763	281	255	217	225
DxConv	895	932	509	230	244	170	192

**Table 15.8** Sample sizes with rDrm + FH and/or APOE enrichment

Outcome measure	No enrichment	FH only	APOE only	rDrm only	rDrm + FH	rDrm + APOE	rDrm + both
MMSE	1367	1668	1015	252	292	301	306
ADAS	>2000	>2000	>2000	930	1001	1151	642
MOCA	>2000	>2000	>2000	655	669	636	669
RAVLT	>2000	>2000	>2000	1556	1102	>2000	1544
PsyMEM	>2000	>2000	>2000	442	496	512	385
PsyEF	>2000	>2000	>2000	>2000	>2000	>2000	941
HippoVol	>2000	>2000	>2000	621	799	698	698
CDR-SB	1586	1787	763	307	316	351	392
DxConv	895	932	509	232	259	219	239

Tables 15.5 and 15.6 further enrich the population after using rDrm and rDrm with extra covariate information like FH and/or APOE status. Specifically, here we explicitly filter out those MCI subjects who are *not* FH and/or APOE positive before performing baseline rDAm or rDrm enrichment. Clearly, the sample sizes further decrease because of this extra filtration as shown in last three columns of Tables 15.5 and 15.6. APOE as a covariate resulted in smallest possible estimates in general (<350 per arm with MMSE, CDR-SB, and DxConv outcomes) across all the outcomes except PsyEF (last two columns in Table 3). Interestingly, they are smaller than the sample sizes from using both APOE and FH as covariates (last column). DxConv as an outcome with rDAm or rDrm + APOE enrichment yields a sample size of 170 and 219, respectively. Figs. 15.5 and 15.6 show the detectable effect sizes as rDAm and rDrm enrichment cut-offs are varied for a fixed sample size of 500 per arm. The detectable effect size ( $1 - \eta$ ) decreases as more weak decliners are filtered out. This can be seen by the increase of  $\eta$  (y-axis) as the rDAm or rDrm cut-offs on x-axis decrease, specifically for MMSE, CDR-SB, and DxConv outcomes. These plots are very useful from a clinician's or practitioner's perspective, as will be dis-

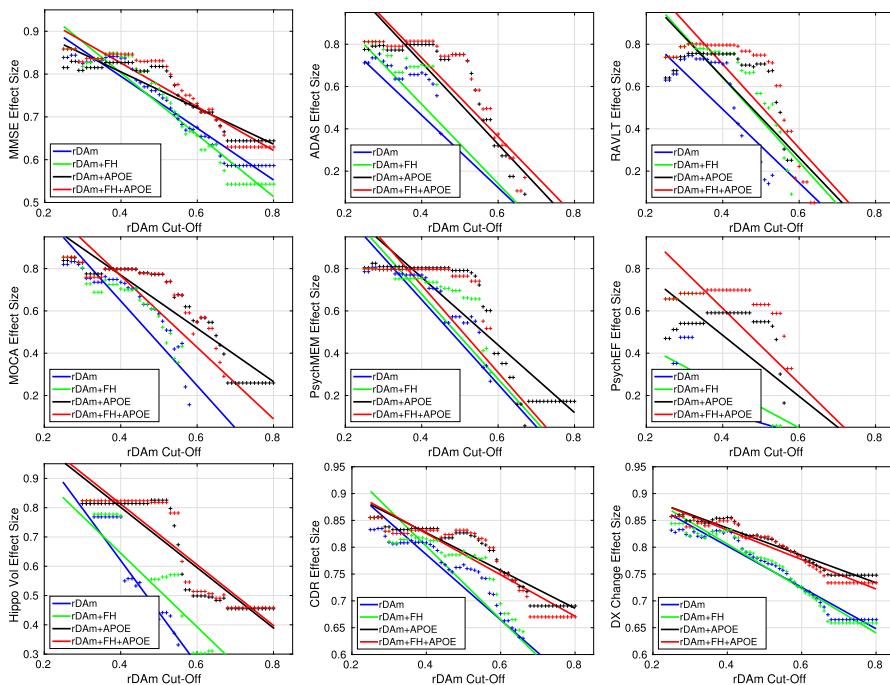
**FIGURE 15.5**

Effect sizes vs. multi-modal baseline rDAm cut-offs.

cussed later in Section 15.6. Finally, Table 15.9 compares our proposed enrichers with other imaging-derived inclusion criteria (the cut-off for all the enrichers corresponds to including the strongest 20% decliners in their respective scales). Both rDAm and rDrm consistently outperformed other alternatives (and between them rDAm was better), with up to 2 times smaller estimates than MKLm, and much larger reductions compared to uni-modal summaries (hippocampal volume, FDG and AV45 ROIs).

## 15.6 DISCUSSION

The ability to design clinical trials with smaller sample sizes but sufficient statistical power will enable the implementation of affordable, tractable, and hopefully, conclusive trials. Efficiency is seriously compromised in trials where there is poor biomarker specificity of disease progression and when the outcomes contain relatively high amounts of error variance. Determining whether promising treatments are effective in the MCI phase of AD requires accurate identification and inclusion of only those MCI participants most likely to convert to AD and selection of outcomes that are both disease related and possess optimal measurement properties. We have

**FIGURE 15.6**

Effect sizes vs. multi-modal baseline rDAm and rDrm vs. other enrichers.

**Table 15.9** Multi-modal baseline rDAm and rDrm vs. other enrichers

Sample enricher	Outcome measure							
	MMSE	ADAS	MOCA	RAVLT	PsyMEM	HippoVol	CDR-SB	DxConv
HippoVol	540	>2000	1005	1606	1009	>2000	389	420
FDG	384	1954	579	>2000	832	752	415	371
AV45	224	>2000	875	>2000	826	698	382	443
FDH	296	>2000	705	>2000	826	722	397	402
MKLM	228	874	827	896	487	877	295	284
rDAm	200	775	449	591	420	543	281	230
rDrm	252	930	655	1556	442	621	524	287

shown that the sample size required to detect a treatment effect can be substantially reduced using the proposed inclusion strategy. The central message of our empirical evaluations is that the multi-modal markers based on our proposed randomized deep network leaning models have good predictive power in identifying future disease progression, as shown in Tables 15.1–15.3 and Figs. 15.3–15.4. Together with the rDA's or rDr's capacity to reduce prediction variance (Table 15.4), we see smaller sample estimates compared to existing imaging-derived enrichers, as shown in Table 15.9, across many trial outcomes.

[Tables 15.1–15.3](#) support the general consensus that imaging data captures disease progression [4,51]. This can be seen from the very strong correlations of baseline rDAm and rDrm with cross-sectional and longitudinal changes in several cognitive scores (last four columns). It should be noted that high correlations with hippocampal volume across all time-points, which is a structural summary from MRI image, are expected because T1 MRI images at baseline is used in the construction of the markers. Although hippocampus voxels are used in rDA and rDr, its inclusion as an outcome in our experiments is primarily for completeness and continuity with existing AD imaging studies, where it has been extensively studied [6,18,50,51]. Interestingly, FH had a lower dependence on the baseline markers which might be because its influence is superseded by actual neurodegeneration once a subject reaches MCI stage (i.e., FH may play a much stronger role in the asymptomatic phase). Note that we did not correct for age (and other covariates like brain volume) because the markers reported in [Tables 15.1–15.3](#) are used directly with no covariate correction in our later evaluations on sample enrichment ([Tables 15.5–15.8](#)). This is based on the assumption that an actual RCT would not need to correct for the individual's age to evaluate eligibility and the baseline markers are agnostic to all such variables.

Observe that most classification based measures which are used as computational disease markers are generally unbounded [5]. These include the prediction score from an SVM based classification model on a test subject [4], or summary measures like S-score, t-score, F-score, etc. [6]. Unlike these measures, the proposed markers are bounded between 0 and 1, using which we can visualize its predictive power without any post-hoc normalization (as shown in [Figs. 15.3–15.4](#)). Except for PsyEF, all other measures used as outcomes had steeper changes (in [Fig. 15.3](#)) over time as baseline rDAm decreased, and in none of the cases was there a clear elbow separating weak and strong decliners. Similar trend is also observed for baseline rDrm. This shows that the disease progression is gradual from healthy to AD, and any classifications (like early and late MCI) are mostly artificial – an observations made earlier in alternate studies on decline [50,54,12]. Nevertheless, it is of clinical interest to analyze different groups of decliners like late and early MCI, and although the baseline MKLm (the current state-of-the-art in AD classification [4]) picks up these group differences as well, the proposed models have higher delineation power ([Table 15.1](#)). In particular, the *p*-values for rDAm and rDrm for FH+ vs. FH– case are an order of magnitude smaller than MKLm. These show that, in terms of classification accuracy, rDA and rDr are at least as good as a current state-of-the-art machine learning derived measures.

It is interesting to see rDAm's and rDrm's high predictive power for DxConv ([Tables 15.2–15.3](#) and [Figs. 15.3–15.4](#)), implying that subjects with smaller baseline rDAm (closer to 0) have very high likelihood of converting from MCI to AD, providing additional evidence that both baseline (and multi-modal) rDAm and rDrm are good predictive disease markers; more so, the predictions from randomized deep networks are good disease markers. Beyond the predictive power, the CVs for proposed models are much smaller than MKLm ([Table 15.4](#)) – the central argument that motivated the design of randomized deep networks for sample enrichment (refer to

Section 15.3). Using CV as a surrogate for prediction variance gives interesting inferences about the stages of the disease, for instance, the CVs for MCIs with FH+ are smaller than that of late MCIs (from Table 15.4). This suggests that a significant number of late MCIs currently have only a mild dementia in terms of both rDAm and rDrm. Most of the prediction power and sample size experiments focused on the multi-modal rDAm and rDrm (using all three modalities – amyloid and FDG PET and T1 MRI) since several existing studies including [4,6,2,55] and many others, and the performance results in Table 15.1, have shown the non-trivial benefit of multi-modal disease markers.

Since the proposed markers are lower bounded to 0 and no elbows are seen in Figs. 15.3–15.4, there is no phase change, and we can always select a fixed fraction of subjects that are closest to 0 on the baseline rDAm or rDrm scales, and claim that they are the strong decliners that should be included in a trial. The exact value of such fraction would depend on the logistics and size of the intended trial. This is the reason for the bottom-fraction based enrichment using multi-modal baseline markers as shown in Tables 15.5–15.9. Further, the high predictive power of baseline rDAm and rDrm solves an important bottleneck with existing approaches to designing inclusion criteria that use longitudinal data (e.g., tensor-based morphometry) [14,5]. Deploying such methods in practice implies that the trial screening time should be at least a year or longer, which is not practical (both in terms of cost involved and other logistics). Although longitudinal signals are much stronger than cross-sectional ones, the results in Tables 15.1–15.3 and Figs. 15.3–15.4 show that the randomized deep network based markers at trial start-point can still be used with no loss of information, saving trial resources and reducing the cost of trial setup.

A broad observation across Tables 15.5–15.9 is that baseline rDAm is better than rDrm with smaller sample sizes overall, although the trends are the same for both. Although both SDA and dropout network work with feature denoising, rDA seems to be clearly better at disease prediction as well (Tables 15.1–15.3). This higher sample estimates for rDr might be driven by its higher prediction variance (Table 15.4). Few reasons for this broad trend are discussed here. First observe that rDr lacks the unsupervised pretraining step unlike rDA (refer to Sections 15.2.3.2 and 15.4.3). Secondly, higher dropout rates might be “unsuitable” for brain images unlike vision or other machine learning datasets. To see this, observe that although complex interactions across voxels/dimensions are common in brain images, they are, nevertheless, registered to a common coordinate space (i.e., unlike object recognition or categorization data, a specific set of voxels in rDA and rDr always correspond to a specific region). In this registered space, the signals on brain images (voxel intensities) are, in general, very weak, and subtle changes in them will correspond to a disease signature [56,4,57,54]. For example, the disease signature in hippocampal region corresponds to loss (or dampening) of voxel intensities in a certain manner. Large dropout rates corrupts the images drastically resulting in loss of signal, and in the worst case, the dropped network from healthy subject might *look very similar to* being a diseased one. Unlike rDr, the post-hoc fine tuning in rDA (which does not involve corruption) compensates for these issues, thereby resulting in better performance. Hence,

for the rest of the discussion, we focus mainly on the sample sizes estimated from multi-modal baseline rDAm enrichment instead of rDrm.

MMSE, CDR-SB, and DxConv sample estimates (in [Tables 15.5–15.8](#)) outperform all other alternate outcomes considered here, even in the no-enrichment regime. This may be counter intuitive because of the simplicity of MMSE compared to other composite scores like PsyMEM and PsyEF (neuro-psych memory and executive function composites). It is possible that the composite nature of these measures increases the outcome variance, and thereby increases the sample estimates when used as trial outcomes. Since our population is entirely MCIs, it may be expected that the distribution of baseline rDAm is fairly uniform between 0 and 1, but is not the case as shown from rDAm enrichment cut-offs at each of the four percentiles considered (the top row of last four columns in [Table 15.5](#)). More precisely, the bottom 50% corresponds to a cut-off of 0.65, and 33% corresponds to 0.52, which indicates that more than two-thirds of MCIs in the ADNI2 cohort are healthier (i.e., weak decliners), and also that enrichment is important. This idea has also been identified by others using cognitive characteristics [58]. Ideally, we expect to observe a particular baseline rDAm cut-off (an elbow) at which there might be the highest decrease in estimates for all outcomes in [Tables 15.5 and 15.7](#). The elbow should be a natural threshold point that separates strong and weak decliners on baseline rDAm scale. However, the trends in the last four columns do not seem to suggest such a threshold, which is not surprising following [Fig. 15.3](#) and the corresponding discussion above. Specifically, ADAS and RAVLT seem to have an elbow between 25% and 33%, while for MMSE, CDR-SB, and DxConv, the elbow is beyond 50%.

Covariate information, or rather, a preliminary selection based on a factor like FH, is almost always helpful in estimating group effects ([Tables 15.7–15.8](#)). It has been observed that subjects with positive FH (either maternal or paternal) and/or APOE e4 positive may have stronger characteristics of dementia [59]. This implies that, instead of starting off with all MCIs, it is reasonable to include only those MCIs with positive FH and/or positive APOE e4, and then perform the baseline rDAm or rDrm enrichment on this smaller cohort. APOE had a higher dependence on both rDAm and rDrm compared to FH (from [Tables 15.2–15.3](#)), which resulted in a smaller sample sizes when using APOE or APOE + FH in tandem with rDAm enrichment (last two columns), than using rDAm + FH (sixth column) for all the cases except MMSE (row 1 in [Table 15.7](#)). Note that [Table 15.7](#) corresponds to bottom 20% baseline rDAm enrichment of which about half were FH and/or APOE positive. The strong performance of DxConv with small sample sizes may be because it summarizes the conversion of MCI to AD using longitudinal information, where as rDAm tries to predict this conversion using baseline information alone. We note that, since we have 267 MCIs to begin with, even with rDAm enrichment alone, a bottom 20% enrichment (third column, [Tables 15.5 and 15.6](#)) corresponds to a population size of 52, implying that the estimates might be noisy.

Overall, [Tables 15.5 and 15.7](#) support the efficacy of rDAm enrichment (similar trends are seen for rDrm enrichment from [Tables 15.6 and 15.8](#)); however, an interesting way to evaluate the strength of rDAm is by fixing the number of trial-enrolled

subjects and computing the detectable treatment size ( $\eta$ ). If in fact, baseline rDAm successfully selects strong decliners, then the trial should be able to detect smaller expected decrease in disease (i.e., smaller  $1 - \eta$  or larger  $\eta$ , refer to (15.1)). Fig. 15.5 shows exactly this behavior for rDAm (and Fig. 15.6 for rDrm), where  $\eta$  (y-axis) increases drastically as rDAm cut-offs (x-axis) are decreased (especially for MMSE, CDR-SB and DxConv). From the perspective of a practitioner, such plots are very useful. Specifically, they give tools for evaluating the minimum treatment effect that can be deemed significant (for the given outcome), from a fixed cut-off and sample size. Such feedback will be helpful to either change the outcomes or change the population size correspondingly.

We discussed in Sections 15.1 and 15.3 that although effective imaging-derived disease markers exist (either based on machine learning models or directly computed from imaging ROIs), they may not lead to the best possible clinical trials. This is supported by the results in Table 15.9, where both rDAm and rDrm (designed to explicitly reduce the prediction variance) are compared to existing markers that have been used as trial inclusion criteria [8,6,18]. For example, ROI summaries from multiple imaging modalities have often been used as trial enrichers [7,8], and rDAm and rDrm significantly outperform these baselines (first four rows in Table 15.9). Further, [6] used SVMs to design an effective disease marker and used it as an inclusion criterion in trials. Correspondingly, we compared rDAm and rDrm to MKLm (based on a multi-kernel SVM), and the results in Table 15.9 show that baseline rDAm and rDrm as enrichers outperform MKLm, and the improvements are higher for MOCA, RAVLT, and hippocampal volume as outcomes. For our experiments, we did not adjust any of the parameters relative to the results reports earlier [4], and they were the defaults for the MKL code-base provided on the webpage ([http://pages.cs.wisc.edu/~hinrichs/MKL\\_ADNI/](http://pages.cs.wisc.edu/~hinrichs/MKL_ADNI/)) by the authors.

The necessity of incorporating multi-modal information in designing any disease markers has been reported earlier [4,2]. This is further supported by the improvement of rDAm estimates over uni-modal measures including hippocampal volume, FDG ROI summaries and florbetapir ROI summaries. These results also build upon the work of [7,8] where such unimodal imaging summaries are used for enrichment. It is possible to demonstrate that the performance gains of rDAm over [7,8] is not merely due to using three distinct modalities but also heavily influenced by the underlying machine learning architecture that exploits this information meaningfully. To see this, compare our proposed markers to the enricher “FAH” which combines three uni-modal measures, FDG, florbetapir, and hippocampal volume in Table 15.9. FAH’s sample estimates are still larger than those obtained from both rDAm and rDrm, implying that the reductions are not merely due to multi-modal data or small population size, but due to the efficacy of the randomized deep learning methods introduced here, and their capacity of picking up strong decliners with high confidence and small variance.

Overall, our evaluations and the resulting trends clearly suggest that rDAm and rDrm enrichment (rDAm better among them) reduce sample sizes significantly leading to practical and cost-effective AD clinical trials. The rDA and rDr models scale

to very large dimensions, learn from only a small number of instances, and can be easily incorporated to design robust multi-modal imaging (or non-imaging, if the corresponding blocks are designed appropriately) markers. The full implementation of the framework is made available at <http://pages.cs.wisc.edu/~vamsi/rda>. The framework can, nevertheless, be improved further, particularly in terms of using richer pooling strategies instead of simple ridge regression, using covariate information like age, CSF levels (or FH, APOE, etc.) in the network construction itself (instead of the pre-selection setup used earlier in our experiments). An interesting extension would be to incorporate multi-modal and multi-domain (e.g., ordinal, continuous, and nominal) information directly into the rDA or rDr construction leading to multi-variate randomized deep network models. These technical issues are of independent interest and will be investigated in future works.

---

## ACKNOWLEDGEMENTS

NIH R01 AG040396; NSF CAREER award 1252725; NIH R01 AG021155; Wisconsin Partnership Program; UW ADRC P50 AG033514; UW ICTR 1ULIRR025011. We thank the anonymous reviewers for their valuable comments.

---

## REFERENCES

1. J. Weuve, L.E. Hebert, P.A. Scherr, D.A. Evans, Deaths in the United States among persons with Alzheimer's disease (2010–2050), *Alzheimer's Dement.* 10 (2) (2014) e40–e46.
2. D. Zhang, Y. Wang, L. Zhou, et al., Multimodal classification of Alzheimer's disease and mild cognitive impairment, *NeuroImage* 55 (3) (2011) 856–867.
3. S.J. Teipel, C. Born, M. Ewers, A.L. Bokde, M.F. Reiser, H.J. Möller, H. Hampel, Multivariate deformation-based analysis of brain atrophy to predict Alzheimer's disease in mild cognitive impairment, *NeuroImage* 38 (2007) 13–24.
4. C. Hinrichs, V. Singh, G. Xu, S.C. Johnson, Predictive markers for AD in a multi-modality framework: an analysis of MCI progression in the ADNI population, *NeuroImage* 55 (2011) 574–589.
5. C. Hinrichs, N. Dowling, S. Johnson, V. Singh, MKL-based sample enrichment and customized outcomes enable smaller ad clinical trials, in: MLINI, in: *Lect. Notes Comput. Sci.*, vol. 7263, Springer, Berlin, Heidelberg, ISBN 978-3-642-34712-2, 2012, pp. 124–131.
6. O. Kohannim, X. Hua, D.P. Hibar, S. Lee, Y.Y. Chou, A.W. Toga, C.R. Jack Jr., M.W. Weiner, P.M. Thompson, Boosting power for clinical trials using classifiers based on multiple biomarkers, *Neurobiol. Aging* 31 (2010) 1429–1442.
7. J.D. Grill, L. Di, P.H. Lu, C. Lee, J. Ringman, L.G. Apostolova, et al., Estimating sample sizes for predementia Alzheimer's trials based on the Alzheimer's Disease Neuroimaging Initiative, *Neurobiol. Aging* 34 (2013) 62–72.
8. J.D. Grill, S.E. Monsell, Choosing Alzheimer's disease prevention clinical trial populations, *Neurobiol. Aging* 35 (3) (2014) 466–471.

