Christian Payer

Integrating Spatial Configuration of Anatomical Structures into Convolutional Neural Networks

DOCTORAL THESIS

to achieve the university degree of
Doktor der technischen Wissenschaften

submitted to

Graz University of Technology

Supervisors

Prof. Dr. Horst Bischof
Institute of Computer Graphics and Vision, Graz University of Technology

Assoc. Prof. Dr. Ben Glocker
Biomedical Image Analysis Group, Imperial College London

Advisor

Dr. Martin Urschler
School of Computer Science, The University of Auckland

Graz, Austria, Oct. 2020

Dr. Evil! I didn't spend six years in evil medical school to be called mister.

Dr. Evil (⋆1939)

Abstract

Localization of anatomical landmarks is an important step in medical image analysis, e.g., to initialize methods for segmentation and registration, or to perform diagnosis based on the landmarks' spatial distribution or on structures near landmarks. Due to the costs of image acquisition and the large manual annotation effort required from experts, in many medical image analysis applications, only a limited amount of training data for machine learning algorithms is available. Training convolutional neural networks (CNNs), which are the currently best performing machine learning method, with only small datasets is a challenging task. Inspired by machine learning methods that combine local feature extraction with graphical models, we propose a CNN architecture that directly integrates a graphical model as part of the network architecture to reduce the overall need for large training datasets. Proposed for the heatmap regression framework for anatomical landmark localization, our fully convolutional SpatialConfiguration-Net (SCN) splits the localization task into two simpler sub-problems. Thus, one component of the SCN is dedicated to generating locally accurate but ambiguous candidate predictions, while the other component improves robustness to ambiguities by incorporating a graphical model of the spatial configuration of landmarks. In an extensive in-depth analysis, we show that the SCN successfully splits up the localization tasks, while the SCN outperforms related methods in terms of landmark localization error on a variety of 2D and 3D landmark localization datasets, i.e., hand radiographs, lateral cephalograms, hand MRIs, and spine CTs. Especially when using only limited amounts of training data, the performance improvement of our SCN to other baseline networks is substantial. Moreover, we show the generic applicability of the SCN on several tasks, i.e., face point detection, human pose estimation, and age estimation on living individuals, as well as multi-label whole heart segmentation, and vertebrae localization and segmentation. Comparisons on two MICCAI challenges shows the superior performance of our proposed method, outperforming all other participants and ranking first in both.

Keywords. medical image analysis, convolutional neural networks, local appearance, spatial configuration, landmark localization, heatmap regression, multi-label segmentation

Kurzfassung

Lokalisierung von anatomischen Landmarken ist eine wichtige Aufgabe der medizinischen Bildanalyse und wird beispielsweise für die Initialisierung von Segmentierungs- und Registrierungsmethoden oder für Diagnosen anhand der räumlich Verteilung von Landmarken oder ihnen benachbarter Strukturen verwendet. Wegen der hohen Kosten der Bildaufnahme verbunden mit dem hohen Aufwand für Experten manuelle Annotationen zu erzeugen sind in der medizinischen Bildanalyse oftmals nur wenige Trainingsdaten für maschinelle Lernalgorithmen verfügbar. Bedauerlicherweise ist das Trainieren von Convolutional Neural Networks (CNNs), einer der derzeit besten maschinellen Lernalgorithmen, mit wenigen Daten herausfordernd. Inspiriert von anderen Lernalgorithmen, welche lokale Merkmale extrahieren und mit graphischen Modellen kombinieren, stellen wir eine neue CNN Architektur vor, welche direkt ein graphisches Modell integriert, um den Bedarf an großen Trainingsdatensätzen zu reduzieren. Angewandt im Heatmap Regression Framework für die Lokalisierung von anatomischen Landmarken, teilt unser fully-convolutional SpatialConfiguration-Net (SCN) Lokalisierung in zwei kleinere Teilprobleme auf. Eine Komponente des SCN widmet sich einer lokal akkuraten aber möglicherweise nicht eindeutigen Vorhersage, während die andere Komponente die Robustheit zu solchen Mehrdeutigkeiten reduziert, indem sie ein graphisches Modell der räumlichen Anordnung der Landmarken integriert. In unserer umfangreichen und detaillierten Analyse zeigen wir, dass das SCN erfolgreich die Lokalisierungsaufgabe in zwei Teile aufteilt, wobei das SCN bessere Ergebnisse als ähnliche Methoden in einer Reihe von 2D- und 3D-Lokalisierungsdatensätzen liefert, im Speziellen bei Röntgenbildern der Hand, lateralen Kephalogrammen, MR-Bildern der Hand und CT-Bildern der Wirbelsäule. Insbesondere wenn nur wenige Trainingsdaten verfügbar sind, ist der Leistungssprung unseres SCN verglichen zu anderen Basismodellen beträchtlich. Des Weiteren zeigen wir die generische Anwendbarkeit unseres SCN, sowohl bei der Lokalisierung von Punkten im Gesicht und der Bestimmung der Pose eines Menschen als auch bei der Segmentierung des Herzens in seine einzelnen Teilstrukturen und der Lokalisierung und Segmentierung einzelner Wirbel. Ergebnisse in zwei MICCAI Wettbewerben zeigen die überlegene Leistung unserer vorgestellten Methode verglichen mit denen anderer Teilnehmer, wobei bei beiden Wettbewerben der ersten Platz errungen wurde.