9. V. Jelic, M. Kivipelto, B. Winblad, Clinical trials in mild cognitive impairment: lessons for the future, *J. Neurol. Neurosurg. Psychiatry* 77 (4) (2006) 429–438.
10. R.C. Petersen, Mild cognitive impairment: current research and clinical implications, in: *Seminars in Neurology*, vol. 27, 2007, pp. 22–31.
11. P.S. Aisen, Clinical trial methodologies for disease-modifying therapeutic approaches, *Neurobiol. Aging* 32 (2011) S64–S66.
12. M.S. Albert, S.T. DeKosky, D. Dickson, B. Dubois, H.H. Feldman, N.C. Fox, A. Gamst, D.M. Holtzman, W.J. Jagust, R.C. Petersen, et al., The diagnosis of mild cognitive impairment due to Alzheimer's disease: recommendations from the National Institute on Aging–Alzheimer's Association workgroups on diagnostic guidelines for Alzheimer's disease, *Alzheimer's Dement.* 7 (3) (2011) 270–279.
13. A.J. Mitchell, M. Shiri-Feshki, Rate of progression of mild cognitive impairment to dementia – meta-analysis of 41 robust inception cohort studies, *Acta Psychiatr. Scand.* 119 (4) (2009) 252–265.
14. M. Lorenzi, M. Donohue, D. Paternico, C. Scarpazza, S. Ostrowitzki, O. Blin, E. Irving, G. Frisoni, A.D.N. Initiative, et al., Enrichment through biomarkers in clinical trials of Alzheimer's drugs in patients with mild cognitive impairment, *Neurobiol. Aging* 31 (8) (2010) 1443–1451.
15. J.M.S. Leoutsakos, A.L. Bartlett, S.N. Forrester, C.G. Lyketsos, Simulating effects of biomarker enrichment on Alzheimer's disease prevention trials: conceptual framework and example, *Alzheimer's Dement.* 10 (2) (2014) 152–161.
16. N. Mattsson, U. Andreasson, S. Persson, M.C. Carrillo, S. Collins, S. Chalbot, N. Cutler, D. Dufour-Rainfray, A.M. Fagan, N.H. Heegaard, et al., CSF biomarker variability in the Alzheimer's Association quality control program, *Alzheimer's Dement.* 9 (3) (2013) 251–261.
17. J. Escudero, J.P. Zajicek, E. Ifeachor, Machine learning classification of MRI features of Alzheimer's disease and mild cognitive impairment subjects to reduce the sample size in clinical trials, in: *Engineering in Medicine and Biology Society, EMBC, 2011 Annual International Conference of the IEEE*, IEEE, 2011, pp. 7957–7960.
18. P. Yu, J. Sun, R. Wolz, D. Stephenson, J. Brewer, N.C. Fox, P.E. Cole, C.R. Jack, D.L. Hill, A.J. Schwarz, et al., Operationalizing hippocampal volume as an enrichment biomarker for amnestic mild cognitive impairment trials: effect of algorithm, test-retest variability, and cut point on trial cost, duration, and sample size, *Neurobiol. Aging* 35 (4) (2014) 808–818.
19. Y. Bengio, I.J. Goodfellow, A. Courville, Deep Learning, a MIT book in preparation. Draft chapters available at <http://www.deeplearningbook.org/>, 2015.
20. Y. Bengio, Learning deep architectures for AI, *Found. Trends Mach. Learn.* 2 (2009) 1–127.
21. Y. Bengio, A. Courville, P. Vincent, Representation learning: a review and new perspectives, *IEEE Trans. Pattern Anal. Mach. Intell.* 35 (8) (2013) 1798–1828.
22. D. Erhan, Y. Bengio, A. Courville, P.A. Manzagol, P. Vincent, S. Bengio, Why does unsupervised pre-training help deep learning?, *J. Mach. Learn. Res.* 11 (2010) 625–660.
23. K. Kavukcuoglu, P. Sermanet, Y. Boureau, K. Gregor, M. Mathieu, Y. LeCun, Learning convolutional feature hierarchies for visual recognition, in: *Advances in Neural Information Processing Systems*, vol. 1, 2010, p. 5.
24. A. Krizhevsky, I. Sutskever, G. Hinton, ImageNet classification with deep convolutional neural networks, in: *Advances in Neural Information Processing Systems*, vol. 1, 2012, p. 4.

25. G. Dahl, D. Yu, L. Deng, A. Acero, Large vocabulary continuous speech recognition with context-dependent DBN-HMMs, in: Proceedings of Acoustics, Speech and Signal Processing (ICASSP), 2011, pp. 4688–4691.
26. F. Bach, Breaking the curse of dimensionality with convex neural networks, arXiv: 1412.8690, 2014.
27. V.K. Ithapu, S. Ravi, V. Singh, On the interplay of network structure and gradient convergence in deep learning, arXiv:1511.05297, 2015.
28. H.I. Suk, D. Shen, Deep learning-based feature representation for AD/MCI classification, in: K. Mori, I. Sakuma, Y. Sato, C. Barillot, N. Navab (Eds.), MICCAI, in: Lect. Notes Comput. Sci., vol. 8150, Springer, Berlin, Heidelberg, ISBN 978-3-642-40762-8, 2013, pp. 583–590.
29. A. Gupta, M. Ayhan, A. Maida, Natural image bases to represent neuroimaging data, in: Proceedings of the 30th ICML, 2013, pp. 987–994.
30. S.M. Plis, D.R. Hjelm, R. Salakhutdinov, V.D. Calhoun, Deep learning for neuroimaging: a validation study, arXiv:1312.5847, 2013.
31. L.M. Friedman, C.D. Furberg, D.L. DeMets, Sample size, in: Fundamentals of Clinical Trials, Springer, 2010, pp. 133–167.
32. P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, P. Manzagol, Stacked denoising autoencoders: learning useful representations in a deep network with a local denoising criterion, *J. Mach. Learn. Res.* 11 (2010) 3371–3408.
33. N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: a simple way to prevent neural networks from overfitting, *J. Mach. Learn. Res.* 15 (1) (2014) 1929–1958.
34. T.V. Sakpal, Sample size estimation in clinical trial, *Perspect. Clin. Res.* 1 (2) (2010) 67–69.
35. M. Minsky, S. Papert, Perceptrons, 1969.
36. G.E. Dahl, T.N. Sainath, G.E. Hinton, Improving deep neural networks for LVCSR using rectified linear units and dropout, in: 2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2013, pp. 8609–8613.
37. L. Bottou, Large-scale machine learning with stochastic gradient descent, in: Proceedings of COMPSTAT'2010, Springer, 2010, pp. 177–186.
38. Y.A. LeCun, L. Bottou, G.B. Orr, K.R. Müller, Efficient BackProp, in: Neural Networks: Tricks of the Trade, Springer, 2012, pp. 9–48.
39. C.M. Bishop, Pattern Recognition and Machine Learning, Springer, 2006.
40. N. Le Roux, Y. Bengio, Representational power of restricted Boltzmann machines and deep belief networks, *Neural Comput.* 20 (6) (2008) 1631–1649.
41. P. Baldi, P. Sadowski, The dropout learning algorithm, *Artif. Intell.* 210 (2014) 78–122.
42. S. Boyd, L. Vandenberghe, Convex Optimization, Cambridge University Press, 2004.
43. S. Kay, Fundamentals of Statistical Signal Processing: Estimation Theory, Prentice Hall, 1993.
44. S.T. Smith, Statistical resolution limits and the complexified Cramér–Rao bound, *IEEE Trans. Signal Process.* 53 (5) (2005) 1597–1609.
45. T.G. Dietterich, Ensemble methods in machine learning, in: Multiple Classifier Systems, 2000, pp. 1–15.
46. D. Erhan, P. Manzagol, Y. Bengio, S. Bengio, P. Vincent, The difficulty of training deep architectures and the effect of unsupervised pre-training, in: Proceedings of the International Conference on Artificial Intelligence and Statistics, 2009, pp. 153–160.

47. J. Ngiam, A. Coates, A. Lahiri, B. Prochnow, Q.V. Le, A.Y. Ng, On optimization methods for deep learning, in: Proceedings of the 28th International Conference on Machine Learning (ICML-11), 2011, pp. 265–272.
48. C. Hinrichs, V. Singh, G. Xu, S.C. Johnson, A.D.N. Initiative, et al., Predictive markers for AD in a multi-modality framework: an analysis of MCI progression in the ADNI population, *NeuroImage* 55 (2) (2011) 574–589.
49. R. Bourgon, R. Gentleman, W. Huber, Independent filtering increases detection power for high-throughput experiments, *Proc. Natl. Acad. Sci. USA* 107 (21) (2010) 9546–9551.
50. C.R. Jack, D.S. Knopman, W.J. Jagust, R.C. Petersen, M.W. Weiner, P.S. Aisen, L.M. Shaw, P. Vemuri, H.J. Wiste, S.D. Weigand, et al., Tracking pathophysiological processes in Alzheimer's disease: an updated hypothetical model of dynamic biomarkers, *Lancet Neurol.* 12 (2) (2013) 207–216.
51. M.W. Weiner, D.P. Veitch, P.S. Aisen, L.A. Beckett, N.J. Cairns, R.C. Green, D. Harvey, C.R. Jack, W. Jagust, E. Liu, et al., The Alzheimer's disease neuroimaging initiative: a review of papers published since its inception, *Alzheimer's Dement.* 9 (5) (2013) e111–e194.
52. J. Ashburner, VBM Tutorial, 2010.
53. J. Ashburner, G. Barnes, C. Chen, J. Daunizeau, G. Flandin, K. Friston, et al., SPM8 Manual, 2008.
54. M.W. Weiner, D.P. Veitch, P.S. Aisen, L.A. Beckett, N.J. Cairns, J. Cedarbaum, R.C. Green, D. Harvey, C.R. Jack, W. Jagust, et al., 2015 update of the Alzheimer's disease neuroimaging initiative: a review of papers published since its inception, *Alzheimer's Dement.* 11 (6) (2015) e1–e120.
55. C. Hinrichs, V. Singh, J. Peng, S. Johnson, Q-MKL: matrix-induced regularization in multi-kernel learning with applications to neuroimaging, in: Advances in Neural Information Processing Systems, 2012, pp. 1421–1429.
56. S. Klöppel, C.M. Stonnington, C. Chu, B. Draganski, R.I. Scahill, J.D. Rohrer, N.C. Fox, C.R. Jack, J. Ashburner, R.S. Frackowiak, Automatic classification of MR scans in Alzheimer's disease, *Brain* 131 (3) (2008) 681–689.
57. C. Davatzikos, P. Bhatt, L.M. Shaw, K.N. Batmanghelich, J.Q. Trojanowski, Prediction of MCI to AD conversion, via MRI, CSF biomarkers, and pattern classification, *Neurobiol. Aging* 32 (2011) 2322.e19.
58. E.C. Edmonds, L. Delano-Wood, L.R. Clark, A.J. Jak, D.A. Nation, C.R. McDonald, D.J. Libon, R. Au, D. Galasko, D.P. Salmon, et al., Susceptibility of the conventional criteria for mild cognitive impairment to false-positive diagnostic errors, *Alzheimer's Dement.* 11 (4) (2015) 415–424.
59. W. Huang, C. Qiu, E. von Strauss, B. Winblad, L. Fratiglioni, APOE genotype, family history of dementia, and Alzheimer disease risk: a 6-year follow-up study, *Arch. Neurol.* 61 (12) (2004) 1930–1934.

---

## NOTES

1. The ADNI was launched in 2003 by the National Institute on Aging (NIA), the National Institute of Biomedical Imaging and Bioengineering (NIBIB), the Food and Drug Administration (FDA), private pharmaceutical companies and non-profit organizations, as a 60 million, 5-year public-private partnership. The primary goal of ADNI has been to test whether MRI, PET, other biological markers, including clinical and neuropsychological assessment can be combined to measure the progression of MCI and early AD.

2. <http://adni.loni.usc.edu/methods/documents/mri-protocols/>, <http://adni.loni.usc.edu/methods/pet-analysis/pet-acquisition/>.
3. <http://adni.loni.usc.edu/methods/documents/>, [http://www.alz.org/research/science/earlier\\_alzheimers\\_diagnosis.asp](http://www.alz.org/research/science/earlier_alzheimers_diagnosis.asp).
4. FDG ROIs include Left Angular Lobe, Right Angular Lobe, Left Temporal Lobe, Right Temporal Lobe, and Cingulate. AV45 ROIs include Frontal Lobe, Temporal Lobe, Parietal Lobe, and Cingulate gray matter. The corresponding ROI measures are summed up to obtain single global summary for each of FDG and AV45.

PART

Others

6

This page intentionally left blank

# Deep Networks and Mutual Information Maximization for Cross-Modal Medical Image Synthesis

# 16

Raviteja Vemulapalli\*, Hien Van Nguyen†, S. Kevin Zhou‡

*University of Maryland, College Park, MD, United States\** *Uber Advanced Technology Center, Pittsburgh, PA, United States*† *Siemens Healthineers Technology Center, Princeton, NJ, United States*‡

## CHAPTER OUTLINE

16.1	Introduction .....	382
16.2	Supervised Synthesis Using Location-Sensitive Deep Network .....	384
16.2.1	Backpropagation .....	386
16.2.2	Network Simplification .....	387
16.2.3	Experiments .....	388
16.3	Unsupervised Synthesis Using Mutual Information Maximization .....	390
16.3.1	Generating Multiple Target Modality Candidates .....	392
16.3.2	Full Image Synthesis Using Best Candidates .....	393
16.3.2.1	Global Mutual Information Maximization .....	393
16.3.2.2	Local Spatial Consistency Maximization .....	394
16.3.2.3	Combined Formulation .....	394
16.3.2.4	Optimization .....	395
16.3.3	Refinement Using Coupled Sparse Representation .....	396
16.3.4	Extension to Supervised Setting .....	396
16.3.5	Experiments .....	397
16.4	Conclusions and Future Work .....	401
	Acknowledgements .....	401
	References .....	401
	Note .....	403

## CHAPTER POINTS

---

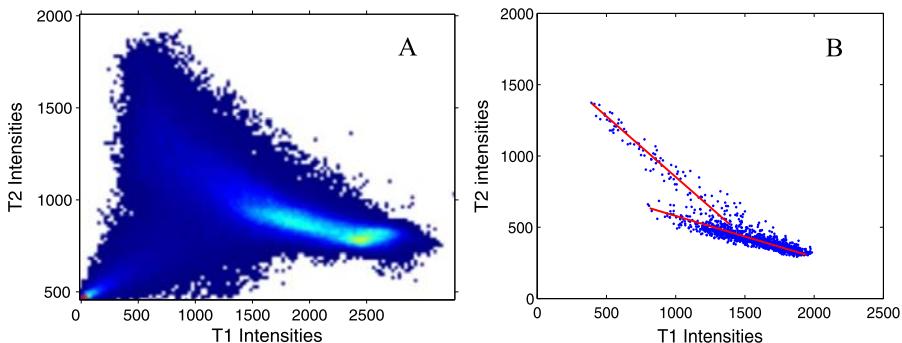
- Location-sensitive deep network integrates spatial information with intensity-based features by modeling the first hidden layer nodes as products of certain location-sensitive functions and nonlinear features computed from voxel intensity values
  - The mutual information maximization-based approach can be used for synthesis in the unsupervised setting by combining it with cross-modal nearest neighbor search
  - The presented approaches show promising synthesis capabilities when evaluated by synthesizing MR contrast
- 

### 16.1 INTRODUCTION

Currently, a multitude of imaging modalities such as Computed Tomography, X-ray, Magnetic Resonance Imaging (MRI), Ultrasound, etc., are being used for medical imaging research and clinical diagnosis. Each of these modalities has different contrast and noise mechanisms and hence captures different characteristics of the underlying anatomy. The intensity transformation between any two modalities is highly nonlinear.

Due to variations in the image characteristics across modalities, medical image analysis algorithms trained with data from one modality may not work well when applied to data from a different modality. A straightforward way to address this issue is to collect a large amount of training data in each modality. But, this is impractical since collecting medical images is time consuming, expensive, and involves exposing patients to radiation. Hence, it is highly desirable to have a general cross-modal synthesis approach that generates subject-specific scans in the desired target modality from the given source modality images. The ability to synthesize medical images without real acquisitions has various potential applications such as super-resolution [10,21], atlas construction [5], multimodal registration [4,18,19], segmentation [9,20], and virtual enhancement [15].

Due to its potential applications, cross-modal synthesis has received significant attention from the medical imaging community in the recent past, and various approaches have been proposed for this task. In [12], a set of paired low/high resolution images was used to learn the joint probability distribution of low/high resolution patches. This probability distribution was then used for super-resolution. A regression-based synthesis approach was proposed in [11] using random forest. In [4], a cross-modal image synthesis approach was proposed based on coupled sparse representation. While training, this approach used paired data from source and target modalities to learn coupled dictionaries that establish cross-modal correspondences. These learned dictionaries were then used for sparse coding-based image synthesis. Similar sparse coding-based approaches have also been used in [3, 19,21,24] for image synthesis applications. In [20], a codebook of paired training patches was used for MR contrast synthesis. For each test patch, few best matches



**FIGURE 16.1**

(A) Intensity correspondences between T1-MRI and T2-MRI over an entire image. Brighter color indicates higher density region. (B) Intensity correspondences over a region of  $10 \times 10 \times 10$  voxels. Red lines indicate the main directions of variation. Images are registered using rigid transformations.

(among the source modality patches) were found from the codebook and the target patches corresponding to the best matches were averaged to generate target MR contrast. A similar approach based on image analogies [7] was also used in [9]. A label propagation [17] based iterative synthesis approach, called modality propagation, was proposed in [27]. Alternative to cross-modal medical image synthesis approaches, compressed sensing-based methods such as magnetic resonance fingerprinting [13] have also been proposed in the recent past to reduce scanning time and cost.

Most of the existing synthesis approaches either do not use the spatial information or use it in the form of a hard constraint. For example, modality propagation [27] restricts the nearest neighbor search to a small window around the voxel's location to incorporate the spatial information. To see the importance of spatial information, we plot the T1-T2 MRI intensity correspondences in Fig. 16.1A using the (registered) brain scans of a subject from the NAMIC multimodality database. We can notice that the intensity transformation is not only nonlinear but also far from unique, i.e., there are several feasible intensity values in T2-MRI modality for one intensity value in T1-MRI modality, and vice versa. This non-uniqueness comes from the fact that the intensity transformation depends on the region in which voxels reside. If we restrict the region of interest to a local neighborhood, say of size  $10 \times 10 \times 10$  voxels, the intensity transformation becomes much simpler as shown in Fig. 16.1B. This shows that spatial information helps in simplifying the transformations between modalities, which in turn enables more accurate predictions. Unfortunately, most of the existing approaches do not have a systematic way to incorporate spatial information.

Section 16.2 of this chapter presents a novel deep network architecture, referred to as location-sensitive deep network (LSDN), that integrates image intensity-based

features and spatial information in a principled manner. This network, which was initially proposed in [14], models the first hidden layer nodes as products of feature responses computed from voxel intensity values and certain location-sensitive functions. As a result, LSDN is able to capture the joint distribution of intensity features and spatial information. In addition, Section 16.2 also presents a network simplification method for reducing the LSDN synthesis time while maintaining the prediction accuracy.

All the above mentioned synthesis approaches work under the supervised setting, i.e., they require training data in both source and target modalities from the same set of subjects. The availability of such paired data is limited in many cases. Also, collecting multiple scans from each subject is not desirable especially when patients are exposed to radiation. Hence, there is a need for a cross-modal synthesis approach that works in the unsupervised setting, i.e., when paired training data is not available.

Section 16.3 of this chapter presents an approach based on cross-modal nearest neighbor search and mutual information maximization, which can be used in the unsupervised setting. Given a source modality image, this approach first generates multiple target modality candidate values independently for each voxel by performing cross-modal nearest neighbor search over the training database. Then, the best candidate values are selected jointly for all the voxels by simultaneously maximizing a global mutual information-based cost function and a local spatial consistency cost function. To the best of our knowledge, this approach, which was initially proposed in [23], is the first approach that addresses the cross-modal image synthesis problem in an unsupervised setting. This approach can also be used in the supervised setting by replacing the cross-modal nearest neighbor search with source-modal nearest neighbor search.

**Notations.** Matrices are denoted using boldface capital letters and vectors are denoted using boldface small letters. The  $\ell_0$  and  $\ell_2$  norms of a vector  $v$  are denoted by  $\|v\|_0$  and  $\|v\|_2$ , respectively. We use  $\odot$  to denote the Hadamard product. The transpose and Frobenius norm of a matrix  $A$  are denoted by  $A^\top$  and  $\|A\|_F$ , respectively. The  $i$ th column of a matrix  $A$  is denoted using  $A(:, i)$ . We use  $\mathbb{I}$  to denote the indicator function and  $P$  to denote probability.

## 16.2 SUPERVISED SYNTHESIS USING LOCATION-SENSITIVE DEEP NETWORK

Cross-modal image synthesis can be seen as a regression problem with the image in source modality as input and the image in target modality as output. In this section, we will use a neural network as the regressor. It is ineffective to train a network that operates on the entire image since the number of parameters would be too large for the network to generalize well. Hence, we will use a network that takes a source modality image patch as input and predicts the target modality intensity

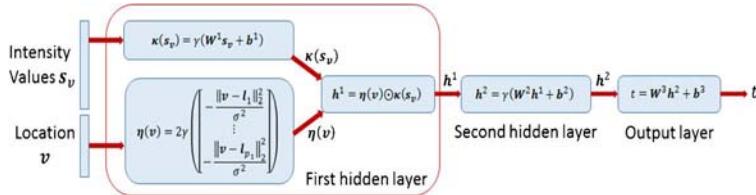


FIGURE 16.2

Location-sensitive deep network with two hidden layers.

value for the center voxel of the patch. Note that the synthesized target modality image will be produced in the same coordinate system as the given source modality image.

Let  $s_v$  represent the vectorized version of a  $d \times d \times d$  patch centered on voxel  $\mathbf{v} = (x, y, z)$  in the source modality input image. As discussed earlier (Fig. 16.1), the intensity transformation between modalities depends on voxel's spatial location.<sup>1</sup> Hence, the neural network regressor should take the voxel location  $\mathbf{v}$  as an input along with the intensity features  $s_v$ . One straightforward way to do this is to concatenate  $s_v$  and  $\mathbf{v}$  into a single vector and train a standard multilayer perceptron (MLP) with the concatenated vector as input. However, this strategy does not work well (as shown later in Section 16.2.3) since the intensity features and spatial location are two heterogeneous sources of information.

In the section, we present a new network architecture, referred to as location-sensitive deep network, which combines the intensity input  $s_v$  and the spatial location  $\mathbf{v}$  in a principled manner. This network is similar to an MLP with sigmoid nonlinearity except that the first hidden layer nodes are modeled as products of feature responses computed from the intensity input  $s_v$  and certain spatial response functions. Fig. 16.2 shows an LSDN with two hidden layers. The  $i$ th node in the first hidden layer performs the following computations:

$$h_i^1 = \eta_i(\mathbf{v})\kappa_i(s_v), \quad \eta_i(\mathbf{v}) = 2\gamma\left(-\frac{\|\mathbf{v} - \mathbf{l}_i\|_2^2}{\sigma^2}\right), \quad (16.1)$$

$$\kappa_i(s_v) = \gamma\left(\mathbf{w}_i^{1\top} s_v + b_i^1\right).$$

Here,  $\gamma$  denotes the standard sigmoid function,  $\eta_i(\mathbf{v})$  is a spatial response function parametrized by a location  $\mathbf{l}_i$  which will be learned during training and a constant  $\sigma$  (we could also use a different  $\sigma$  for each node and learn them during training), and  $\kappa_i(s_v)$  is an intensity-based feature response function parametrized by a linear filter  $\mathbf{w}_i^1$  and bias  $b_i^1$  which will be learned during training. The computations performed

by the entire first hidden layer with  $p_1$  nodes can be written as

$$\begin{aligned} \mathbf{h}^1 &= \begin{bmatrix} h_1^1 \\ \vdots \\ h_{p_1}^1 \end{bmatrix} = \eta(\mathbf{v}) \odot \kappa(s_{\mathbf{v}}), \quad \eta(\mathbf{v}) = \begin{bmatrix} \eta_1(\mathbf{v}) \\ \vdots \\ \eta_{p_1}(\mathbf{v}) \end{bmatrix} = 2\gamma \begin{pmatrix} -\frac{\|\mathbf{v}-\mathbf{l}_1\|_2^2}{\sigma^2} \\ \vdots \\ -\frac{\|\mathbf{v}-\mathbf{l}_{p_1}\|_2^2}{\sigma^2} \end{pmatrix}, \\ \kappa(s_{\mathbf{v}}) &= \begin{bmatrix} \kappa_1(s_{\mathbf{v}}) \\ \vdots \\ \kappa_{p_1}(s_{\mathbf{v}}) \end{bmatrix} = \gamma (\mathbf{W}^1 s_{\mathbf{v}} + \mathbf{b}^1), \quad \mathbf{W}^1 = \begin{bmatrix} \mathbf{w}_1^{1\top} \\ \vdots \\ \mathbf{w}_{p_1}^{1\top} \end{bmatrix}, \quad \mathbf{b}^1 = \begin{bmatrix} b_1^1 \\ \vdots \\ b_{p_1}^1 \end{bmatrix}. \end{aligned} \quad (16.2)$$

All the other hidden layers of an LSDN are similar to the hidden layers of a standard MLP, i.e., they perform the following computations:

$$\mathbf{h}^k = \gamma (\mathbf{W}^k \mathbf{h}^{k-1} + \mathbf{b}^k). \quad (16.3)$$

The output layer of an LSDN with  $K$  layers (i.e.,  $K - 1$  hidden layers) computes the target modality intensity value of the center voxel of the input patch using

$$t = \mathbf{W}^K \mathbf{h}^{K-1} + \mathbf{b}^K. \quad (16.4)$$

Note that the  $i$ th node of first hidden layer will be active only when the voxel location  $\mathbf{v}$  is close enough to the location  $\mathbf{l}_i$ . Hence, based on the voxel location  $\mathbf{v}$  some of the first hidden layer nodes will be active and the others will not be. Different combinations of on/off nodes effectively create multiple sub-networks, each of which is tuned to a small spatial region in the image. This novel property is the main advantage of an LSDN compared to a standard MLP. Recall the observation from Fig. 16.1 that the input–output mapping becomes simpler when restricted to a smaller spatial region. Therefore, LSDN has the potential to yield more accurate predictions.

To ensure that the spatial location  $\mathbf{v}$  conveys meaningful information, training and test images are registered to a reference image using rigid transformations. After the registration, the same voxel location in different images corresponds to roughly the same anatomical region. Alternatively, one could eliminate the need for registration by using relative coordinates with respect to some landmarks.

### 16.2.1 BACKPROPAGATION

Similar to an MLP, an LSDN can also be trained using stochastic gradient descent by computing the gradients using standard backpropagation. In this section, we show how to backpropagate the error derivatives through the first hidden layer of an LSDN. We skip the derivative formulas corresponding to the subsequent layers since they are standard MLP layers.

The derivatives  $dE/d\mathbf{h}^1$  of the error function  $E$  with respect to the first hidden layer outputs  $\mathbf{h}^1$  can be computed by backpropagating the error derivatives from the output layer till the second hidden layer. Given these derivatives, we can compute the derivatives of  $E$  with respect to  $\eta(\mathbf{v})$  and  $\kappa(s_v)$  using

$$\frac{dE}{d\eta(\mathbf{v})} = \frac{dE}{d\mathbf{h}^1} \odot \kappa(s_v), \quad \frac{dE}{d\kappa(s_v)} = \frac{dE}{d\mathbf{h}^1} \odot \eta(\mathbf{v}). \quad (16.5)$$

Given the derivatives  $dE/d\eta(\mathbf{v})$ , we can compute the derivatives of  $E$  with respect to the network location parameters  $\mathbf{l}_i$  using

$$\frac{dE}{d\mathbf{l}_i} = \frac{4}{\sigma^2} \gamma \left( -\frac{\|\mathbf{v} - \mathbf{l}_i\|_2^2}{\sigma^2} \right) \left( 1 - \gamma \left( -\frac{\|\mathbf{v} - \mathbf{l}_i\|_2^2}{\sigma^2} \right) \right) \frac{dE}{d\eta_i(\mathbf{v})} (\mathbf{v} - \mathbf{l}_i). \quad (16.6)$$

Given the derivatives  $dE/d\kappa(s_v)$ , the derivatives  $dE/d\mathbf{W}^1$  and  $dE/d\mathbf{b}^1$  can be computed by backpropagating through the computation of  $\kappa(s_v)$  which is nothing but a standard MLP layer.

### 16.2.2 NETWORK SIMPLIFICATION

Applying LSDN on every  $d \times d \times d$  patch during the synthesis process could be computationally expensive since medical images usually contain hundreds of thousands of voxels. In this section, we present a post-processing method for simplifying the network in order to improve the speed of LSDN without losing much in terms of prediction accuracy. At each hidden layer of the network, this method tries to find a small subset of features/nodes that can be used to reconstruct the entire layer approximately.

Let  $\mathcal{I}^k$  denote the index set of such a subset at the  $k$ th hidden layer. Then, we have

$$h_{i,n}^k \approx \sum_{j \in \mathcal{I}^k} \alpha_{ij}^k h_{j,n}^k, \quad i \in \{1, 2, \dots, p_k\}, \quad n \in \{1, 2, \dots, N\}, \quad (16.7)$$

where  $p_k$  is the number of nodes in the  $k$ th hidden layer,  $N$  is the number of training samples,  $h_{i,n}^k$  is the response of the  $i$ th node in the  $k$ th hidden layer for the  $n$ th training sample, and  $\alpha_{ij}^k$  are the reconstruction coefficients. The index set  $\mathcal{I}^k$  and coefficients  $\alpha_{ij}^k$  can be obtained by solving the following optimization problem:

$$\underset{\mathbf{A}^k}{\text{minimize}} \quad \|\mathbf{H}^k - \mathbf{A}^k \mathbf{H}^k\|_F^2, \quad \text{subject to} \quad \|\mathbf{A}^k\|_{col-0} \leq T^k, \quad (16.8)$$

$$\mathbf{A}_{ij}^k = \begin{cases} \alpha_{ij}^k, & \text{if } j \in \mathcal{I}^k, \\ 0, & \text{otherwise,} \end{cases} \quad \mathbf{H}^k = \begin{pmatrix} h_{1,1}^k & \dots & h_{1,N}^k \\ \vdots & \ddots & \vdots \\ h_{p_k,1}^k & \dots & h_{p_k,N}^k \end{pmatrix}. \quad (16.9)$$

By constraining the quasi-norm  $\|\mathbf{A}^k\|_{col-0}$ , which is the number of nonzero columns, to be less than a certain threshold  $T^k$ , the above optimization problem identifies a small subset of nodes that can approximately reconstruct the entire  $k$ th layer. Since the formulation in (16.8) is a special case of simultaneous sparsity, simultaneous orthogonal matching pursuit [22] can be used to efficiently minimize the cost function. The index set  $\mathcal{I}^k$  can be obtained from the indices of nonzero columns in  $\mathbf{A}^k$ .

Once we have the index sets  $\mathcal{I}^k$  and the corresponding reconstruction coefficients, we can simplify the LSDN by shrinking the number of connections in each layer, also referred to as *ShrinkConnect* in the rest of this chapter. This can be done by removing the hidden layer nodes whose indices are not in the index sets  $\mathcal{I}^k$ . Let  $\mathbf{h}_{\mathcal{I}^k}^k$  represent the  $k$ th hidden layer in the simplified network. Note that the output of the  $k$ th hidden layer is the input to the  $(k+1)$ th hidden layer. Hence, when we remove nodes from the  $k$ th hidden layer, we need to update  $\mathbf{W}^{k+1}$  such that the outputs of applying the new filters on  $\mathbf{h}_{\mathcal{I}^k}^k$  are approximately equal to the original LSDN outputs  $\mathbf{W}_{row \in \mathcal{I}^{k+1}}^{k+1} \mathbf{h}^k$ . Using the approximation in (16.8), the entire ShrinkConnect operation can be described by the following update rule:

$$\mathbf{W}^{k+1} \leftarrow \mathbf{W}_{row \in \mathcal{I}^{k+1}}^{k+1} \mathbf{A}_{column \in \mathcal{I}^k}^k, \quad \mathbf{b}^{k+1} \leftarrow \mathbf{b}_{ind \in \mathcal{I}^{k+1}}^{k+1}, \quad (16.10)$$

where  $\mathbf{W}_{row \in \mathcal{I}^{k+1}}^{k+1}$  is the matrix formed by the rows of  $\mathbf{W}^{k+1}$  whose indices are in  $\mathcal{I}^{k+1}$ ,  $\mathbf{A}_{column \in \mathcal{I}^k}^k$  is the matrix formed by the columns of  $\mathbf{A}^k$  whose indices are in  $\mathcal{I}^k$ , and  $\mathbf{b}_{ind \in \mathcal{I}^{k+1}}^{k+1}$  is a vector formed by the entries of  $\mathbf{b}^{k+1}$  whose indices are in  $\mathcal{I}^{k+1}$ .

### 16.2.3 EXPERIMENTS

In this section, we evaluate the LSDN architecture by generating T1-MRI scans from T2-MRI scans, and vice versa.

**Dataset and Pre-Processing.** We used the T1 and T2 MRI scans of 19 subjects from the NAMIC brain multimodality database (<http://hdl.handle.net/1926/1687>). Along with the MRI scans, this database also provides brain masks for skull-stripping. Following [27], all the images were skull stripped, linearly registered, inhomogeneity corrected, histogram matched within each modality, and resampled to 2 mm resolution. For registration, we used the first subject as reference. First, the T2 scan of the reference subject was registered to the corresponding T1 scan, and then the T1 and T2 scans of the remaining subjects were registered to the reference subject. We used 3D Slicer ([www.slicer.org](http://www.slicer.org)) software for data pre-processing.

**Evaluation Setting and Metric.** We used leave-one-out cross-validation setting, in which 18 image pairs were used for training and the remaining one was used for testing. Since the dataset has both T1 and T2 MRI scans for each subject, we directly compare the synthesized and ground truth target modality scans for evaluation. We use signal-to-noise ratio (SNR) as evaluation metric.

**Parameters.** The LSDN input patch size  $d$  and the spatial response function parameter  $\sigma$  were chosen as 3 and 0.5, respectively. When  $d = 3$ , the input to LSDN is

**Table 16.1** SNR values and computation times for various different approaches

Approach	SNR (T1→T2) (dB)	SNR (T2→T1) (dB)	Training (h)	Synthesis (s)
MP [27]	13.64 ± 0.67	15.13 ± 0.88	n/a	928
CSR [4]	13.72 ± 0.64	15.24 ± 0.85	2.8	245
VDN	12.67 ± 0.6	14.19 ± 0.82	1.2	23.5
CDN	13.79 ± 0.68	15.36 ± 0.88	1.2	23.6
LSDN-small	12.53 ± 0.75	13.85 ± 0.86	0.6	9.2
<b>LSDN-1</b>	<b>14.82 ± 0.72</b>	<b>17.09 ± 0.94</b>	<b>1.4</b>	<b>29.5</b>
<b>LSDN-2</b>	<b>14.93 ± 0.73</b>	<b>17.39 ± 0.91</b>	<b>2.5</b>	<b>68.0</b>
<b>LSDN-1+ShrinkConnect</b>	<b>14.79 ± 0.72</b>	<b>17.05 ± 0.91</b>	<b>1.4</b>	<b>9.2</b>
<b>LSDN-2+ShrinkConnect</b>	<b>14.80 ± 0.74</b>	<b>17.1 ± 0.86</b>	<b>2.5</b>	<b>21.5</b>

30-dimensional (27 intensity values and 3 spatial coordinates). We investigated three network configurations denoted LSDN-small, LSDN-1, and LSDN-2, whose sizes are [30-50-5-1], [30-200-20-1], and [30-400-40-1], respectively. In ShrinkConnect, the threshold  $T^k$  for each hidden layer was set to one-fourth of the layer's original size.

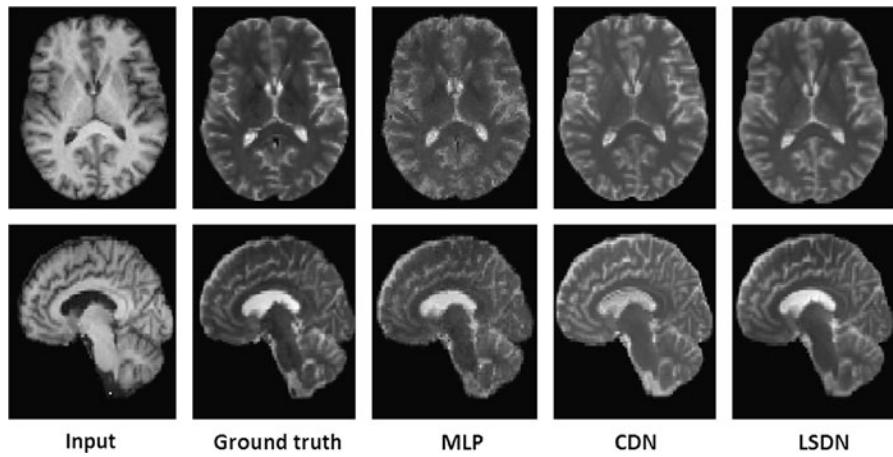
**Training.** For training, we randomly sampled around one million patches from the training images. We trained the networks using stochastic gradient descent for 300 epochs with mean squared error as loss function. We used an initial learning rate of 0.25 and slowly decreased the learning rate by multiplying it with 0.99 after each iteration. After ShrinkConnect, we fine-tuned the simplified networks for 10 epochs.

**Testing.** To synthesize a target modality image, LSDN was applied to the source modality image in a sliding window fashion.

**Comparisons.** We compare LSDN with the following approaches:

- Vanilla deep network (VDN). Standard MLP of size [27-400-40-1] that takes only the intensity values as input.
- Concatenation deep network (CDN). Standard MLP of size [30-400-40-1] that takes both intensity values and spatial coordinates as input.
- Modality propagation (MP) [27] and coupled sparse representation (CSR) [4].

**Results.** Table 16.1 shows the average SNR values for various different approaches. As we can see, the SNR values for LSDN-1 and LSDN-2 are higher when compared to all the other approaches. It is interesting to see that synthesizing T1 from T2 produces much better results compared to synthesizing T2 from T1. We conjecture that more details of the brain are visible under T2 than T1. As expected, VDN which uses only the intensity values as input performs poorly. While CDN which uses both intensity values and spatial location as input performs better than VDN, it still performs poorly compared to LSDN-2 which has the same size as CDN. This clearly shows that standard MLPs are not appropriate for fusing intensity values and spatial

**FIGURE 16.3**

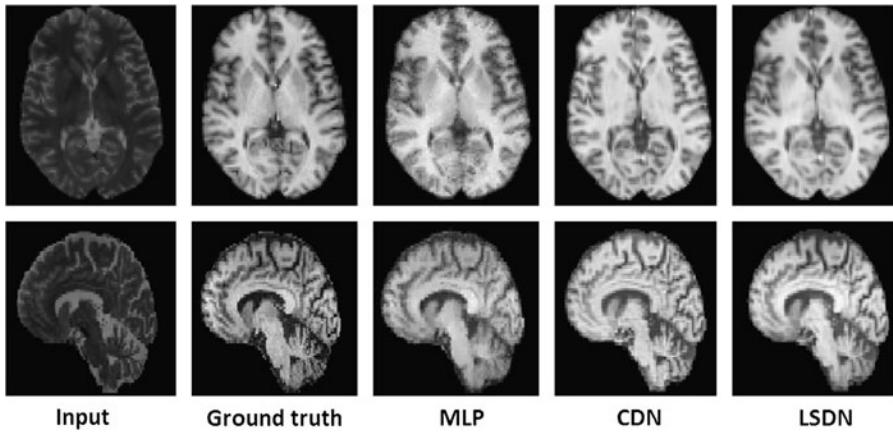
Comparison of various supervised synthesis approaches. T2-MRI synthesis from T1-MRI.

locations. ShrinkConnect reduces the size of LSDN-1 to [30-50-5-1] and the size of LSDN-2 to [30-100-10-1] without losing much in terms of prediction accuracy (refer to the last two rows of [Table 16.1](#)). Note that the size of LSDN-small is same as the size of (LSDN-1 + ShrinkConnect). The poor performance of LSDN-small shows that training a bigger network first and then reducing its size using ShrinkConnect is more effective than directly training the smaller network. The results of the LSDN networks also indicate that increasing the network size would improve the prediction at the cost of computation time. [Figs. 16.3 and 16.4](#) show some visual examples comparing various different approaches.

[Table 16.1](#) also provides the training and synthesis times for different approaches. The experiments were run on a 20-core machine with Intel X5650 processor using MATLAB implementation. The average time LSDN-1 takes to synthesize an image is 29.5 s. With ShrinkConnect, the synthesis time is reduced to 9.2 s per image, which is 26 $\times$  faster than CSR and 100 $\times$  faster than modality propagation.

### 16.3 UNSUPERVISED SYNTHESIS USING MUTUAL INFORMATION MAXIMIZATION

The LSDN-based approach presented in the above section is a supervised synthesis approach as it uses paired training data from source and target modalities to learn the network parameters. In this section, we present an unsupervised cross-modal medical image synthesis approach that works without paired data. Given a source modality image of a subject, this approach synthesizes the corresponding target modality image by making use of target modality images from a different set of subjects.

**FIGURE 16.4**

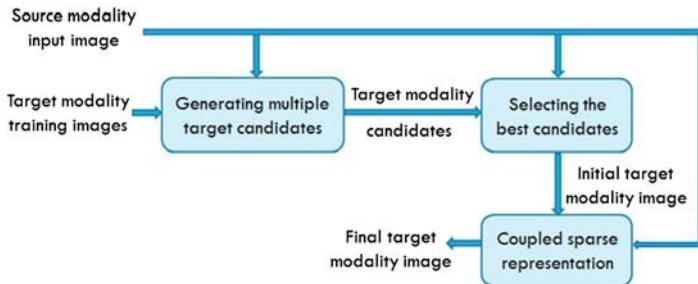
Comparison of various supervised synthesis approaches. T1-MRI synthesis from T2-MRI.

Synthesizing an image with  $D$  voxels can be seen as estimating a  $D$ -dimensional quantity. The following are two possible (extreme) strategies for solving this problem: (i) Estimating the intensities of all the voxels jointly, (ii) Estimating the intensity of each voxel independently. Each strategy has its own advantages and disadvantages. While the first one takes interactions between voxels into account, it is fairly complex given the large set of possible values for each voxel. While the second one simplifies the problem by considering each voxel independently, it does not take into account the image level context.

The presented approach takes advantage of both these strategies by following a two-step synthesis strategy. In the first step, multiple target modality candidate values are generated for each voxel independently. In the second step, a full target modality image is synthesized by selecting the best candidate values jointly for all the voxels by taking the image level context into account. The rationale behind generating multiple candidates in the first step is that at least one of the top  $K$  candidates would be an appropriate value for synthesis when the image context is considered in the second step. This can also be interpreted as restricting the set of possible intensity values for each voxel so that the joint estimation step becomes more tractable.

Since this approach works without paired training data, the quality of the synthesized images would be usually low when compared to the supervised approaches. These results could be improved further by using coupled sparse representation as a refinement step. Recently, various CSR-based approaches have been successfully used for synthesis in the supervised setting [3,4,8,19]. Fig. 16.5 shows the block diagram of the overall synthesis process.

**Notations.** Let  $\Phi^v$  denote the set consisting of voxel  $v$  and its six neighbors that are at unit distance from  $v$ . We use the notation  $\Phi^v(p, q, r)$  to represent the elements of  $\Phi^v$ .

**FIGURE 16.5**

Unsupervised cross-modal image synthesis approach.

Here,  $\Phi^v(p, q, r)$  refers to the voxel  $(v + (p, q, r))$ . We use the notation  $v \sim v'$  to indicate that voxels  $v$  and  $v'$  are neighbors.

### 16.3.1 GENERATING MULTIPLE TARGET MODALITY CANDIDATES

In the first step, given a source modality image  $I_s$ , multiple target modality candidate intensity values are generated for the set  $\Phi^v$  at each voxel independently. To generate the target values for  $\Phi^v$ , a  $d_1 \times d_1 \times d_1$  patch centered on  $v$  extracted from the given source image  $I_s$  is used. If we have paired *Source–Target* images during training, we could possibly learn a predictor/regressor

$$\begin{aligned} g : (\text{Source modality patch at voxel } v) &\longrightarrow \\ &(\text{Multiple target modality candidate values for } \Phi^v). \end{aligned}$$

But, since such paired training data is not available in the unsupervised setting, the target modality candidates are obtained using cross-modal nearest neighbor search. For each  $d_1 \times d_1 \times d_1$  patch from the given source image  $I_s$ ,  $K$  nearest  $d_1 \times d_1 \times d_1$  target patches are obtained by searching across the target modality training images. The intensity values of the center voxel and its neighbors from these  $K$  nearest patches are used as target candidate values for the set  $\Phi^v$ .