Affidavit

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly indicated all material which has been quoted either literally or by content from the sources used.

The text document uploaded to TUGRAZonline is identical to the present doctoral thesis.

Date	Signature

Acknowledgments

First and foremost I would like to thank the members of our Medical Image Processing group, specifically Dr. Martin Urschler and Dr. Darko Štern, for their guidance throughout my PhD studies. Without their invested time in supervising me and our paper writing sessions, the final outcome and quality of my publications would not have been possible. I would also like to thank Prof. Horst Bischof for his support and input that helped to improve several crucial parts of my publications. I also like to thank Assoc. Prof. Ben Glocker for agreeing and taking the time to serve as a second supervisor of my thesis. Furthermore, I thank my fellow colleagues at the Institute of Computer Graphics and Vision for the fruitful discussions and occasional after-work beers. Lastly, I would also like to thank my girlfriend, friends, and family for their moral support during the ups and downs of my PhD studies.

Contents

1 Introduction **1**
- 1.1 Motivation . . . 1
- 1.2 Contributions . . . 5
- 1.3 Structure of the Thesis . . . 6

2 Deep Learning **7**
- 2.1 Machine Learning . . . 8
  - 2.1.1 Machine Learning Categories . . . 8
  - 2.1.2 Generalization . . . 10
  - 2.1.3 Training, Validation, and Test Sets . . . 12
- 2.2 Artificial Neural Networks . . . 13
  - 2.2.1 Artificial Neuron . . . 14
  - 2.2.2 Layered Neural Networks . . . 15
  - 2.2.3 Activation Functions . . . 17
  - 2.2.4 Loss Functions . . . 19
- 2.3 Neural Networks Optimization . . . 20
  - 2.3.1 Variants of Stochastic Gradient Descent Optimizers . . . 22
  - 2.3.2 Backpropagation . . . 24
  - 2.3.3 Weight Initialization . . . 25
  - 2.3.4 Normalization . . . 26
- 2.4 Convolutional Neural Networks . . . 27
  - 2.4.1 Convolution Layers . . . 28
    - 2.4.1.1 Properties of Convolution Layers . . . 29
    - 2.4.1.2 Terminology . . . 30
  - 2.4.2 Pooling Layers . . . 31
  - 2.4.3 Fully Connected Layers . . . 32

2.4.4 Transposed Convolution Layers . . . 33
2.4.5 Other Convolution Layers . . . 34
2.5 Network Regularization . . . 34
2.5.1 Weight Regularization . . . 34
2.5.2 Dropout . . . 35
2.5.3 Data Augmentation . . . 36
2.6 Summary . . . 38