For cross-modal nearest neighbor search, we need a similarity measure that is robust to changes in modality. One such measure is the voxel intensity-based mutual information, which has been successfully used in the past as a cross-modal similarity measure for medical image registration [16]. Given two image patches  $A$  and  $B$ , their mutual information is given by

$$MI(A, B) = H(X_a) + H(X_b) - H(X_a, X_b), \quad (16.11)$$

where  $H$  denotes the Shannon entropy function,  $X_a$  and  $X_b$  are random variables representing the voxel intensities in patches  $A$  and  $B$ , respectively.

### 16.3.2 FULL IMAGE SYNTHESIS USING BEST CANDIDATES

In the second step, given  $K$  target modality candidate intensity values for the set  $\Phi^v$  at each voxel, a full target modality image  $\tilde{I}_t$  is synthesized by selecting one among the  $K$  candidates at each voxel. The value of  $\Phi^v(0, 0, 0)$  from the selected candidate is used to synthesize voxel  $v$  in  $\tilde{I}_t$ .

Let  $X_s$  and  $X_t$  be two random variables with support  $\Psi = \{f_1, \dots, f_L\}$ , representing the voxel intensity values of images  $I_s$  and  $I_t$ , respectively. Let  $I_s(v)$  denote the intensity value of voxel  $v$  in image  $I_s$ . Let  $V$  represent the set of all voxels with cardinality  $D$ . Let  $\{\phi^{v1}, \dots, \phi^{vK}\}$  denote the  $K$  target modality candidate values for the set  $\Phi^v$  at voxel  $v$ . Let  $w_{vk} = \mathbb{I}[\text{Candidate } \phi^{vk} \text{ is selected at voxel } v]$ .

Since the candidates have been obtained for each voxel independently, the selection problem is solved jointly for all the voxels based on the following criteria: (i) Mutual information maximization, which is a global criterion, and (ii) Spatial consistency maximization, which is a local criterion.

#### 16.3.2.1 Global Mutual Information Maximization

Motivated by the assumption that regions of similar tissues (and hence similar gray values) in one modality image would correspond to regions of similar gray values in the other modality image (though the values could be different across modalities), mutual information has been successfully used in the past as a cost function for cross-modal medical image registration [16]. Motivated by this, here, mutual information is used as a cost function for cross-modal medical image synthesis. Since we are interested in generating subject-specific scans, the synthesized target modality image  $\tilde{I}_t$  should have high mutual information with the given source modality image  $I_s$ , i.e., the amount of information  $I_s$  and  $\tilde{I}_t$  contain about each other should be maximal. This global criterion helps in transferring the image level structure across modalities.

The mutual information between images  $I_s$  and  $\tilde{I}_t$  is given by

$$MI(X_s, X_t) = H(X_s) + H(X_t) - H(X_s, X_t). \quad (16.12)$$

Since the entropy  $H(X_s)$  is constant for a given source modality image, maximizing  $MI(X_s, X_t)$  is equivalent to maximizing  $H(X_t) - H(X_s, X_t)$ , where

$$\begin{aligned} H(X_t) &= - \sum_{b=1}^L P_b \log[P_b], \\ P_b &= P(X_t = f_b) = \frac{1}{D} \sum_{v \in V} \sum_{k=1}^K w_{vk} \mathbb{I}[\phi^{vk}(0, 0, 0) = f_b], \\ H(X_s, X_t) &= - \sum_{a,b=1}^L P_{ab} \log[P_{ab}], \\ P_{ab} &= P(X_s = f_a, X_t = f_b) = \frac{1}{D} \sum_{v \in V} \sum_{k=1}^K w_{vk} \mathbb{I}[I_s(v) = f_a, \phi^{vk}(0, 0, 0) = f_b]. \end{aligned} \quad (16.13)$$

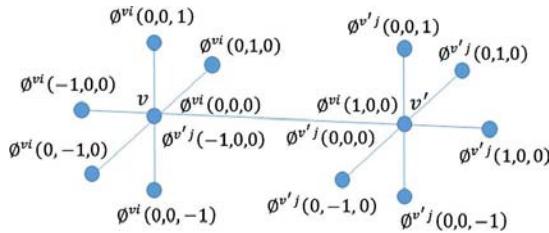


FIGURE 16.6

The values assigned by  $\phi^{vi}$  and  $\phi^{v'j}$  to the sets  $\Phi^v$  and  $\Phi^{v'}$ , respectively.

### 16.3.2.2 Local Spatial Consistency Maximization

Let  $v, v' \in V$  be two neighboring voxels. If we select a candidate  $\phi^{vi}$  at voxel  $v$ , along with assigning the value  $\phi^{vi}(0, 0, 0)$  to voxel  $v$ , it also assigns the value  $\phi^{vi}(v' - v)$  to the neighboring voxel  $v'$ . Similarly if we select a candidate  $\phi^{v'j}$  at voxel  $v'$ , along with assigning the value  $\phi^{v'j}(0, 0, 0)$  to voxel  $v'$ , it also assigns the value  $\phi^{v'j}(v - v')$  to the neighboring voxel  $v$ . Fig. 16.6 shows this pictorially. In this case, we would ideally want the selected candidates  $\phi^{vi}$  and  $\phi^{v'j}$  to be spatially consistent, i.e.,

$$\phi^{vi}(0, 0, 0) = \phi^{v'j}(v - v'), \quad \phi^{v'j}(0, 0, 0) = \phi^{vi}(v' - v). \quad (16.14)$$

Hence, in order to promote spatial consistency among the selected candidates, the following cost function (note the minus sign in the cost) is maximized:

$$SC(\mathbf{W}) = - \sum_{\substack{\mathbf{v}, \mathbf{v}' \in V \\ \mathbf{v} \sim \mathbf{v}'}} \begin{bmatrix} w_{v1} & \dots & w_{vK} \end{bmatrix} \begin{bmatrix} C_{11}^{vv'} & \dots & C_{1K}^{vv'} \\ \vdots & \ddots & \vdots \\ C_{K1}^{vv'} & \dots & C_{KK}^{vv'} \end{bmatrix} \begin{bmatrix} w_{v'1} \\ \vdots \\ w_{v'K} \end{bmatrix}, \quad (16.15)$$

$$\text{where } C_{ij}^{vv'} = \sqrt{(\phi^{vi}(0, 0, 0) - \phi^{v'j}(v - v'))^2 + (\phi^{v'j}(0, 0, 0) - \phi^{vi}(v' - v))^2}.$$

Note that here  $C_{ij}^{vv'}$  is the spatial consistency cost between neighbors  $v$  and  $v'$  when  $\phi^{vi}$  is selected at  $v$  and  $\phi^{v'j}$  is selected at  $v'$ .

### 16.3.2.3 Combined Formulation

Combining the global mutual information cost and the local spatial consistency cost, the candidate selection step can be formulated as the following optimization problem:

$$\begin{aligned} & \underset{\{w_{vk}\}}{\text{maximize}} \quad H(X_t) - H(X_s, X_t) + \lambda SC(\mathbf{W}) \\ & \text{subject to} \quad \sum_{k=1}^K w_{vk} = 1, \quad w_{vk} \in \{0, 1\}, \text{ for } k = 1, \dots, K, \quad \forall v \in V, \end{aligned} \quad (16.16)$$

where  $\lambda$  is a trade-off parameter.

The optimization problem (16.16) is combinatorial in nature due to the binary integer constraints on  $w_{vk}$  and is difficult to solve. Hence, the binary integer constraints are relaxed to positivity constraints to get the following relaxed problem:

$$\begin{aligned} & \underset{\{w_{vk}\}}{\text{maximize}} \quad H(X_t) - H(X_s, X_t) + \lambda SC(\mathbf{W}) \\ & \text{subject to} \quad \sum_{k=1}^K w_{vk} = 1, \quad w_{vk} \geq 0, \text{ for } k = 1, \dots, K, \quad \forall v \in V. \end{aligned} \quad (16.17)$$

#### 16.3.2.4 Optimization

The cost function  $H(X_t) - H(X_s, X_t) + \lambda SC(\mathbf{W})$  is differentiable and its derivative with respect to  $w_{vk}$  can be computed using:

$$\begin{aligned} \frac{dH(X_t)}{dw_{vk}} &= -\sum_{b=1}^L (1 + \log[P(X_t = f_b)]) \frac{d}{dw_{vk}} P(X_t = f_b) \\ &= -\frac{1}{D} \sum_{b=1}^L (1 + \log[P(X_t = f_b)]) \mathbb{I}[\phi^{vk}(0, 0, 0) = f_b] \\ &= -\frac{1}{D} \left( 1 + \log[P(X_t = \phi^{vk}(0, 0, 0))] \right), \\ \frac{dH(X_s, X_t)}{dw_{vk}} &= -\sum_{a,b=1}^L (1 + \log[P_{ab}]) \frac{dP_{ab}}{dw_{vk}} \\ &= -\frac{1}{D} \sum_{a,b=1}^L (1 + \log[P_{ab}]) \mathbb{I}[I_s(v) = f_a, \phi^{vk}(0, 0, 0) = f_b] \\ &= -\frac{1}{D} \left( 1 + \log[P(X_s = I_s(v), X_t = \phi^{vk}(0, 0, 0))] \right), \\ \frac{dSC(\mathbf{W})}{dw_{vk}} &= \sum_{v' \sim v} \left( \sum_{p=1}^K C_{kp}^{vv'} w_{v'p} \right). \end{aligned} \quad (16.18)$$

The optimization problem (16.17) has a differentiable cost function with linear equality and inequality constraints. Hence, it can be solved using reduced gradient ascent approach, in which the gradient computed from (16.18) is projected onto the constraint set in each iteration. Once we obtain  $w_{vk}$ , we use  $\phi^{vk^*}(0, 0, 0)$  to synthesize voxel  $v$  in  $\tilde{I}_t$ , where  $k^* = \text{argmax}_k w_{vk}$ .

Note that though the optimization problem (16.17) is a relaxation of problem (16.16), the unit  $\ell_1$ -norm constraints on the weights promote a sparse solution [2,6] pushing most of  $w_{vk}$  toward zero. Since the cost function in (16.17) is non-convex, the reduced gradient ascent approach is not guaranteed to find the global optimum. In the experiments, we use the local optimum obtained by initializing all

the variables  $w_{vk}$  with a value of  $\frac{1}{K}$ . This initialization can be interpreted as giving equal weight to all the  $K$  candidates at the beginning of the optimization. During the optimization, in each iteration, along with projecting the gradient on to the constraint set, the ascent direction is also adjusted such that the variables satisfying  $w_{vk} = 0$  remain as zero. In each iteration, the learning rate is chosen as the maximum possible value such that none of the variables  $w_{vk}$  goes below zero.

### 16.3.3 REFINEMENT USING COUPLED SPARSE REPRESENTATION

Recently, coupled sparse representation has been shown to be a powerful model when dealing with coupled signal spaces in applications like super-resolution [21,25,26], cross-modal image synthesis [4,8,19], etc. Sparse representations are robust to noise and artifacts present in the data. Hence, CSR is used to refine the synthesized target modality image  $\tilde{I}_t$  and generate the final target modality image  $I_t$ .

At each voxel  $v \in V$ , small  $d_2 \times d_2 \times d_2$  patches are extracted from the given source modality image  $I_s$  and the synthesized target modality image  $\tilde{I}_t$ . Let  $Z_v^s$  and  $Z_v^t$  denote the patches at voxel  $v$  from images  $I_s$  and  $\tilde{I}_t$ , respectively. Using  $\{(Z_v^s, Z_v^t) \mid v \in V\}$  as signal pairs from the source and target modalities, coupled sparse representation can be formulated as the following optimization problem:

$$\begin{aligned} & \underset{\mathbf{D}_s, \mathbf{D}_t, \{\alpha_v\}}{\text{minimize}} \sum_{v \in V} \left( \|Z_v^s - \mathbf{D}_s \alpha_v\|_2^2 + \|Z_v^t - \mathbf{D}_t \alpha_v\|_2^2 \right) \\ & \text{subject to } \|\alpha_v\|_0 \leq T_0 \quad \forall v \in V, \\ & \quad \|\mathbf{D}_s(:, j)\|_2 = 1, \quad \|\mathbf{D}_t(:, j)\|_2 = 1 \quad \forall j, \end{aligned} \tag{16.19}$$

where  $\mathbf{D}_s$  and  $\mathbf{D}_t$  are over-complete dictionaries with  $M$  atoms in the source and target modalities respectively,  $\alpha_v$  is the coupled sparse code for signals  $Z_v^s$  and  $Z_v^t$  in their respective dictionaries, and  $T_0$  is the sparsity parameter.

The dictionaries  $\mathbf{D}_s$ ,  $\mathbf{D}_t$  and the coupled sparse codes  $\alpha_v$  can be learned by solving the optimization problem (16.19) using the K-SVD [1] algorithm with explicitly re-normalizing the columns of  $\mathbf{D}_s$  and  $\mathbf{D}_t$  to unit norm after each iteration. Once the dictionaries and sparse codes are obtained, the target modality patches are reconstructed at every voxel using  $\hat{Z}_v^t = \mathbf{D}_t \alpha_v$ , and the center voxel value from  $\hat{Z}_v^t$  is used to synthesize voxel  $v$  in the final target modality image  $I_t$ .

### 16.3.4 EXTENSION TO SUPERVISED SETTING

The above described unsupervised synthesis approach can be extended to the supervised setting by simply replacing the cross-modal nearest neighbor search in the candidate generation step with source-modal nearest neighbor search. For each voxel  $v \in V$ , a  $d_3 \times d_3 \times d_3$  patch centered on  $v$  is extracted from the given source modality test image  $I_s$  and  $K$  nearest  $d_3 \times d_3 \times d_3$  patches are found from the source modality training images using standard Euclidean distance. Note that a nearest neighbor search (even within source modality) needs to be performed because the training and

**Table 16.2** SNR values for the unsupervised synthesis approach

SNR (T1→T2)		SNR (T2→T1)	
No CSR	CSR	No CSR	CSR
12.78 ± 0.61	13.35 ± 0.65	16.23 ± 0.64	16.52 ± 0.70

test images are from different subjects. Once the  $K$  nearest source modality training patches are found, the corresponding target modality training patches are used for generating the target modality candidates for  $\Phi^v$ .

In the CSR step, the paired training data is used to learn coupled dictionaries, and the learned dictionaries are used for refining the synthesized target modality images.

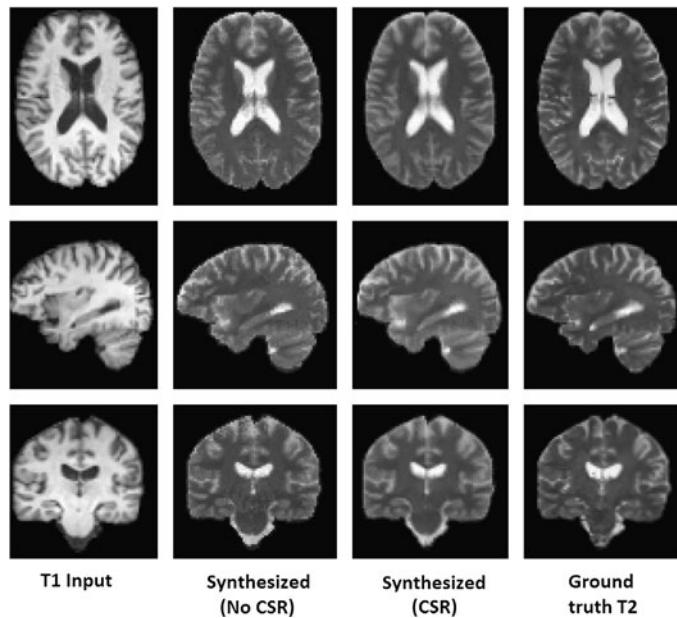
### 16.3.5 EXPERIMENTS

In this section, we evaluate the above described image synthesis approach by generating T1-MRI scans from T2-MRI scans, and vice versa. We use the same dataset and evaluation metric described in Section 16.2.3. We follow the leave-one-subject-out test setting in which one subject is used for testing and the target modality scans of the remaining 18 subjects are used for training.

**Implementation Details.** Since exhaustively searching the images (to find nearest neighbors) is highly computational and all the images in the dataset are roughly aligned, we restricted the search in each image to an  $h \times h \times h$  (with  $h = 7$ ) region around the voxel of interest. The patch sizes  $d_1$  and  $d_3$  used for cross-modal and source-modal nearest neighbor searches were chosen as 9 and 3, respectively. The patch size used for cross-modal search is much larger than the patch size used for source-modal search because for reliable estimation of mutual information, the patch should have sufficient number of voxels. The number of nearest neighbors  $K$  was chosen as 10. Since MRI scans have a high dynamic range, the mutual information computed using the original intensity values would be highly unreliable. Hence, we quantized the intensity values to  $L = 32$  levels for computing the mutual information.

Note that the spatial consistency cost (16.15) involves the sum of errors over all pairs of neighboring voxels. As the number of pairs in an image is very large, the magnitude of (16.15) will be much higher than the mutual information cost. Hence, we chose the value of parameter  $\lambda$  in (16.17) such that the mutual information and spatial consistency costs have values that are of the same order of magnitude. For the unsupervised setting, we used  $\lambda = 10^{-8}$  and for the supervised setting we used  $\lambda = 10^{-7}$ . For the CSR step, we used patches with  $d_2 = 3$ . The sparsity parameter  $T_0$  and the number of dictionary atoms  $M$  were chosen as 5 and 500, respectively.

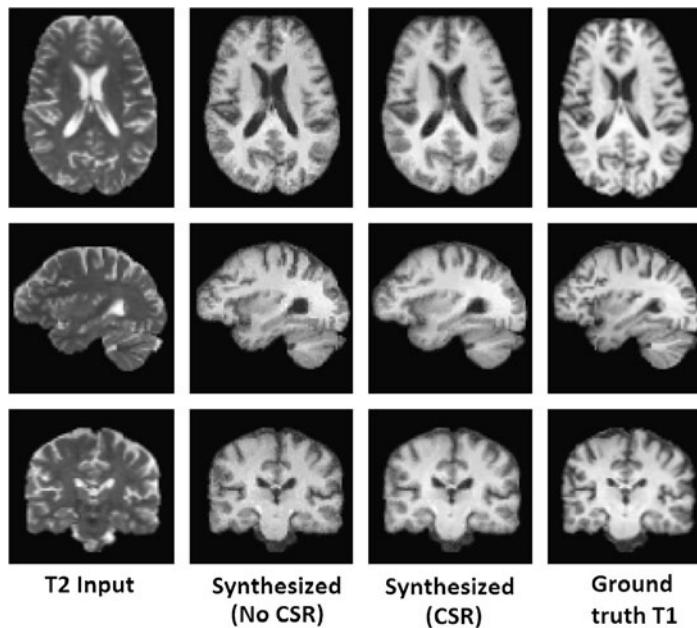
**Results.** Table 16.2 shows the average SNR values. Figs. 16.7 and 16.8 show some visual examples comparing the unsupervised synthesis results with the ground truth. Similar to the LSDN results, synthesis of T1-MRI from T2-MRI produces better results compared to the synthesis of T2-MRI from T1-MRI. Images without CSR look a bit noisy compared to the images with CSR (please zoom Figs. 16.7 and 16.8).

**FIGURE 16.7**

Comparison of the unsupervised synthesis results with ground truth. T2-MRI synthesis from T1-MRI.

**Comparisons.** To the best of our knowledge this approach is the first unsupervised cross-modal synthesis approach, and hence there is no existing state-of-the-art to compare with under the unsupervised setting. To show the effectiveness of the candidate selection approach presented in Section 16.3.2, we compare it with the following methods:

1. **First nearest neighbor (F-NN).** We use the center voxel value of the first nearest neighbor for synthesis.
2. **Average of nearest neighbors (A-NN).** We use the average of the center voxel values of all the  $K$  nearest neighbors for synthesis.
3. **Candidate selection using only mutual information (MI-only).** We use the center voxel value of the best candidate selected by optimizing only the global mutual information cost. This is equivalent to removing  $SC(W)$  from optimization problem (16.17).
4. **Candidate selection using only spatial consistency (SC-only).** We use the center voxel value of the best candidate selected by optimizing only the local spatial consistency cost. This is equivalent to removing  $H(X_t) - H(X_s, X_t)$  from optimization problem (16.17).

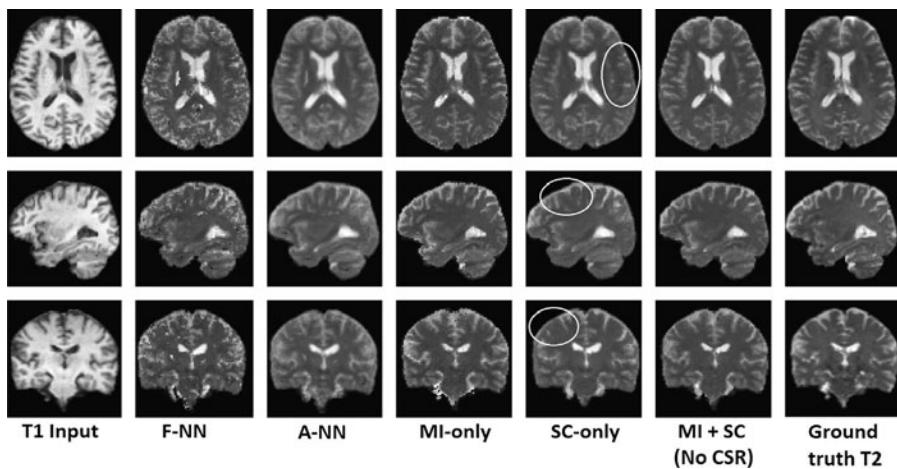
**FIGURE 16.8**

Comparison of the unsupervised synthesis results with ground truth. T1-MRI synthesis from T2-MRI.

**Table 16.3** SNR values for various candidate selection approaches

Approach	SNR (T1→T2)	SNR (T2→T1)
F-NN	$10.10 \pm 0.45$	$13.30 \pm 0.34$
A-NN	$12.41 \pm 0.61$	$15.45 \pm 0.57$
MI only	$11.72 \pm 0.61$	$14.88 \pm 0.59$
SC only	$12.11 \pm 0.56$	$15.19 \pm 0.53$
MI + SC (No CSR)	<b><math>12.78 \pm 0.61</math></b>	<b><math>16.23 \pm 0.64</math></b>

**Table 16.3** compares various candidate selection approaches in terms of average SNR values and [Fig. 16.9](#) shows some visual examples. We can clearly see that the presented approach (MI + SC, No CSR) gives the best synthesis results. The low SNR values of F-NN and A-NN methods indicate that directly using the first nearest neighbor or the average of  $K$  nearest neighbors is not sufficient for obtaining good synthesis results. While the F-NN method produces very noisy images with spurious structures, the A-NN method produces blurred images. The low SNR values of MI-only and SC-only methods suggest that using only the global mutual information criterion or only the local spatial consistency criterion would produce inferior synthesis results compared to using both criteria. While the images synthesized by the

**FIGURE 16.9**

Comparison of various candidate selection approaches.

**Table 16.4** Comparison with supervised approaches

Method	SNR (T1 → T2)	SNR (T2 → T1)
CSR [4]	$13.72 \pm 0.64$	$15.24 \pm 0.85$
MP [27]	$13.64 \pm 0.67$	$15.13 \pm 0.88$
LSDN [14]	$14.93 \pm 0.73$	$17.39 \pm 0.91$
MI + SC + CSR (unsupervised)	$13.35 \pm 0.65$	$16.52 \pm 0.70$
MI + SC + CSR (supervised)	<b><math>15.30 \pm 0.94</math></b>	<b><math>18.33 \pm 1.15</math></b>

MI-only method are corrupted by salt and pepper type noise, the images synthesized by the SC-only method are missing a lot of structural details (see the circled areas in Fig. 16.9). The presented approach, which uses both MI and SC criteria, is able to get rid of the noise without loosing the structural details.

**Supervised Synthesis Results.** Table 16.4 compares the synthesis results of the presented approach under the supervised and unsupervised settings with modality propagation [27], coupled sparse representation [4] and location-sensitive deep network [14] methods. We can clearly see that the presented approach outperforms all the three methods under the supervised setting. In fact, the unsupervised approach is able to outperform the supervised modality propagation and CSR methods while synthesizing T1-MRI from T2-MRI.

**Computation Time.** When ran on a machine with Intel X5650 processor (2.66 GHz, 20 cores), the candidate generation step took 17 min, the candidate selection step took 15 min, and the sparse coding step took 7 min.

---

## 16.4 CONCLUSIONS AND FUTURE WORK

In this chapter, we presented two approaches for cross-modal medical image synthesis. The first approach is based on a deep network architecture, referred to as location-sensitive deep network, which integrates spatial information with intensity-based features in a principled manner. LSDN models the first hidden layer nodes as products of certain location-sensitive functions and nonlinear features computed from voxel intensity values. LSDN is a supervised synthesis approach since paired training data is used to learn the network parameters. Along with LSDN, we also presented a sparse coding-based network simplification approach, referred to as ShrinkConnect, to reduce the size of LSDN without losing much in terms of prediction accuracy.

The second approach is based on the principle of mutual information maximization. Given a source modality image, this approach first generates multiple target modality candidate values independently for each voxel by performing nearest neighbor search over the training database. Then, the best candidate values are selected jointly for all the voxels by simultaneously maximizing a global mutual information-based cost function and a local spatial consistency cost function. Finally, coupled sparse representation is used to further refine the synthesized images. This approach can be used under both the unsupervised and supervised settings. We experimentally demonstrated the synthesis capabilities of both the approaches by generating T1-MRI scans from T2-MRI scans and vice versa.

In our experiments, we mainly focused on MR contrast synthesis. In the future, we will apply the presented approaches to other medical imaging modalities. We also plan to use the synthesized images for improving image analysis algorithms like detection and segmentation.

---

## ACKNOWLEDGEMENTS

We would like to acknowledge Siemens Healthcare Technology Center for funding this research.

---

## REFERENCES

1. M. Aharon, M. Elad, A. Bruckstein, K-SVD: an algorithm for designing overcomplete dictionaries for sparse representation, *IEEE Trans. Signal Process.* 54 (11) (2006) 4311–4322.
2. E.J. Candès, J.K. Romberg, T. Tao, Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information, *IEEE Trans. Inf. Theory* 52 (2) (2006) 489–509.
3. T. Cao, V. Jovic, S. Modla, D. Powell, K. Czumek, M. Niethammer, Robust multimodal dictionary learning, in: MICCAI, 2013.
4. T. Cao, C. Zach, S. Modla, D. Powell, K. Czumek, M. Niethammer, Multi-modal registration for correlative microscopy using image analogies, *Med. Image Anal.* 18 (6) (2014) 914–926.

5. O. Commowick, S.K. Warfield, G. Malandain, Using Frankenstein's creature paradigm to build a patient specific atlas, in: MICCAI, 2013.
6. D.L. Donoho, Compressed sensing, *IEEE Trans. Inf. Theory* 52 (4) (2006) 1289–1306.
7. A. Hertzmann, C.E. Jacobs, N. Oliver, B. Curless, D.H. Salesin, Image analogies, in: Annual Conference on Computer Graphics and Interactive Techniques, 2001.
8. D. Huang, Y.F. Wang, Coupled dictionary and feature space learning with applications to cross-domain image synthesis and recognition, in: ICCV, 2013.
9. J.E. Iglesias, E. Konukoglu, D. Zikic, B. Glocker, K.V. Leemput, B. Fischl, Is synthesizing MRI contrast useful for inter-modality analysis?, in: MICCAI, 2013.
10. A. Jog, A. Carass, J.L. Prince, Improving magnetic resonance resolution with supervised learning, in: ISBI, 2014.
11. A. Jog, S. Roy, A. Carass, J.L. Prince, Magnetic resonance image synthesis through patch regression, in: ISBI, 2013.
12. E. Konukoglu, A.J.W. van der Kouwe, M.R. Sabuncu, B. Fischl, Example-based restoration of high resolution magnetic resonance image acquisitions, in: MICCAI, 2013.
13. D. Ma, V. Gulani, N. Seiberlich, K. Liu, J.L. Sunshine, J.L. Duerk, M.A. Griswold, Magnetic resonance fingerprinting, *Nature* 495 (2013) 187–192.
14. H.V. Nguyen, S.K. Zhou, R. Vemulapalli, Cross-domain synthesis of medical images using efficient location-sensitive deep network, in: MICCAI, 2015.
15. J. Nuyts, G. Bal, F. Kehren, M. Fenchel, C. Michel, C. Watson, Completion of a truncated attenuation image from the attenuated PET emission data, *IEEE Trans. Med. Imaging* 32 (2) (2013) 237–246.
16. J.P.W. Pluim, J.B.A. Maintz, M.A. Viergever, Mutual information-based registration of medical images: a survey, *IEEE Trans. Med. Imaging* 22 (8) (2003) 986–1004.
17. F. Rousseau, P.A. Habas, C. Studholme, A supervised patch-based approach for human brain labeling, *IEEE Trans. Med. Imaging* 30 (10) (2011) 1852–1862.
18. S. Roy, A. Carass, A. Jog, J.L. Prince, J. Lee, MR to CT registration of brains using image synthesis, in: SPIE Medical Imaging, 2014.
19. S. Roy, A. Carass, J.L. Prince, Magnetic resonance image example-based contrast synthesis, *IEEE Trans. Med. Imaging* 32 (12) (2013) 2348–2363.
20. S. Roy, A. Carass, N. Shiee, D.L. Pham, J.L. Prince, MR contrast synthesis for lesion segmentation, in: ISBI, 2010.
21. A. Rueda, N. Malpica, E. Romero, Single-image super-resolution of brain MR images using overcomplete dictionaries, *Med. Image Anal.* 17 (1) (2013) 113–132.
22. J.A. Tropp, A.C. Gilbert, M.J. Strauss, Simultaneous sparse approximation via greedy pursuit, in: ICASSP, 2005.
23. R. Vemulapalli, H.V. Nguyen, S.K. Zhou, Unsupervised cross-modal synthesis of subject-specific scans, in: ICCV, 2015.
24. S. Wang, L. Zhang, Y. Liang, Q. Pan, Semi-coupled dictionary learning with applications to image super-resolution and photo-sketch synthesis, in: CVPR, 2012.
25. J. Yang, Z. Wang, Z. Lin, X. Shu, T.S. Huang, Bilevel sparse coding for coupled feature spaces, in: CVPR, 2012.
26. J. Yang, J. Wright, T.S. Huang, Y. Ma, Image super-resolution via sparse representation, *IEEE Trans. Image Process.* 19 (11) (2010) 2861–2873.
27. D.H. Ye, D. Zikic, B. Glocker, A. Criminisi, E. Konukoglu, Modality propagation: coherent synthesis of subject-specific scans with data-driven regularization, in: MICCAI, 2013.

---

**NOTE**

1. [Fig. 16.1](#) shows the intensity mapping of voxels within a small window across modalities. Small window size means that all the voxels are closer to a specific  $(x, y, z)$  location. Hence, the importance of smaller window size is effectively same as the importance of a voxel's spatial location.

This page intentionally left blank

# Natural Language Processing for Large-Scale Medical Image Analysis Using Deep Learning

# 17

**Hoo-Chang Shin, Le Lu, Ronald M. Summers**

*National Institutes of Health Clinical Center, Bethesda, MD, United States*

## CHAPTER OUTLINE

---

<b>17.1</b>	<b>Introduction</b>	406
<b>17.2</b>	<b>Fundamentals of Natural Language Processing</b>	407
17.2.1	Pattern Matching	407
17.2.1.1	<i>Structuring a Free Form Text</i>	408
17.2.1.2	<i>Extracting the Image Information</i>	409
17.2.1.3	<i>Anonymization</i>	409
17.2.1.4	<i>Negation Detection</i>	409
17.2.2	Topic Modeling	410
<b>17.3</b>	<b>Neural Language Models</b>	411
17.3.1	Word Embeddings	411
17.3.2	Recurrent Language Model	412
<b>17.4</b>	<b>Medical Lexicons</b>	414
17.4.1	UMLS Metathesaurus	414
17.4.2	RadLex	414
<b>17.5</b>	<b>Predicting Presence or Absence of Frequent Disease Types</b>	414
17.5.1	Mining Presence/Absence of Frequent Disease Terms	414
17.5.2	Prediction Result and Discussion	415
17.5.2.1	<i>Discussion</i>	417
<b>17.6</b>	<b>Conclusion</b>	419
<b>Acknowledgements</b>		419
<b>References</b>		419

## CHAPTER POINTS

---

- Structured information can be obtained from the radiology reports in Picture Archiving and Communication Systems (PACS) using natural language processing (NLP)
  - Recent advances in deep learning enables us to analyze large amount of images effectively, though collecting and annotating many medical images has been always a challenge
  - Using NLP, lots of radiology images can be collected, annotated, and analyzed, without much human manual efforts
- 

## 17.1 INTRODUCTION

Recent advances in deep learning methods enable us to analyze complex data types such as images in a much larger scale more effectively. Computer vision research and application has made leaping improvements with the state-of-the-art deep learning methods, setting new records every year.

Despite the tremendous progress in computer vision, there has not been an attempt for machine learning on very large-scale medical image databases. The foremost challenges in performing large-scale medical image analysis somewhat comparable to the scale of recent computer vision studies are:

- The challenge in collecting the images
- The challenge in annotating such collected images

The ImageNet [9] (now-standard computer vision large-scale image dataset) is constructed by collecting images from the Internet, and annotating them using crowdsourcing. Such approach is infeasible for medical images, due to:

- Privacy concerns
- Lack of expertise (of crowd-sourced annotators)

On the other hand, large collections of radiology images and reports are stored in many modern hospitals' Picture Archiving and Communication Systems (PACS). The invaluable semantic diagnostic knowledge inhabiting the mapping between hundreds of thousands of clinician-created high-quality text reports and linked image volumes remains largely unexplored.

Natural language processing (NLP) provides methods that can help the conversion of text into a structured representation, thereby enabling computers to derive meaning from human input, such as text or speech. Applied on radiology reports, NLP methods can help us to automatically identify and extract relevant informations from the reports so that (i) image collection and (ii) image annotation can be automated.

A radiology report provides a diagnostic imaging referral, and is used for communication and documentation purposes. Although there exist some guidelines for reporting diagnostic imaging, a report mostly contains free text, often organized in a few standard sections. For that reason, it is challenging to convert such free text

into a representation that computer can efficiently manage and interact. NLP is a set of method to enable that, to (i) convert unstructured text into a structured form, so that (ii) computer can efficiently store and interact with the information stored.

In this chapter, we show and demonstrate how we can use NLP to collect large radiology image dataset by analyzing radiology text reports. Furthermore, we introduce some existing NLP tools and ontology databases which can be used to annotate large dataset of radiology images. With NLP, images already stored in most hospital PACS database can be retrieved and analyzed, so that we can leverage upon the vast learning capacity of recent deep learning methods.

---

## 17.2 FUNDAMENTALS OF NATURAL LANGUAGE PROCESSING

### 17.2.1 PATTERN MATCHING

The most fundamental method and an integral part of many other NLP tasks is pattern matching. A pattern is a sequence of characters that can be matched, character for character, to a given text. Pattern matching can be often performed with regular expressions (called REs, or regexes, or regex patterns) [20,4], which are essentially a tiny, highly specialized programming language embedded inside most of major programming languages such as Python or Java. A set of possible strings can be matched by specifying the rules using this little programming language.

There are some metacharacters that can be used to match specific character patterns. For example, [ and ] are used for specifying a character class. Characters can be listed individually, or a range of characters can be indicated by giving two characters and separating them by a ‘-’. For example, any lowercase letters can be matched with a regular expression of [ a - z ].

By including a ‘^’ as the first character of the class, any characters not listed within the class by complementing the set can be matched. For example, [ ^0 ] will match any character except ‘0’.

One of the most important metacharacters for regex is the backslash \, which can be followed by various characters to signal different special sequences. It is also used to escape all the metacharacters so that they can be matched in patterns. For example, [ or \ can be matched by preceding them with a backslash to remove their special meaning.

Some of the special sequences beginning with ‘\’ represent predefined sets of characters that are often useful, such as the set of digits, the set of letters, or the set of anything that is not a whitespace. The following predefined special sequences are a subset of those:

- \d – matches any decimal digit; equivalent to [ 0 - 9 ]
- \D – matches any non-digit character; equivalent to [ ^0 - 9 ]
- \s – matches any whitespace character; equivalent to [ \t\n\r\f\v ]
- \S – matches any non-whitespace character; equivalent to [ ^\t\n\r\f\v ]

```

Dictating Radiologist: Jane Doe 11/11/2011
Transcribed by:
Signed by:
Signed by: John Doe 11/11/2011
Approved by: John Doe 11/11/2011 11:11 AM"
123456,"MR1000012340","MRI Results

Exam: Exam Date: Accession #: Ordered By:
MRI Brain - 11/11/2011 MR1000012340 HOMER, SIMPSON
Patient: LISA SIMPSON

REASON FOR EXAM: Stroke

CLINICAL HISTORY: Glycogen storage disease. Marked leukomalacia bilaterally ...

TECHNIQUE: Sagittal T1, axial T1 pre- and postcontrast axial FLAIR, axial T2, axial gradient echo, coronal T1 postcontrast, sagittal T1 3-D postcontrast ...

FINDINGS: There is evidence of left parietal encephalomalacia consistent with known history of prior stroke. Small focal area of hemosiderin deposition along the lateral margins of the left lateral ventricle series 601 image 18. The orbits and globes are normal ...

IMPRESSION: 1. Diffuse white matter encephalomalacia that is stable when compared to XX/XX/XXXX examination but has minimally increased ...

```

**FIGURE 17.1**

An example of a radiology report stored as a free-form text. Pattern matching can be applied to the free-form text of radiology report to (i) convert to a structured text and to (ii) extract image mentioned in the report (highlighted blue).

- \w – matches any alphanumeric character; equivalent to [ a - zA - Z0 - 9\_ ]
- \W – matches any non-alphanumeric character; equivalent to [ ^a - zA - Z0 - 9\_ ]

### **17.2.1.1 Structuring a Free Form Text**

With pattern matching, a free-form text downloaded from a PACS containing many radiology reports can be automatically structured. An example of a radiology report stored in a free-form text is shown in Fig. 17.1. If many reports are stored in a free-form text, pattern matching can be applied to divide them into the reports of separate incidents. For example, a regex like the following example can be built to match the start of a radiology report:

```

re1 = '( Dictating )(\s+)( Radiologist )(:)'
re2 = '(\s+'
re3 = '((?:[ a-z ][ a-z ]+))(\s+)((?:[ a-z ][ a-z ]+))'
re4 = '(\s+'
re5 = '(?:[0]?[1-9][1][012])[-:\.\. ]'
re6 = '(?:(?:[0-2]?\d{1})|(?:[3][01]{1}))[-:\.\. ]'
re7 = '(?:(?:[1]{1}\d{1}\d{1}\d{1})|(?:[2]{1}\d{3}))'
re8 = '(?![\d])'
re = re1+re2+re3+re4+'('+re5+re6+re7+')'+re8

```

In the above regex, `re1` matches the exact pattern ‘Dictating Radiologist:’, `re2` a white space afterward, `re3` a pattern for a name, `re4` a following white space, and `re5` to `re8` are for matching a calendar date pattern.

### 17.2.1.2 Extracting the Image Information

An important pattern to extract image information is the *accession number*, *series number*, and *image (slice) number*. The information about the image modality (MR, CT, etc.) can be relevant too, and that can be extracted from the accession number: accession numbers start with MR for studies with magnetic resonance imaging (MRI), CT for studies with computed tomography (CT), and NM for those with positron emission tomography (PET) imaging. Patient identification number (ID) can be relevant if patient history is of an interest.

Patient ID, accession number are listed in the sixth line of the radiology report shown in Fig. 17.1, which can be matched by regex like:

```
re = '(\d+)(\s+)(MR)(\d+)(\s+)((?:[a-z][a-z]+))'
```

Once the accession number (and possibly the patient ID too) is extracted with regex like shown above, the series number and the image number can be extracted with regex like the following example to get the images mentioned in the report:

```
re1 = ((?:[a-z][a-z]+))(\s+)
re2 = (series)(\s+)(\d+)(\s+)
re3 = (image)(\s+)(\d+)(\.)
re = re1+re2+re3+re4
```

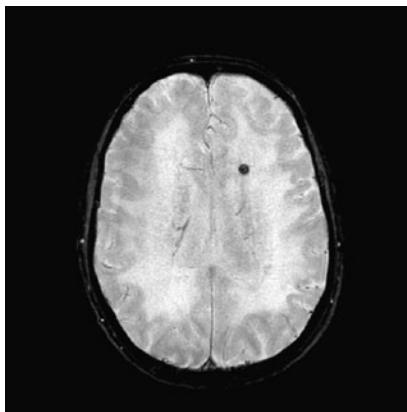
An example of a pulled image from the radiology report shown in Fig. 17.1, using the pattern matching methods listed above, is shown in Fig. 17.2. We can indeed see that the pulled image is a magnetic resonance image indicating a stroke.

### 17.2.1.3 Anonymization

Before conducting a research with patient data, the obtained data should be properly anonymized. Anonymization can be also performed using pattern matching, for example, matching the likely patterns of names, patient IDs, dates, telephone numbers, or any other unique identifying number. Moreover, machine learning methods can be applied to complement the pattern matching methods, in order to achieve better performance. More complete studies about anonymization can be found in [2,26,12,39].

### 17.2.1.4 Negation Detection

Pattern matching is also used for negation detection algorithms such as the popular NegEx [6] algorithm. Some common negation phrases like *no signs of*, *unlikely*,

**FIGURE 17.2**

---

An example of pulled image from the radiology report shown in Fig. 17.1 with pattern matching.

*absence of, or no evidence of* are found with the pattern matching. These patterns are associated with the following disease mentioning to form negated mentioning of the disease. Nonetheless, machine learning methods can be used for negation detection as well to complement the pattern matching. More details on negation detection can be found in [6,8,21,37].

### 17.2.2 TOPIC MODELING

Topic modeling is a useful method to summarize a dataset with large text corpus and to obtain gross insight over the dataset. Many popular topic modeling methods characterize document content based on key terms and estimate topics contained within documents. For example, documents associated with the topic *MRI of brain tumor* would comprise one cluster with key terms such as *axial, contrast, mri, sagittal, enhancement, etc.* Documents associated with *arthritis imaging* would comprise another cluster with key terms like *joint, views, hands, lateral, feet, etc.*

One of the most popular methods for topic modeling is Latent Dirichlet Allocation (LDA), originally proposed in [5] to find latent topic models for a collection of text documents such as newspaper articles. There are some other popular methods for document topic modeling, such as Probabilistic Latent Semantic Analysis (pLSA) by [13] and Non-negative Matrix Factorization (NMF) by [18]. Simpler methods such as *k-means clustering* or *k-nearest neighbor* are also often used for topic modeling. More details of different topic modeling methods and comparisons of them can be found in [33,36,38].

For the application of topic modeling in radiology text, document topic modeling using LDA was applied to large corpus of radiology report in [30]. Combined with the

images extracted and identified to be associated with each radiology report using pattern matching, convolutional neural networks (CNNs) are trained to classify images into document topics. Given the document topic is useful, it can provide a first-level understanding of a new image classified into a document topic of radiology reports.

---

## 17.3 NEURAL LANGUAGE MODELS

Statistical language modeling aims to predict the next word in textual data given context. Therefore, we deal with sequential data prediction problem when constructing language models. Traditionally, such statistical models were the approaches that are very specific for language domain. For instance, assuming that natural language sentences can be described by parse trees, or that they can be considered as morphology of words, syntax, and semantics. Most widely used language models are n-gram models, which assume that language consists of sequences of words that form sentences.