3 Related Work on Landmark Localization 39
3.1 Landmark Localization . . . 39
3.2 Graphical Models for Landmark Localization . . . 40
3.2.1 Active Shape Models . . . 40
3.2.2 Pictorial Structures . . . 42
3.2.3 Markov Random Fields and Hidden Markov Models . . . 44
3.2.4 Other Graphical Models . . . 46
3.3 Image Feature Responses for Landmark Localization . . . 46
3.3.1 Random Forests . . . 47
3.3.2 Convolutional Neural Networks . . . 49
3.4 CNNs in Anatomical Landmark Localization . . . 53
3.5 Our Contributions to Landmark Localization . . . 55

4 Using CNNs for Landmark Localization 57
4.1 CNNs for Landmark Localization . . . 57
4.2 CNNs for Coordinate Regression . . . 58
4.3 CNNs for Regressing Gaussian Heatmaps . . . 59
4.3.1 Learning Gaussian Heatmaps . . . 60
4.3.2 Obtaining the Landmark Coordinates . . . 61
4.3.3 Fully Convolutional Network Architectures . . . 63
4.4 Summary . . . 64

5 Integrating Spatial Configuration into CNNs 65
5.1 Fundamental Concept of the SpatialConfiguration-Net . . . 65
5.1.1 Transforming Local Responses of Anatomical Structures . . . 66
5.1.2 Optimization of Local Appearance $\odot$ Spatial Configuration . . . 69
5.1.3 Interaction of Local Appearance $\Leftrightarrow$ Spatial Configuration . . . 69
5.2 SpatialConfiguration-Net for Heatmap Regression . . . 70
5.2.1 Local Appearance . . . 70
5.2.2 Spatial Configuration . . . 71
5.3 Summary . . . 72

6 Experimental Setup **73**
6.1 Overview of the Framework . . . 73
6.2 Data Preprocessing and Augmentation . . . 74
6.2.1 Intensity Transformations . . . 74
6.2.2 Spatial Transformations . . . 75
6.3 Dataset for Anatomical Landmark Localization . . . 76
6.4 Networks for Anatomical Landmark Localization . . . 79
6.5 Evaluation Metrics . . . 82
6.5.1 Landmark Localization Evaluation Metrics . . . 82
6.5.2 Segmentation Evaluation Metrics . . . 83

7 Anatomical Landmark Localization Results **85**
7.1 Comparison to the State-Of-The-Art . . . 85
7.1.1 Hand Radiographs (2DHand) . . . 86
7.1.1.1 Comparison of Network Architectures . . . 88
7.1.1.2 Improvements of the SpatialConfiguration-Net . . . 88
7.1.1.3 Comparison to Other Methods . . . 90
7.1.2 Hand MRIs (3DHand) . . . 91
7.1.3 Lateral Cephalograms (2DSkull) . . . 93
7.1.4 Spine CTs (3DSpine) . . . 95
7.2 Learning the Size $\boldsymbol{\sigma}$ of the Gaussian Heatmaps . . . 98
7.3 Interacting Local Appearance $\Leftrightarrow$ Spatial Configuration . . . 103
7.3.1 Individual Components of the SCN . . . 103
7.3.2 Representative Heatmap Outputs . . . 104
7.4 Reducing the Number of Training Images . . . 109
7.5 Summary . . . 111

8 Applications and Evaluations on Other Domains **113**
8.1 Landmark Localization in Computer Vision Datasets . . . 114
8.1.1 Method . . . 114
8.1.2 Evaluation . . . 114
8.1.2.1 Datasets . . . 115
8.1.2.2 Implementation Details . . . 115
8.1.2.3 Results . . . 116
8.1.3 Conclusion . . . 118
8.2 Fully Automatic Age Estimation . . . 119
8.2.1 Method . . . 120
8.2.2 Evaluation . . . 122
8.2.2.1 Dataset . . . 122
8.2.2.2 Implementation Details . . . 122
8.2.2.3 Age Estimation from Hand MR Images . . . 123