Likewise in computer vision, the recent advances in deep learning has changed NLP research. Some of the previous state-of-the-art NLP methods are replaced by deep learning based approaches, and the trend is continuing. In this section we will briefly introduce neural language modeling – a more recent approach to NLP using neural networks, and discuss how they are used in radiology text mining applications.

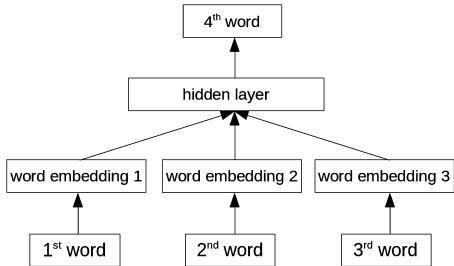
### 17.3.1 WORD EMBEDDINGS

Most of the machine learning algorithms benefits from rich feature representations – the better the feature representations are (usually), the better a machine learning algorithm performs. On the contrary, even very sophisticated algorithms are basically limited by the dimension of the features representing the data. Traditionally in NLP, words in text were given a unique ID (e.g., an integer number). Such encodings are not only arbitrary, but also provide no useful information regarding the relationships or similarities between the individual symbols.

Word embedding is a technique to convert words in to vectors, where words with similar vectors would have close cosine distance, and can improve many NLP tasks. By representing words as vectors, the data representation is less sparse, thereby statistical models can be more successfully trained with less data.

Words can be converted to vectors in an unsupervised manner using feed-forward neural networks. A feed-forward neural network such as shown in Fig. 17.3 can be used to learn word embeddings from a large corpus of text. The feed-forward neural network shown in Fig. 17.3 learns the neural word embeddings by learning to predict the next word after seeing three consecutive words in the text corpus.

There are many options and advanced techniques to learn better word embeddings, which can be reviewed in [23,22,25,24]. Fig. 17.4 shows some example of the words trained with the word-to-vector model of [23,22,25,24], applied to large cor-

**FIGURE 17.3**

An example of a feed-forward neural network architecture to learn word embeddings.

"heart"	"brain"	"liver"
lungs 0.526600	t1 0.615066	spleen 0.759884
mediastinum 0.517008	mri 0.595027	gallbladder 0.648075
consolidating 0.486605	sagittal 0.580841	hepatomegaly 0.642022
pa 0.449816	flair 0.565445	gallstones 0.611837
chest 0.433362	t2 0.555053	pancreas 0.608356
infiltrates 0.428404	axial 0.554040	gallstone 0.606063
hyperinflated 0.413326	spgr 0.520954	steatosis 0.601081
cardiomegaly 0.410785	weighted 0.502047	dome 0.594812
hyperlucent 0.400836	technique 0.487768	portal 0.570008
pectus 0.396142	astrocytoma 0.480527	ascites 0.551869
great 0.395712	gbm 0.476956	hepatosplenomegaly 0.540501
ectatic 0.394560	gradient 0.476593	hepatic 0.537453
shifted 0.389205	oligodendrogioma 0.465892	cirrhosis 0.530389
ray 0.389091	postcontrast 0.463686	fatty 0.522134

**FIGURE 17.4**

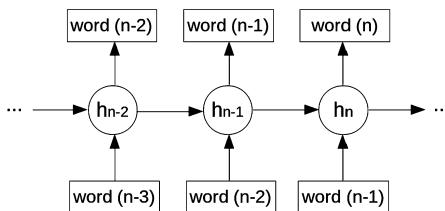
Word-to-vector models trained on a collection of radiology reports. Search words (with quotes) and their closest words in vector-space cosine similarity (the higher the better) are listed in a descending order. Image is adapted from [29].

pus of radiology text [30]. Some search words (with quotes) and their closest words in vector-space cosine similarity (the higher the better).

We can see that words with similar meanings are reasonably well encoded with the word embeddings. Word embedding was applied to radiology text reports in [30] to overcome ambiguities of radiology words with similar meanings, and to generate word vectors of keywords describing radiology images.

### 17.3.2 RECURRENT LANGUAGE MODEL

Neural language models based on feed-forward neural networks have a limitation that the input dimension is fixed. Still, many attempts to obtain such statistical mod-

**FIGURE 17.5**

An example of a recurrent neural network language model.

els involve approaches that are very specific for language domain – for example, assumption that natural language sentences can be described by parse trees, or that we need to consider morphology of words, syntax and semantics.

Even the most widely used and general models, based on n-gram statistics, assume that language consists of sequences of atomic symbols – words – that form sentences, and where the end of sentence symbol plays important and very special role. For example, for the word embedding model shown in Fig. 17.3, once we fix the window size (3: *1st word ... 3rd word*), the entire model is limited to the fixed size.

Recurrent language models avoid this limitation, so that a predicted output word can be conditioned on a sequence of arbitrary number of words appeared in the past, not limited by fixed window size. This is an important improvement to language modeling, because a dependency of a word can span to ten or more occurrences of the word in the past. For example, more than ten words in the past needs to be linked to the image location in order to successfully relate the disease mentioned with the image:

‘‘Comparison is made with a study of XX/XX/XXXX. No change not attributable to a difference in technique and bowel content is seen. Findings include 1 liver lesions series 601 image 31.’’

An example of a recurrent neural language model is shown in Fig. 17.5. As shown in the figure, the dependency for an *n*th word prediction can be infinitely long to the occurrences of the previous words. Using recurrent neural networks for NLP, state-of-the-art results were obtained in machine translation [34].

A combination of convolutional neural networks and recurrent neural networks are often used for image caption generation [15,11,35], and this setting was used in [31] to generate image descriptions of chest X-rays. More details of recurrent neural language model can be found in [34,23,3].

## 17.4 MEDICAL LEXICONS

Lexicons are collections of unique concepts accompanied by a preferred “term” (name) and a list of synonyms and derivational forms. Some of the popular medical lexicons used in radiology are UMLS Metathesaurus [28] and RadLex [16].

### 17.4.1 UMLS METATHESAURUS

The Unified Medical Language System (UMLS) of [19,14] integrates and distributes key terminology, classification and coding standards, and associated resources to promote the creation of more effective and inter-operable biomedical information systems and services, including electronic health records. It is a compendium of many controlled vocabularies in the biomedical sciences, created in 1986 and maintained by the National Library of Medicine.

The Metathesaurus [28] forms the base of the UMLS and comprises over 1 million biomedical concepts and 5 million concept names, where all of them are collected from the over 100 incorporated controlled vocabularies and classification systems. Arthritis is defined in the Metathesaurus by the unique alphanumeric code (or Concept Unique Identifier (CUI)) C0003864. It includes synonymous terms for each concept (for example, *stroke* and *cerebrovascular accident*) for CUI C0038454.

Moreover, specific semantic types for each concept and their types and relationships are organized as an ontology. There are 133 semantic types in the UMLS Metathesaurus ontology, such as “T017: anatomical structure”, “T074: medical device”, “T184: sign or symptom”, “T033: finding”, and “T047: disease or syndrome.”

### 17.4.2 RADLEX

RadLex [16] is a unified language to organize and retrieve radiology imaging reports and medical records. While the Metathesaurus has a vast resource of biomedical concepts, RadLex offers more radiology-specific terms, including imaging techniques and radiology devices. It is designed as a comprehensive lexicon for standardizing indexing and retrieval of radiology information resources.

---

## 17.5 PREDICTING PRESENCE OR ABSENCE OF FREQUENT DISEASE TYPES

### 17.5.1 MINING PRESENCE/ABSENCE OF FREQUENT DISEASE TERMS

A large collection of radiology images can be retrieved and labeled with associated disease words, using the lexicons and the pattern matching techniques discussed above. We apply these to the PACS radiology database used in [30] containing about 780,000 imaging examinations of about 60,000 unique patients. We extract images mentioned with (i) radiology words appearing in RadLex; (ii) having “T047: disease or syndrome” semantics in the UMLS Metathesaurus that also appears in the RadLex.

**Table 17.1** Some statistics of images-to-disease assertion/negation label matching

# images		Per image mean/std		# assertions per image		# negations per image	
Total matching	18,291	# assertions mean	1.05	1/image	16,133	1/image	1581
With assertions	16,827	# assertions std	0.23	3/image	81	3/image	0
With negations	1665	# negations std	0.22	4/image	0	4/image	0

Furthermore, we use negation detection algorithm [6,7] to detect presence and absence of the disease terms in the associated image. Some frequently mentioned disease terms detected with assertion are: *cyst*, *disease*, *infection*, *pneumonia*, *bronchiectasis*, *abscess*, with each of them being detected more than 300 times. These terms are similarly mentioned more than 300 times with negation. We ignore the disease terms mentioned less than 10 times in the dataset.

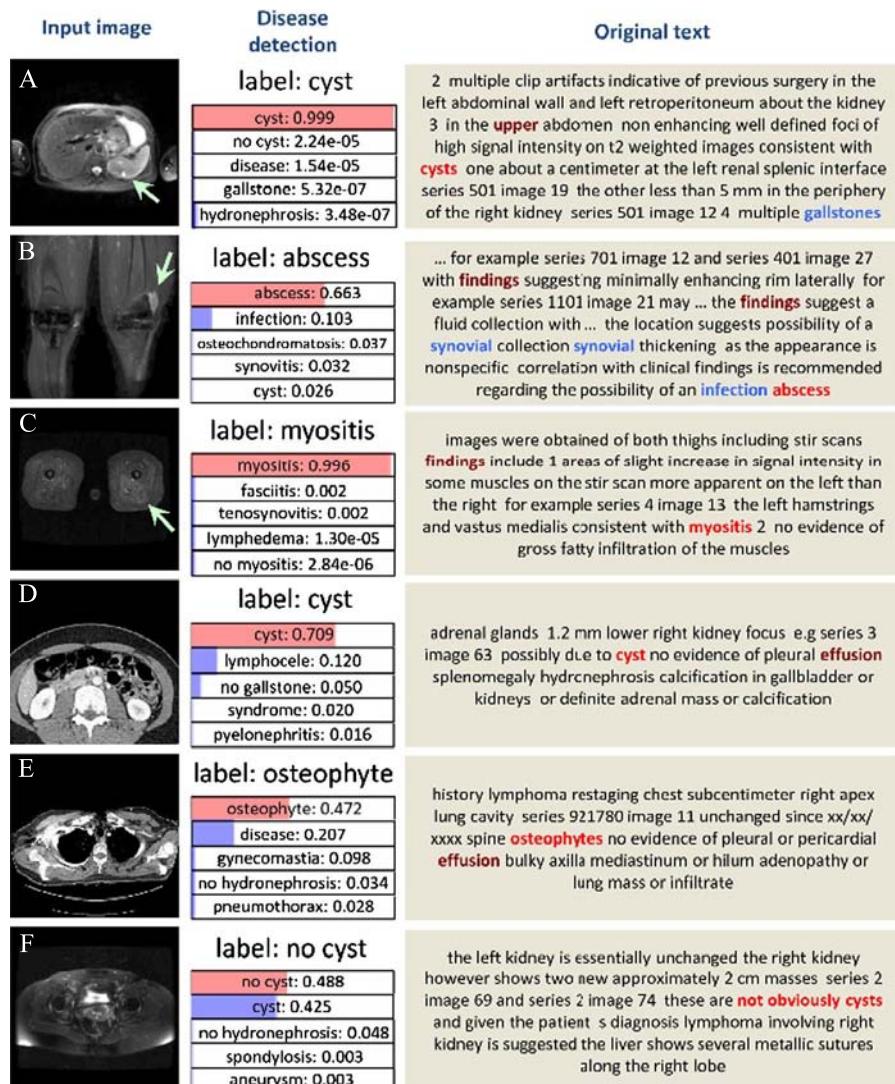
The total number of disease terms with assertion and negation are 77 (59 with assertion and 18 with negation). If more than one disease term is mentioned for an image, we simply assign the terms multiple times for an image. Some statistics on the number of assertion/negation occurrences per image are shown in Table 17.1.

### 17.5.2 PREDICTION RESULT AND DISCUSSION

When we train a deep convolutional neural network model of [32] on the image-to-disease-label dataset we mined from the PACS dataset [30], and achieve the top-1 test accuracy achieved of 0.71, and top-5 accuracy of 0.88. Some examples of test cases where top-1 probability output matches the originally assigned disease labels are shown in Fig. 17.6. It is noticeable that disease words are detected with high probability when there is one disease word per image, but with relatively lower top-1 probability for one disease word and other words within the top-5 probabilities (Fig. 17.6B – “... infection abscess”).

It can be also observed that automatic label assignment to images can sometimes be challenging. In Fig. 17.6D “cyst” is assigned as the correct label based on the original statement “... possibly due to cyst ...”, but it would be unclear whether cyst will be present in the image (and the cyst is not visibly apparent). It applies similarly to Fig. 17.6E where the presence of “osteophyte” is not clear from the referring sentence but is assigned as the correct label (and osteophyte is not visibly apparent on the image). In Fig. 17.6F “no cyst” is labeled and predicted correctly, but it is not obvious what to derive from this prediction that indicates an absence of a disease versus a presence.

Some examples of test cases where top-1 probability does not match the originally assigned labels are shown in Fig. 17.7. Four (A, C, E, and F) of the six examples, however, contain the originally assigned label in the top-5 probability predictions, which is coherent with the relatively high (88%) top-5 prediction accuracy.

**FIGURE 17.6**

Some examples of final outputs for automated image interpretation, where top-1 probability matches the originally assigned label. Specific disease words assigned as label mentioned in the reports are shown in bold red, and disease words predicted with top-5 probability in the reports are shown in bold blue. The probability assigned to the originally assigned label is shown with a red bar, and the other top-5 probabilities are shown with blue bars. Disease region identified in an image is pointed by arrow. Image is adapted from [29].

Here again, Fig. 17.7A is automatically labeled as “cyst”, but the cyst is not clearly visible on the image where the original statement “...too small to definitely characterize cyst ...” supports this. The example of Fig. 17.7B shows a failed case of assertion/negation algorithm, where “cyst” is detected as negated based on the statement “... small cyst”. Nonetheless, true label (“cyst”) is detected as its top-1 probability. For Fig. 17.7C “cyst” is predicted where the true label assigned was “abscess”; however, cyst and abscess are sometimes visibly similar.

It is unclear whether we should expect to find emphysema in the image from the statement such as “...possibly due to emphysema” (and emphysema is not visibly present), similarly to Fig. 17.6D. Therefore, it would be challenging to correctly interpret such statement for label assignment. Fig. 17.7E shows a disease which can be bronchiectasis, though it is not clear from the image. Nonetheless, bronchiectasis is predicted with the second highest probability. Bronchiectasis is visible in Fig. 17.7F, and it is predicted with second highest probability, too.

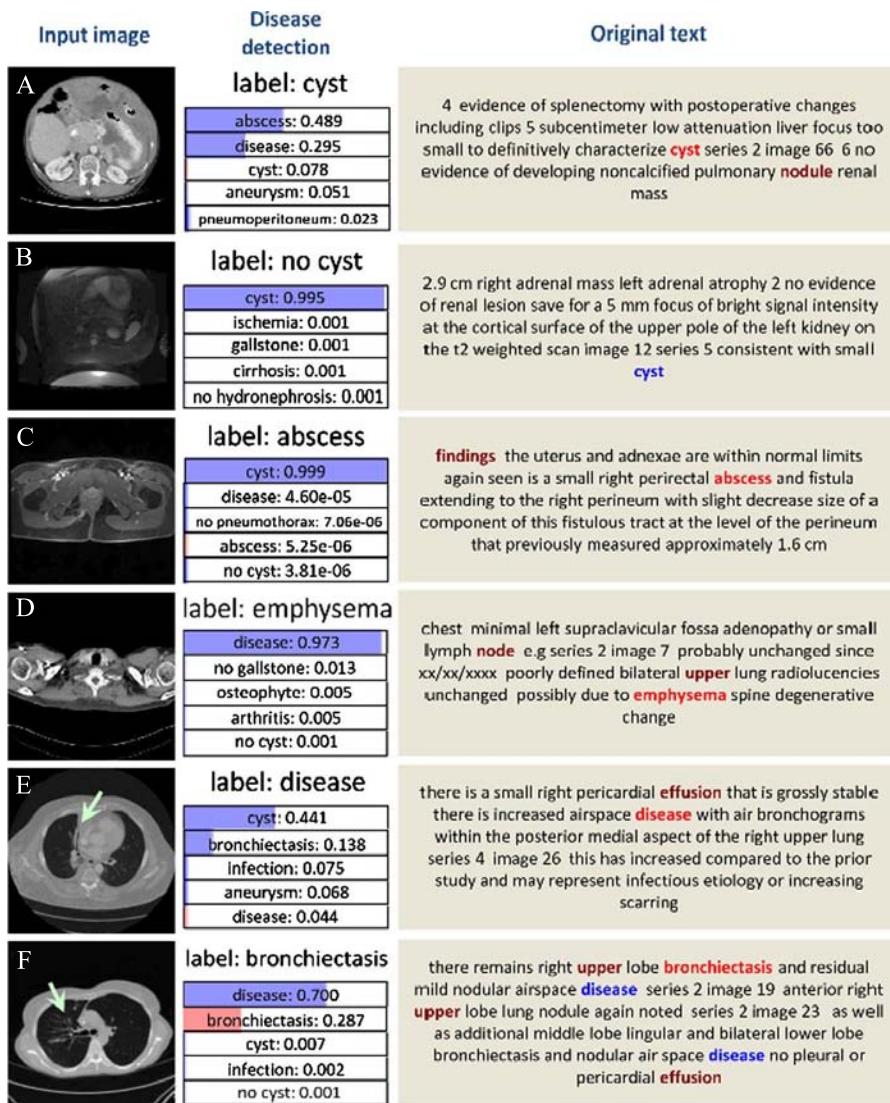
#### 17.5.2.1 Discussion

Automated mining of disease terms enables us to predict a disease from a large amount of images with promising result. However, matching the images to more specific disease words and assigning labels is not always straightforward as was shown for the examples in Fig. 17.7. A big part of challenge is on developing a better natural language processing algorithms to (i) infer better image labels, to (ii) deal with uncertainties in radiology text, and to (iii) better relate words of different sentences for more composite labeling.

Some NLP challenges in mining more images and assigning better image labels are:

- Stemming (e.g., kidneys → kidney, vertebrae → vertebra)
- Anaphora resolution [17] (e.g., findings are mentioned in a different sentence than the sentence referring to the image)
- Part-of-speech (POS) tagging [10] (e.g., whether “spleen” is a noun, or “unremarkable” is an adjective, etc.)
- Quantification of hedging statements [27,37]
  - “Too small to characterize” → present with 40% confidence
  - “possibly degenerative” → present with 50% confidence
- Coreference resolution [1] (identify words for size, location, etc., and relate these with the words they are describing)

Some of the above mentioned challenges are specific to NLP in radiology text (e.g., stemming, POS tagging are regarded not challenging in general NLP), though the others are more generic NLP challenges. Also, comprehensive analysis of hospital discharge summaries, progress notes, and patient histories might address the need to obtain more specific information relating to an image even when the original image descriptions are not very specific.

**FIGURE 17.7**

Some examples of final outputs for automated image interpretation where top-1 probability does not match the originally assigned label. One of the top-5 probabilities match the originally assigned labels in the examples of images A, C, D, and F. None of the top-5 probabilities match the originally assigned labels in the examples of image B and D. However, label assignment for image B is incorrect, as a failed case of assertion/negation detection algorithm used. Nonetheless, the CNN predicted “true” label correctly (“cyst”). Image is adapted from [29].

---

## 17.6 CONCLUSION

It has been unclear how to extend the significant success in image classification using deep convolutional neural networks from computer vision to medical imaging. What are the clinically relevant image labels to be defined, how to annotate the huge amount of medical images required by deep learning models, and to what extent and scale the deep CNN architecture is generalizable in the medical image analysis are open questions.

In this chapter, we present an approach to mine and label images from a hospital PACS database using natural language processing. Natural language processing enables us to conduct large-scale medical image analysis, which has been very challenging with manual data collection and annotation approaches. Basic natural language processing methods as well as more recent deep-learning-based approaches for mining large radiology reports are introduced.

We also demonstrated to mine and match frequent disease types using disease ontology and semantics from a large PACS database, and demonstrate prediction of the presence/absence of disease from a radiology image with probability outputs. Exploring effective deep learning models on this database opens new ways to parse and understand large-scale radiology image informatics.

---

## ACKNOWLEDGEMENTS

This work was supported in part by the Intramural Research Program of the National Institutes of Health Clinical Center, and in part by a grant from the KRIBB Research Initiative Program (Korean Biomedical Scientist Fellowship Program), Korea Research Institute of Bioscience and Biotechnology, Republic of Korea. This study utilized the high-performance computational capabilities of the Biowulf Linux cluster at the National Institutes of Health, Bethesda, MD (<http://biowulf.nih.gov>). We thank NVIDIA for the K40 GPU donation.

---

## REFERENCES

1. E. Apostolova, D. Demner-Fushman, Towards automatic image region annotation: image region textual coreference resolution, in: Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers, Association for Computational Linguistics, 2009, pp. 41–44.
2. A. Aronson, O. Bodenreider, H. Chang, S. Humphrey, J. Mork, S. Nelson, T. Rindflesch, W. Wilbur, I. Initiative, et al., A Report to the Board of Scientific Counselors of the Lister Hill National Center for Biomedical Communications, 1999.
3. D. Bahdanau, K. Cho, Y. Bengio, Neural machine translation by jointly learning to align and translate, arXiv:1409.0473, 2014.
4. S. Bird, NLTK: the natural language toolkit, in: Proceedings of the COLING/ACL on Interactive Presentation Sessions, Association for Computational Linguistics, 2006, pp. 69–72.

5. D.M. Blei, A.Y. Ng, M.I. Jordan, Latent Dirichlet allocation, *J. Mach. Learn. Res.* 3 (2003) 993–1022.
6. W.W. Chapman, W. Bridewell, P. Hanbury, G.F. Cooper, B.G. Buchanan, A simple algorithm for identifying negated findings and diseases in discharge summaries, *J. Biomed. Inform.* 34 (5) (2001) 301–310.
7. W.W. Chapman, D. Hilert, S. Velupillai, M. Kvist, M. Skeppstedt, B.E. Chapman, M. Conway, M. Tharp, D.L. Mowery, L. Deleger, Extending the NegEx lexicon for multiple languages, *Stud. Health Technol. Inform.* 192 (2013) 677.
8. M.-C. De Marneffe, A.N. Rafferty, C.D. Manning, Finding contradictions in text, in: ACL, vol. 8, 2008, pp. 1039–1047.
9. J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, ImageNet: a large-scale hierarchical image database, in: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2009, IEEE, 2009, pp. 248–255.
10. S.J. DeRose, Grammatical category disambiguation by statistical optimization, *Comput. Linguist.* 14 (1) (1988) 31–39.
11. J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, T. Darrell, Long-term recurrent convolutional networks for visual recognition and description, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 2625–2634.
12. O. Ferrández, B.R. South, S. Shen, F.J. Friedlin, M.H. Samore, S.M. Meystre, Bob, a best-of-breed automated text de-identification system for VHA clinical documents, *J. Am. Med. Inform. Assoc.* 20 (1) (2013) 77–83.
13. T. Hofmann, Probabilistic latent semantic indexing, in: Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, 1999, pp. 50–57.
14. B.L. Humphreys, D.A. Lindberg, H.M. Schoolman, G.O. Barnett, The unified medical language system an informatics research collaboration, *J. Am. Med. Inform. Assoc.* 5 (1) (1998) 1–11.
15. A. Karpathy, L. Fei-Fei, Deep visual-semantic alignments for generating image descriptions, in: IEEE Conference on Computer Vision and Pattern Recognition, CVPR, June 2015.
16. C.P. Langlotz, RadLex: a new method for indexing online educational materials 1, *RadioGraphics* 26 (6) (2006) 1595–1597.
17. S. Lappin, H.J. Leass, An algorithm for pronominal anaphora resolution, *Comput. Linguist.* 20 (4) (1994) 535–561.
18. D.D. Lee, H.S. Seung, Learning the parts of objects by non-negative matrix factorization, *Nature* 401 (6755) (1999) 788–791.
19. D.A. Lindberg, B.L. Humphreys, A.T. McCray, The unified medical language system, *Methods Inf. Med.* 32 (4) (1993) 281–291.
20. J.H. Martin, D. Jurafsky, *Speech and Language Processing*, International Edition, 2000, p. 710.
21. S. Mehrabi, A. Krishnan, S. Sohn, A.M. Roch, H. Schmidt, J. Kesterson, C. Beesley, P. Dexter, C.M. Schmidt, H. Liu, et al., DEEPEN: a negation detection system for clinical text incorporating dependency relation into NegEx, *J. Biomed. Inform.* 54 (2015) 213–219.
22. T. Mikolov, K. Chen, G. Corrado, J. Dean, Efficient estimation of word representations in vector space, arXiv:1301.3781, 2013.
23. T. Mikolov, M. Karafiat, L. Burget, J. Cernocky, S. Khudanpur, Recurrent neural network based language model, in: INTERSPEECH, vol. 2, 2010, p. 3.

24. T. Mikolov, I. Sutskever, K. Chen, G.S. Corrado, J. Dean, Distributed representations of words and phrases and their compositionality, in: Advances in Neural Information Processing Systems, 2013, pp. 3111–3119.
25. T. Mikolov, W.-t. Yih, G. Zweig, Linguistic regularities in continuous space word representations, in: HLT-NAACL, 2013, pp. 746–751.
26. I. Neamatullah, M.M. Douglass, H.L. Li-wei, A. Reisner, M. Villarroel, W.J. Long, P. Szolovits, G.B. Moody, R.G. Mark, G.D. Clifford, Automated de-identification of free-text medical records, *BMC Med. Inform. Decis. Mak.* 8 (1) (2008) 1.
27. F. Salager-Meyer, Hedges and textual communicative function in medical English written discourse, *Engl. Specif. Purp.* 13 (2) (1994) 149–170.
28. P.L. Schuyler, W.T. Hole, M.S. Tuttle, D.D. Sherertz, The UMLS metathesaurus: representing different views of biomedical concepts, *Bull. Med. Libr. Assoc.* 81 (2) (1993) 217.
29. H.-C. Shin, L. Lu, L. Kim, A. Seff, J. Yao, R.M. Summers, Interleaved text/image deep mining on a large-scale radiology database for automated image interpretation, *J. Mach. Learn. Res.* 17 (107) (2016) 1–31.
30. H.-C. Shin, L. Lu, L. Kim, A. Seff, J. Yao, R.M. Summers, Interleaved text/image deep mining on a very large-scale radiology database, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 1090–1099.
31. H.-C. Shin, K. Roberts, L. Lu, D. Demner-Fushman, J. Yao, R.M. Summers, Learning to read chest X-rays: recurrent neural cascade model for automated image annotation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016.
32. K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, arXiv:1409.1556, 2014.
33. K. Stevens, P. Kegelmeyer, D. Andrzejewski, D. Buttler, Exploring topic coherence over many models and many topics, in: Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, Association for Computational Linguistics, 2012, pp. 952–961.
34. I. Sutskever, O. Vinyals, Q.V. Le, Sequence to sequence learning with neural networks, in: Advances in Neural Information Processing Systems, 2014, pp. 3104–3112.
35. O. Vinyals, A. Toshev, S. Bengio, D. Erhan, Show and tell: a neural image caption generator, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 3156–3164.
36. H.M. Wallach, Topic modeling: beyond bag-of-words, in: Proceedings of the 23rd International Conference on Machine Learning, ACM, 2006, pp. 977–984.
37. A.S. Wu, B.H. Do, J. Kim, D.L. Rubin, Evaluation of negation and uncertainty detection and its impact on precision and recall in search, *J. Digit. Imaging* 24 (2) (2011) 234–242.
38. P. Xie, E.P. Xing, Integrating document clustering and topic modeling, arXiv:1309.6874, 2013.
39. R. Yeniterzi, J. Aberdeen, S. Bayer, B. Wellner, L. Hirschman, B. Malin, Effects of personal identifier resynthesis on clinical text de-identification, *J. Am. Med. Inform. Assoc.* 17 (2) (2010) 159–168.

This page intentionally left blank

# Index

## A

Action recognition, 38, 41  
Activations, 9, 13, 29, 31, 33, 45, 59, 226, 253, 308, 327  
Active shape model (ASM), 56, 63, 216, 219  
ADNI (Alzheimer's Disease Neuroimaging Initiative), 260, 360, 372, 377  
ADNI dataset, 248, 259, 262  
Agent, 65–70, 75, 77  
AlexNet, 26, 30, 32, 33, 36, 41, 96, 126, 330  
AlexNet model, 27, 32  
Algorithmic strategies, 274, 286, 290, 292, 293  
Alzheimer's disease (AD), 342, 360, 368, 370, 372  
Anatomical structures, 57, 64, 199, 224, 249, 414  
Anatomies, 57, 71, 83, 223, 233, 237, 382  
Anatomy detection, 56, 71, 100  
Answer, 43  
APOE (Apolipoprotein E), 360, 362, 367, 372, 374  
Architecture of randomized deep network, 356  
Artificial agent, 57, 64–66, 78  
Atlas images, 199, 206  
Atlas patches, 208  
Attributes, 43  
Auto-encoder (AE), 12, 201, 250, 348  
    basic, 202  
    single, 252  
    stacked, 12, 200, 247, 250, 252  
Automated system, 301, 315  
Automatic segmentation, 238  
Auxiliary tasks, 43, 140

## B

Background, 92, 139, 143, 147, 164, 197, 199, 205, 206, 281, 330, 344, 350  
    voxels, 205  
Backpropagation algorithm, 6, 11, 14, 30, 161  
Bag-of-visual-words (BoVW), 302, 305, 310

Bag-of-words (BoW), 305

Baseline, 259, 335, 344, 359, 360, 362, 364, 370, 373

Baseline markers, 361, 362, 370

Batch normalization (BN), 21, 22, 31

Binary masks, 187, 189

Biomedical image analysis tasks, 157, 165

Blocks, 354–359

Body sections, 85, 91

Body-part recognition, 84, 86, 87, 91, 92, 95

Boundaries, 64, 115, 187, 232

BoVW model, 305

Brain images, 371

Brain MR images, 252, 255, 265

Brain MRI images, 238, 240

Brain regions, 230, 354

Breast cancer, 139, 322

    histology images, 157, 166

## C

C++, 22, 45, 169, 285

Cancer, 301, 322

Cardiac histopathology, 180, 182, 184, 191

Cardiac histopathology images, 180, 182, 186, 191

Cardiovascular disease (CVD), 106

Carotid artery, 107, 112, 114, 125  
    common, 108, 109, 113, 124

Carotid bulb, 108, 112–115, 121, 124

Carotid intima–media thickness (CIMT), 106, 124

Cell detection, 166

Cells

    complex, 28, 110

    simple, 28

Central processing units (CPU), 11, 72, 109

Centroid distances, 228, 230, 233, 239

Centroids, 112, 115, 140, 229, 230, 287

Cerebral microbleed detection, 143

Cerebral microbleeds (CMBs), 134, 135, 143–147, 149

Cerebral-spinal fluid (CSF), 259, 342, 343

Chest radiograph, 300, 302

- CIMT measurements, 106, 123, 128  
 CIMT video interpretation, 106, 127  
 CIMT videos, 108–110, 117, 127  
 Class membership, 160, 187, 188, 343  
 Classes  
   body section, 92  
   non-informative, 91  
 Classification accuracies, 33, 93, 331, 343, 370  
 Classification of breast lesions  
   benign, 323, 324, 326, 330–332, 334  
   malignant, 323, 324, 326, 330–332, 334  
 Classification performance, 33, 205, 317  
 Classifier, 56, 58, 61, 64, 86, 87, 90, 91, 97, 148, 158, 166, 185, 188, 191, 310, 315  
   main, 63, 71, 73  
 Clinical dementia rating sum of boxes (CDR-SB), 361, 364, 366, 367, 372, 373  
 Clinical trials, 344, 368, 373  
 CMBs  
   detection, 135, 143, 146  
   true, 144, 148  
 CNN  
   2D, 143, 148, 150  
   3D, 135, 144, 226  
 CNN flavors, 34  
 CNN model, 38, 165, 280, 293, 307, 327  
 CNN regression model, 286, 292  
 CNN structure, 88, 92, 157  
 CNN-based methods, 234, 238–240  
 CNNs  
   local patch-based, 94  
   standard, 86, 93, 99  
   trained, 113  
 Co-occurrence of local anisotropic gradient orientations (CoLlAGE), 186  
 Coarse retrieval model, 136  
 Comparison of deep learning, 186  
 Computational complexity, 110, 226, 305  
 Computational limitations, 78  
 Computed tomography (CT), 84, 106, 272, 382, 409  
 Computer aided diagnosis (CAD), 83, 134, 272, 322  
 Computer vision, 26, 37, 86, 106, 109, 180, 200, 226, 248, 253, 273, 302, 324, 343, 352  
 Computer vision problems, 26, 230, 239  
 Computer vision tasks, 30, 191, 273  
 Computer-aided diagnosis, 134, 149, 322  
 Concatenation deep network (CDN), 389  
 Conditional random field (CRF), 37, 325  
 Confidence maps, 113, 116  
 Connection weights, 5, 8, 11, 14, 21, 203  
 Constrained ROI localization, 113, 118  
 Convolution kernel, 74, 137, 144, 158, 226, 227  
 Convolution layer, 8, 27, 109  
 Convolutional filters, 28, 87, 100  
 Convolutional layers, 8, 29, 33, 86, 109, 110, 117, 126, 137, 158, 163, 167, 226, 227, 307, 326  
 Convolutional networks, 33, 248  
 Convolutional neural network architecture, 226  
 Convolutional neural network (CNN), 8, 26, 27, 34, 35, 40, 85, 109, 137, 157, 166, 273, 280, 411  
 Convolutional SAE, 247, 256, 257, 259, 263, 264  
 Convolutional SAE network (CSAE), 253, 263  
 Convolutions, 8, 27, 28, 36, 37, 138, 226, 254, 281  
 Coupled sparse representation (CSR), 382, 389, 396  
 Cranio-caudal (CC), 322  
 Cross-correlation (CC), 289, 290, 329, 331, 332  
 Cross-modal medical image synthesis, 393, 401  
 Cross-modal nearest neighbor search, 382, 384, 392, 396
- D**
- Data augmentation, 31, 113, 125, 331  
 Dataset, 71, 74, 95, 140, 146, 219, 228, 286, 309, 326, 328–330, 388  
 DDSM, 324, 329–332, 334  
 InBreast, 324, 329–334  
 large, 146, 353, 407

- Decliners  
 strong, 370, 371  
 weak, 342, 363, 367, 372
- Deconvolution, 37
- Deep architecture, 12, 74, 126, 166, 205, 250, 343, 344, 353
- Deep belief network (DBN), 15, 17
- Deep Boltzmann machine (DBM), 15, 18
- Deep cascaded networks, 136
- Deep convolutional neural networks, 69, 135, 180, 181
- Deep learning, 8, 57, 65, 85, 86, 134, 157, 180–182, 184, 191, 200, 248, 250, 262, 324, 325, 346, 347
- approach, 182, 185, 191, 224, 248, 302, 315
- architecture, 109, 209, 265
- for medical image, 87, 157, 223, 239, 240
- for segmentation, 188, 191, 225
- methods, 22, 84, 200, 207, 223, 240, 303, 325, 353, 406
- models, 26, 180, 182, 185, 186, 188, 250, 252, 324, 334, 350, 419
- network, 187, 253, 255
- software for, 45
- tools for, 22
- unsupervised, 247, 250
- Deep learning features, 200, 207, 211, 248, 324
- Deep models, 11, 20, 22, 247, 250
- Deep networks, 11, 14, 20, 27, 29, 35, 73, 88, 203, 227, 306, 344, 352, 358
- location-sensitive, 382, 383, 385, 401
- very, 30, 31, 33
- Deep neural networks, 11, 12, 18, 20, 67, 70, 135, 224–226, 303
- Deep Q network (DQN), 69
- Deep voting, 156, 163, 164
- Deep voting model, 157, 159, 163, 165
- Deep-learned features, 199, 206, 212
- Deformable model, 209, 210, 216
- Descriptor, 273, 304
- Detection, 33, 34, 57, 58, 84, 86, 100, 300
- accurate, 134, 143
- computer-aided, 106
- lymph node, 325
- microbleed, 134, 240
- negation, 409
- Detection accuracy, 34, 76, 117, 142, 165
- Detection network, 86
- Detection of emphysema, 302, 417
- Detection time, 136, 142
- Dice ratios, 212, 260
- Dice scores, 230, 233
- Diffeomorphic demons, 257, 259, 264, 265
- Digitally reconstructed radiograph (DRR), 272
- Disease, 181, 310, 311, 345, 351–353, 359, 361, 368, 410, 415, 417, 419
- Disease markers, 342, 358, 362, 363, 370, 373
- Disease progression, 345, 358, 368, 370
- Dropout, 20, 31, 347, 358
- Dropout networks, 349, 353, 354
- DSC (direct splatting correlation), 293
- DV-1 (deep voting with no stride), 163–165
- DV-3 (deep voting with stride 3), 163–165
- DxConv, 361, 364, 365, 367, 370, 372, 373
- E**
- Edge-hypersampling, 183, 187
- Edges, 28, 185–187, 189, 190, 279, 286, 300, 305, 312
- Effect size, 345, 361, 367
- Effectiveness, 121, 135, 143, 199–201, 207, 212, 216, 398
- Efficacy, 141, 142, 147, 150, 343, 344, 372, 373
- End-diastolic ultrasonographic frames (EUFs), 106, 108, 110–113, 117, 118, 123–125, 127
- Enrichment, 343, 359, 364, 373
- Ensemble learning, 224, 239, 351, 352
- Errors, 75, 228, 231, 233, 237
- boundary, 232, 237, 239
- labeling, 237
- localization, 12, 120, 202, 348
- segmentation, 74, 231, 233, 237
- Evaluation, 71, 74, 75, 77, 212, 216
- Experience replay, 70
- Experimental results, 134, 135, 150, 164, 169, 259, 260, 263, 310, 330
- Experiments, 71, 73, 117, 163, 186–188, 211, 228, 258, 260, 283, 285, 287, 309, 360, 388

- Experts, 122  
 Extracting the image information, 409
- F**  
 False negative (FN), 141, 142, 170, 315  
 False positive (FP), 93, 135, 137, 140–142,  
     147, 150, 163, 170, 237, 315, 325  
 Fast scanning, 164  
 Feature extraction, 56, 277  
 Feature maps, 9, 227  
 Feature representations, 200, 214, 225, 248,  
     262, 305, 411  
 abstract, 224  
 intrinsic, 248–250, 255  
 latent, 251, 253, 255, 260, 265  
 low-dimensional, 249, 256, 259  
 Feature selection, 246, 247, 258, 265, 309,  
     313, 317  
 Feed-forward neural networks, 4, 6, 412  
 Feldman, 191  
 FH (family history), 360, 361, 370, 372, 374  
 Fine discrimination model, 134, 136, 139  
 Fine-tuning, 14, 32, 331  
 Fine-tuning process, 328, 331  
 Frame selection, 108, 110, 118, 126, 128  
 Fully connected hidden layers, 109  
 Fully connected neural networks, 9, 59  
 Fully convolutional network (FCN), 35–37,  
     44, 127, 136, 137  
 2D, 145  
 3D, 144, 145, 147, 148, 150  
 Fully-connected layers, 29, 33, 168, 229,  
     280, 286  
 Function  
     activation, 4, 10, 59, 158, 168, 227, 346  
     network response, 59  
     optimal action-value, 66, 69  
 Fundamentals of natural language  
     processing, 407  
 Fusion process, 169
- G**  
 Gaussian mixture model (GMM), 248, 249  
 Gaussian smoothing, 119, 125  
 Generative models  
     deep, 14  
 GIST, 310, 311
- GLCM (gray-level co-occurrence matrix),  
     303, 304, 310  
 Gradient correlation (GC), 284, 289  
 Graphics processing units (GPUs), 11, 22,  
     27, 106, 110, 230, 285, 324  
 Gray matter (GM), 259  
 Ground truth, 59, 77, 117, 140, 141, 146,  
     159, 211, 234, 246, 247, 274, 284,  
     287, 397  
 Ground-truth regions, 163
- H**  
 HAMMER, 259, 263–265  
 Handcrafted features, 137, 139, 140, 143,  
     157, 182, 191, 199–201, 207, 209,  
     214, 246, 248, 256, 273, 323, 324  
 Heart failure, 181, 191, 300  
 Hidden layers, 5, 6, 12, 28, 92, 94, 202, 227,  
     250, 286, 385  
     dimension of, 203  
     first, 14, 207, 385  
     second, 14, 205, 387  
 Hidden nodes, 87, 96, 203, 251, 259, 264  
 High-power fields (HPFs), 140, 142  
 Hippocampal volume, 361, 362, 364, 368,  
     370, 373  
 Hippocampus, 224, 260, 263, 265  
 Histogram of oriented gradients (HOG), 26,  
     33, 85, 199–201, 214, 219, 303  
 Histology images, 139, 141  
 Hyperparameters, 358
- I**  
 ICPR MITOSIS dataset, 140  
 Image analysis, 106, 180, 227  
 Image classification, 22, 26, 32, 34, 87, 93,  
     95, 135, 334, 419  
 Image classification tasks, 32, 35, 85, 90, 93,  
     99, 100  
 Image patches, 34, 35, 45, 93, 113, 115, 156,  
     166, 169, 186, 202, 227, 246, 247  
 Image registration, 239, 246, 249, 256, 259,  
     265  
     methods, 246, 257  
 Image representation, 42, 157, 303, 305, 306  
 Image representation, schemes, 303  
 Image segmentation, 180

Image-based tool for counting nuclei (ITCN), 171  
 ImageNet, 27, 303, 307, 331, 332, 334, 406  
 ImageNet classification, 26, 307  
 ImageNet data, 303, 307  
 Images  
   fluoroscopic, 272, 275, 283  
   hematoxylin or eosin grayscale, 186  
   original, 88, 138, 188  
   radiology, 406, 407, 412, 414, 419  
 Implementation, 73, 93, 163, 182, 183, 305, 324, 368, 374  
 Improvement, 34, 72, 94, 100, 140, 260, 312, 313, 332, 334, 344, 373  
 Inclusion criteria, 359  
 Information  
   contextual, 143, 184, 249, 323  
   topological, 166, 172  
 Input channels, 280  
 Input data, 13, 84, 202, 249, 251, 324, 348, 352  
 Input feature maps, 8, 158  
 Input features, 13, 19, 229, 277  
 Input image, 9, 36, 42, 84, 109, 137, 158, 182, 226, 304, 307, 330, 334  
 Input layer, 5, 11, 158, 202, 229, 250, 358  
 Input patches, 10, 118, 228, 251, 386  
 Input training patches, 202, 253  
 Input vector, 13, 19, 158, 202  
 Intelligence, 65  
 Intensity, 143, 144, 147, 148, 185, 207, 209, 214, 216, 219, 225, 272, 279, 391  
 Intensity features, 186, 216, 384  
 Intensity patch, 199, 208, 219  
 Intensity transformation, 382  
 Intensity values, 94, 125, 246, 383, 389, 392, 393, 397  
 Intensity-based methods, 272, 285, 288  
 Intervention, 342, 344  
 Invariant, 85, 203, 254, 346  
 Iterations, 8, 258, 286, 292, 396  
 Iterative radial voting (IRV), 171

**K**

K-nearest neighbor for pose estimation, 273  
 Krizhevsky network, 307

**L**  
 Label fusion, 93  
 Labels  
   assigned, 415  
   correct, 88, 315, 415  
   true, 59, 230, 239  
 Landmark detection, 57, 67, 71, 74, 78  
   accurate, 43, 74  
   anatomical, 83  
   robust, 75  
 Landmarks, 68, 74, 77, 206, 288, 289, 386  
 Language, 42, 183, 411, 413  
 Latent Dirichlet allocation (LDA), 410  
 Layer-wise learning, 14, 252  
 Layers, 5, 6, 8, 13, 15, 17, 18, 27, 31–34, 36, 37, 45, 117, 118, 158, 160, 161, 202, 253, 312, 313  
   connections between, 13  
   final, 42, 227, 306, 327  
   first, 17, 28, 158  
   last, 158, 166, 170, 347  
   neighboring, 5, 18, 202  
   penultimate, 158, 312  
   second, 17, 229  
   single, 4, 38, 41, 247, 313  
   sub-sampling, 96, 326  
 Learned feature representations, 86, 203, 205, 248, 256, 257, 259, 262–264  
 Learned features, 36, 38, 148, 214, 219, 247, 255  
 Learning models, 26, 71, 249, 250, 343, 354, 355  
 Learning problems, 38, 56, 344, 352, 354  
 Leave-1-patient-out cross-validation, 118–122  
 Left consolidation (LCN), 310  
 Left cuneus, 234, 237  
 Left pleural effusion (LPE), 310, 315  
 Lesions, 143, 321–325, 332, 334  
   classification, 305, 322, 325  
   detection, 322, 323, 325  
   segmentation, 322, 323, 325  
 Likelihood map, 209, 210  
 Likelihood ratios, 311  
 Local binary patterns (LBP), 186, 199–201, 214, 219, 302  
 Local image residual (LIRs), 286–288, 290, 292, 293

- Local information, 86, 88, 100  
 Local maxima, 112, 272, 290, 293  
 Local patches, 86, 88, 95, 207  
   discriminative, 88, 94  
   extracted, 95  
 Local regions, 29, 85, 100  
   discriminative, 85, 86, 95, 100  
   non-informative, 87  
 Localization, 34, 44, 106, 125, 157, 302, 317  
 Locations, 34, 56, 58, 111–115, 125, 145,  
   239, 277, 323, 417  
 Logistic regression (LR), 85, 87, 225  
 Logistic sigmoid function, 4, 6, 20, 251  
 Longitudinal change, 359  
 LONI dataset, 248, 262  
 Loss function, 30, 87–90, 110, 157, 161,  
   168, 389  
 LSDN (location-sensitive deep network),  
   382–385, 400  
 LSDN-1, 389  
 LSDN-2, 389  
 LSDN-small, 389  
 LSTM (long short term memory), 41–43  
 Lumen–intima and media–adventitia  
   interfaces, 107, 115–117, 123  
 Lung diseases, 301, 302
- M**
- Machine learning, 4, 56, 58, 64, 200, 248,  
   273, 324, 343, 406  
 Machine learning methods, 180, 342, 343,  
   409  
 Madabhushi, 191  
 Mammograms, 226, 321, 323, 325, 328, 330,  
   334  
 Mammography view, 322, 324, 330  
 Manual ground truth annotations, 183  
 Marginal space, 62, 63, 276  
 Marginal space deep learning (MSDL), 56,  
   61, 77  
 Marginal space learning (MSL), 56, 61, 67  
 Marginal space regression (MSR), 277,  
   286–288, 292  
 Markov decision process (MDP), 65, 67  
 Mass, 325, 330–332, 334  
 Matlab, 165, 169  
 Max pooling, 182, 229, 254, 259, 264, 330,  
   331
- Maximally stable extremal region (MSER),  
   156  
 Media–adventitia interface, 115, 116, 122  
 Medical image analysis (MIA), 16, 74, 83,  
   85, 106, 157, 180, 181, 200, 226,  
   239, 315, 324, 325, 419  
 Medical image applications, 87, 246, 248  
 Medical images, 83, 84, 86, 87, 134, 144,  
   149, 224, 239, 246, 250, 253, 299,  
   335, 387, 406, 419  
 Medio-lateral oblique (MLO), 322, 331  
 Methodology, 58, 87, 158, 166, 324, 326  
 MHD (modified Hausdorff distance), 188  
 Micro-calcifications, 322, 324–326,  
   330–332, 334  
 Microscopy images, 156, 166, 172  
 Mild cognitively impaired subjects (MCIs),  
   342, 360–362, 364, 370–372  
   late, 359–362, 370, 371  
 Mimics, 137, 139, 140, 143–145  
 Mini mental state examination (MMSE),  
   359, 361, 364, 367, 372, 373  
 Mini-batches, 21, 30, 69, 110, 230  
 Minimum variance unbiased (MVUB), 351,  
   355, 358, 359  
 Mitoses, 134, 139, 157  
 Mitosis detection, 134–137, 139, 140, 166,  
   182, 225, 325  
   automated, 141  
 MKL (multi-kernel support vector machine),  
   361  
 MKLm (MKL markers), 361, 362, 368, 370,  
   373  
 Modalities, 38, 43, 272, 284, 323, 342, 355,  
   360, 362, 371, 382, 388, 392, 393,  
   403  
 Modality propagation (MP), 383, 389, 400  
 Model selection and training parameters, 71  
 Model's outputs, 161, 162, 169, 358  
 Montreal cognitive assessment (MOCA),  
   361, 362, 364, 373  
 Morphological signature, 249, 250, 256, 263  
 MR brain images, 248, 255  
 MR images, 74, 197, 198, 211, 214, 247,  
   259, 263  
 MR (magnetic resonance), 84, 135, 197, 409  
 MR prostate images, 200, 207, 219  
 MR volumes, 135, 143, 144

MRI images, 305, 360, 370  
MRI (magnetic resonance images), 224, 228, 272, 353, 355, 360, 371, 382, 409  
MRI scans, 388, 397  
MSDL framework, 71  
MSER (maximally stable extremal region), 156  
MTREproj (mean target registration error in the projection direction), 285, 290, 292  
Multi-atlas, 199–201, 206, 207, 224, 228  
Multi-instance learning (MIL), 86, 93–95, 97–99  
Multi-layer perceptron (MLP), 5, 33, 227, 273, 385  
Multi-modal baseline rDAM, 361, 363–365  
Multi-task learning, 43  
Mutual information maximization, 384, 393, 401  
Mutual information (MI), 231, 272, 289, 382, 384, 392, 393, 397  
Myocytes, 180–183, 185–187

**N**

Natural language processing (NLP), 84, 343, 352, 406, 407, 411, 417, 419  
Neighbors, 93, 166, 169, 391, 392  
NERS (non-overlapping extremal regions selection), 163, 164, 166, 171  
Network, 9, 13, 20, 26, 28, 32, 35, 43, 44, 57, 59, 84, 140, 144, 286, 354, 355  
cascaded, 144  
decoder, 247, 248, 250  
deep belief, 26  
simplified, 388  
smaller, 349, 350, 390  
Network architecture, 21, 76, 138, 146, 148, 163, 183, 230, 307  
learning, 6  
Network parameters learning, 6  
Network representation, 308  
Network structure, 253, 280, 347  
Neural language models, 411  
Neural network model, 45  
deep convolutional, 415  
Neural networks, 4, 12, 22, 29, 30, 34, 59, 67, 70, 76, 148, 160, 225–227, 285, 345, 384

deep max-pooling convolutional, 137  
feed-forward, 6, 411  
multi-layer, 5, 8, 14, 87, 346  
single-layer, 4, 346  
sparse adaptive deep, 59  
two-layer, 11  
Neurons, 4, 11, 30, 139, 182, 227  
Non-informative patches, 92  
Nonlinear transformation, 12, 158, 224, 225, 250  
Nonlinearities, 29  
Number of hidden units, 5, 11–13

**O**

Object detection, 26, 34, 35, 38, 39, 43, 71, 73, 134, 155, 157, 200  
Object recognition, 84, 140, 302, 353, 371  
Optimal enrichment criterion, 345, 350  
Optimization, 93, 230, 290, 395  
Optimization problem, 161, 167, 387, 394, 396  
Optimizer, 272, 289, 290  
Orientations, 56, 58, 61, 72, 124, 277, 278, 283, 304, 312  
Outcome measure, 344, 361  
Output layer, 5, 12, 14, 91, 144, 160, 202, 204, 205, 227, 229, 250, 280, 286, 346, 386, 387  
Outputs, 9, 13, 30, 36, 41, 44, 87, 92, 123, 136, 158, 161, 167, 168, 226, 357, 358  
Overfitting, 9, 20, 21, 59, 61, 94, 97, 230

**P**

Paired t-test, 122, 214, 219, 262  
Parameter space, 56, 61, 276, 286, 347  
Parameter space partitioning (PSP), 276, 286–288, 292  
Parameters, 140, 161, 211, 231, 277  
large number of, 29, 31, 33, 148  
learned, 349, 350  
model's, 30, 159, 161, 162, 168  
out-of-plane rotation, 277  
out-of-plane translation, 277  
tuned, 118  
Patch binarization, 125, 128  
Patch representation, 182, 187

- Patches, 35, 58, 69, 91, 97, 107, 112, 113, 115, 121, 122, 125, 140, 144, 156, 187, 229  
 local image, 35, 166  
 sampled image, 255  
 selected image, 164, 255  
 training image, 159, 252
- Pathologies, 182, 301–303, 310, 313–315, 317, 342  
 digital, 180, 187  
 examined, 310
- Pattern matching, 407, 409
- PCNN, 94, 97
- Perceptron, 4
- Performance, 31–33, 43, 44, 70–74, 77, 78, 107, 125, 126, 134, 135, 140–143, 147, 148, 156, 157, 165, 166, 171, 172, 233, 238, 239, 286, 287
- Performance speedup, 127
- Perturbations, 284
- PHOG (pyramid histogram of oriented gradients), 303, 310
- Picture archiving and communication systems (PACS), 406, 408
- Pixel-wise classification (PWC), 137, 170
- Placebos, 343, 344
- Pneumonia, 300, 301, 415
- Pooling layers, 8, 9, 37, 110, 158, 226, 227, 306
- Population, 344, 345, 350, 359, 362, 372
- Pose estimation via hierarchical learning (PEHL), 274, 285–290, 292, 293
- Positive predictive value (PPV), 188
- Pre-trained CNN, 88, 307  
 model, 95, 306, 307
- Pre-trained models, 22, 26, 32, 45, 332–335
- Precision, 93, 147, 165, 170, 171, 212, 214, 287
- Predictive power, 181, 362, 364, 369, 370
- Preprocessing, 84, 95, 106, 140, 146, 159, 211, 315, 360
- Pretraining, 12, 14  
 layer-wise, 14, 347
- Principle component analysis (PCA), 203, 247–250, 259, 260, 305
- Probability, 5, 112, 114, 116, 126, 136, 138, 144, 160, 187, 255, 304, 310, 311, 343, 416
- Probability signals, 112, 118, 125, 128
- Problem formulation, 58, 67
- Prostate, 199, 206, 211, 219
- Prostate boundary, 197, 198, 200, 201, 210, 211, 214, 219
- Prostate likelihood map, 206, 207, 209, 210, 219
- Prostate region, 198, 206, 209
- Prostate segmentation, 199, 216, 217  
 MR, 199, 219
- Proximity mask, 166, 169
- Proximity patch, 166
- PsyEF (summary score for executive function), 361, 362, 364, 372
- PsyMEM (neuropsychological summary score for memory), 361, 372
- PWC (pixel-wise classification), 137, 170
- Python, 45, 93  
 Theano, 22, 45, 149, 230
- Q**
- Question, 43
- R**
- Radiology text, 410, 412, 417
- RadLex, 414
- Random forest (RF), 148, 156, 182, 185, 186, 188, 189, 191, 382
- Randomized deep networks, 344, 350, 352, 353, 356, 360
- Randomized denoising autoencoder marker (rDAM), 358, 359, 361–364, 367, 368, 370–373
- Randomized dropout network marker (rDrn), 358–368, 370–373
- RAVLT (Rey auditory verbal learning test), 361, 372, 373
- RBM (restricted Boltzmann machines), 15, 26, 347
- RCasNN (randomly initialized model), 141
- RCN (right consolidation), 310
- RDA (randomized denoising autoencoders), 355, 356, 359, 362, 364, 370, 371, 373
- Recall, 93, 141, 147, 163, 165, 170, 171, 262
- Receptive fields, 9
- Recognition, 57, 371

- Reconstructions, 203, 205, 252, 349  
 Recover, 10, 248, 250, 273, 274, 290  
 Rectified linear unit (ReLU), 20, 29, 40, 59, 93, 144, 158, 160, 227, 280  
 Recurrent neural network (RNN), 37, 40–42, 413  
 Registration, 206, 224, 231, 238, 255, 258, 324, 334, 386, 388  
     2-D/3-D, 272–275, 283, 284, 288, 289, 292, 293  
     real-time, 287, 290, 292, 293  
 Registration accuracy, 259, 265, 272, 285  
 Registration methods, 263, 287  
     2-D/3-D, 272, 288, 293  
     baseline HAMMER, 260, 262, 263  
     conventional, 248  
 Registration problems, 273–275, 292, 293  
 Registration-based methods, 223, 224, 228, 231, 233, 234, 238, 239  
 Regressors, 157, 275, 282, 289, 384  
 Reinforcement learning, 65, 67, 69  
 Representations, 12, 14, 17, 26, 35, 42, 55, 59, 84, 148, 184, 225, 226, 254, 275, 311–313  
 Responses, 59, 88, 91, 203  
 Restored wavelets, 111, 112, 125  
 Reward, 68  
 Right pleural effusion, 301, 310  
 Right pleural effusion (RPE), 301, 310, 315  
 RMSDproj (root mean squared distance in the projection direction), 287, 288  
 Robust approach, 156  
 Robust cell detection, 155, 165, 171  
 Robust cell detection using convolutional neural network, 165  
 ROI localization, 108, 115, 121, 126, 128  
 ROI (region of interest), 106, 112–115, 121, 125, 136, 183, 259, 260, 278, 285  
 Root mean squared error (RMSE), 289
- S**
- Sample enrichment, 344, 370  
 Sample sizes, 361, 365, 367, 373  
 Scales, 21, 56, 58, 143, 226, 304, 360  
 Screening stage, 144  
 SDA (stacked DA), 348–350, 354, 356, 357, 371  
 Segmentation, 22, 58, 71, 72, 84, 86, 100, 180, 181, 199, 200, 212, 224, 225, 227, 228, 231–234, 237–240, 302, 315  
     ground-truth, 212, 216  
     registration-based, 228, 231  
     semantic, 35, 135  
     stroma, 180, 182, 185  
 Segmentation accuracy, 107, 128, 214, 219  
 Segmentation maps, 328, 331, 334  
 Shallow models, 247, 250  
 Shapes, 64, 107, 125, 140, 156, 170, 210, 304  
 ShrinkConnect, 388–390, 401  
 SIFT, 26, 33, 85, 199, 248, 256, 303, 304  
 Signal-to-noise ratio (SNR), 246, 263, 322, 388  
 Similarity maps, 201  
 Similarity measures, 231, 272, 273, 285, 289, 292, 392  
 Small sample regime, 344, 350, 353  
 Sonographer, 106, 109, 117, 127  
 Source and target modalities, 382, 384, 390, 396  
 Sparse adaptive deep neural networks (SADNN), 57, 59, 61, 62, 64, 74  
 Sparse auto-encoder (SAE), 13, 14, 17, 203, 204, 247, 248, 252–254, 265  
 Sparse histogramming MI (SHMI), 293  
 Sparse patch matching, 206, 214  
 Sparse representation, 207, 210, 248, 396  
     coupled, 382, 389, 396, 400  
 Sparsely distributed objects, 134, 150  
 Spatial information, 143, 144, 149, 227, 239, 305, 382, 383, 401  
 Spatial locations, 28, 225, 385, 386, 389  
 Spatial resolution, 36, 37, 71  
 Stacked sparse auto-encoder (SSAE), 201, 203–206, 208, 211, 213, 214, 216, 219  
     networks, 205  
 Stages  
     boosting, 88, 91, 96  
 Standard deviation, 16, 117, 122, 163, 171, 186, 260, 265, 284, 292, 304, 310, 331, 345, 350  
 State-of-the-art image classification method, 305

- States, 67, 68  
 Stochastic gradient descent (SGD), 8, 30, 32, 37, 69, 76, 283  
 Stride, 28, 36, 138  
 Stroma, 180–183, 186, 190, 191  
 Stromal tissue, 180, 187  
 Structured regression model, 166–168, 171  
 Success rate, 274, 285  
 Superior performance, 128, 164, 165, 172, 214, 219, 265  
 Supervised SSAE, 201, 205, 206, 213, 214, 216, 217, 219  
 Synthetic data, 93, 294
- T**  
 Target image, 88, 201, 206, 210, 219  
 Target information, 159  
 Target modalities, 384  
 Target modality images, 389, 391  
 Target objects, 138, 199, 272, 277, 284, 285  
 Template image, 255, 258, 264  
 Tensorflow, 22, 45  
 Test patients, 117, 118, 120–123, 125  
 Test set, 71, 117, 230, 331, 363  
 Testing images, 162  
 Texture, 140, 185, 224, 325  
 Tissue, 180, 259  
 Tissue segmentation, 182  
 Topic modeling, 410  
 Total knee arthroplasty (TKA), 283–286, 290  
 Toy example, 40, 89, 90  
 Training, 31, 45, 60, 64, 74, 185, 230, 253, 326, 382, 386, 388, 389  
 two-stage, 331  
 Training annotations, 182, 187, 190, 191  
 Training data, 14, 27, 31, 32, 84, 140, 141, 156, 159, 166, 168, 183, 184, 187, 188, 203, 225, 239, 240, 250  
 paired, 384, 390, 391, 397, 401  
 synthetic, 289  
 Training dataset, 139–141, 230, 238  
 Training images, 32, 87, 110, 141, 239, 246, 255, 259, 260, 264, 331, 389  
 Training patches, 113, 115, 118, 121, 160, 183, 184, 186, 202, 203  
 Training patients, 117–119, 121  
 Training PEHL, 286, 289, 294  
 Training phase, 145, 229
- Training regressors, 274  
 Training samples, 8, 11, 13, 59, 93, 137, 139–141, 161, 167, 168, 247, 283, 347, 387  
 artificial, 332  
 Training set, 13, 63, 64, 71, 88, 93, 112, 115, 117, 145, 147, 182, 185, 203, 228, 230  
 stratified, 113, 115  
 Training time, 20–22, 74, 332  
 Transfer learning, 140, 299, 307, 324  
 Transformation parameters, 56, 61, 273, 278, 289, 290, 293, 349  
 Transformations, 56, 139, 210, 274, 276, 294, 345, 357, 383, 386  
 Translation, 9, 10, 35, 36, 42, 61, 62, 90, 93, 95, 112, 113, 115, 137, 140, 145, 226, 227, 254, 276–278  
 Treatment, 143, 182, 197, 342–345, 350  
 Trial, 200, 342, 350, 368, 371  
 Tricks, 31, 33  
 True negative rate (TNR), 188  
 True negative (TN), 315  
 True positive rate (TPR), 188  
 True positive (TP), 141, 142, 315, 323  
 True targets, 136  
 Tuberculosis, 301
- U**  
 Ultrasound images, 56, 71, 124  
 UMLS Metathesaurus, 414  
 Unified medical language system (UMLS), 414  
 Unsupervised SSAE, 201, 205, 213, 214, 216, 217
- V**  
 Validation set, 76, 230  
 Vanilla deep network (VDN), 389  
 Vanishing gradient problem, 11, 20, 41  
 Ventricle, 210, 234, 237, 255, 259  
 VGG network, 313  
 VGG-L4, 311  
 VGG-L5, 308, 313  
 Videos, 38, 106, 108, 112, 117, 123  
 fluoroscopic, 283  
 Virtual implant planning system (VIPS), 283, 285, 286, 290

Visible layer, 4, 15, 17, 18

Volumes, 39, 145, 147, 224

Volumetric data, 143

Voting confidence, 159

Voting offsets, 159

Voting units, 160

Voxels, 8, 57, 201, 208, 212, 227, 229, 234,

255, 272, 353, 354, 371, 384, 391,

392, 397

center, 385, 386, 392

## W

Weak learners, 352

Weight units, 160

White matter (WM), 259

Whole-slide imaging (WSI), 180, 187

Word embedding, 411

Word-to-vector models, 411

## X

X-ray attenuation map, 272

X-ray echo fusion (XEF), 284, 289

X-ray images, 272, 273, 275, 277, 279, 283,

284, 287, 289

real, 281, 294

synthetic, 275, 279, 281

X-ray imaging model, 274

XEF dataset, 287

## Z

Zone, 277

## Learn how to apply Deep Learning methods to medical imaging

Deep learning is providing exciting solutions for medical image analysis problems and is seen as a key method for future applications. This book gives a clear understanding of the principles and methods of neural network and deep learning concepts, showing how the algorithms that integrate deep learning as a core component have been applied to medical image detection, segmentation and registration, and computer-aided analysis, in a wide variety of application areas.

**Deep Learning for Medical Image Analysis** is a great learning resource for academic and industry researchers in medical imaging analysis, and for graduate students taking courses on machine learning and deep learning for computer vision and medical image computing and analysis.

### With this book you will learn:

- Common research problems in medical image analysis and their challenges
- Deep learning methods and theories behind approaches for medical image analysis
- How the algorithms are applied to a broad range of application areas: chest X-ray, breast CAD, lung and chest, microscopy and pathology, etc.

### About the Editors:

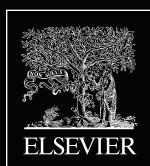
**S. Kevin Zhou, PhD**, Principal Key Expert at Siemens Healthineers Technology Center, Princeton, NJ, United States.

**Hayit Greenspan**, Professor at the Biomedical Engineering Department, Faculty of Engineering, Tel Aviv University, Ramat-Aviv, Israel.

**Dinggang Shen**, Professor of Radiology at the Biomedical Research Imaging Center (BRIC), and Computer Science, and Biomedical Engineering Departments in the University of North Carolina at Chapel Hill (UNC-CH), Chapel Hill, NC, United States.

Engineering/  
Communications Design

ISBN 978-0-12-810408-8



ACADEMIC PRESS

An imprint of Elsevier  
[elsevier.com](http://elsevier.com)