8.2.2.4 Multi-Factorial Age Estimation from MR Images 125
8.2.3 Conclusion 127
8.3 Multi-Label Whole Heart Segmentation 128
8.3.1 Method 128
8.3.1.1 Heart Localization 129
8.3.1.2 Whole Heart Segmentation 130
8.3.2 Evaluation 131
8.3.2.1 Dataset 131
8.3.2.2 Implementation Details 132
8.3.3 Results 133
8.3.4 Conclusion 135
8.4 Vertebrae Segmentation 136
8.4.1 Method 137
8.4.1.1 Spine Localization 137
8.4.1.2 Vertebrae Localization 138
8.4.1.3 Vertebrae Segmentation 140
8.4.2 Evaluation 141
8.4.2.1 Dataset 141
8.4.2.2 Implementation Details 141
8.4.2.3 Results 142
8.4.3 Conclusion 145
8.5 Summary 145

9 Discussion and Conclusion **147**
9.1 Discussion 147
9.1.1 Landmark Localization with CNNs Using Heatmap Regression . . . 148
9.1.2 SpatialConfiguration-Net for Anatomical Landmark Localization . . 149
9.1.3 Other Applications of the SpatialConfiguration-Net 150
9.2 Limitations and Future Work 152
9.3 Conclusion 154

A List of Acronyms **155**

B List of Publications **157**
B.1 Peer-Reviewed Publications Relevant for the Thesis 157
B.2 Additional Peer-Reviewed Publications 161

Bibliography **165**

List of Figures

1.1 Examples of parts of the human body whose spatial configuration is restricted by anatomical constraints. . . . 2
1.2 Example images of local appearance and spatial configuration. . . . 3

2.1 Basic categorization of machine learning with example tasks. . . . 9
2.2 Visualization of the training error, generalization error, and the generalization gap. . . . 10
2.3 Visualization of underfitting, overfitting, and optimal capacity. . . . 11
2.4 Schematic representations of a biological and an artificial neuron. . . . 14
2.5 Schematic representations of a layered neural network. . . . 16
2.6 Sigmoid activation function. . . . 17
2.7 Hyperbolic tangent activation function. . . . 18
2.8 ReLU activation function. . . . 18
2.9 Leaky ReLU activation function. . . . 19
2.10 Example of calculating output values of a discrete convolution for a 2-dimensional image. . . . 28
2.11 Visualization of a convolution layer with multiple input and output channels. 29
2.12 Visualization of a convolution operation without padding and with padding. 31
2.13 Visualization of a convolution operation with stride of two. . . . 31
2.14 Computing the output values of a discrete pooling operation. . . . 32
2.15 Visualization of a fully connected layer. . . . 32
2.16 Computing the outputs of a transposed convolution layer. . . . 33
2.17 Example of the employed data augmentation techniques. . . . 36

3.1 Example of *active shape models*. . . . 41
3.2 Example of *pictorial structures*. . . . 43

3.3 Example graphs for *Markov random fields* and *hidden Markov models*. . . . 45
3.4 Example of features used for *classification* and *regression random forests*. . 47
3.5 Example heatmap outputs from *random regression forests*. 48
3.6 Coordinate regression for human pose estimation using CNNs with iterative refinement. 49
3.7 Heatmap regression for human pose estimation using CNNs. 50
3.8 Heatmap regression for human pose estimation using fully convolutional CNNs. 51
3.9 Heatmap regression for human pose estimation using stacked hourglass networks. 52

4.1 Overview of the *coordinate regression* framework for landmark localization. 58
4.2 Schematic representations of the network architecture for *coordinate regression*. 59
4.3 Overview of the *heatmap regression* framework for landmark localization. . 59
4.4 Example heatmap output for different σ_i for the zoomed region of landmark $\mathcal{L}_i$. 62
4.5 Schematic representations of the network architecture for *heatmap regression*. 63

5.1 Estimate positions of other landmarks by transforming local predictions. . . 66
5.2 Estimate positions of other landmarks with convolutions. 67
5.3 Interaction and optimization of the *local appearance* and *spatial configuration* components of the SpatialConfiguration-Net. 68
5.4 Schematic representation of our proposed SCN for landmark localization. . 71

6.1 Example images of the evaluated datasets. 78

7.1 Example image with all from the SCN predicted landmarks of all images from the 2DHand dataset. 86
7.2 Cumulative distribution of the image-specific point-to-point error on 2DHand dataset. 87
7.3 Cumulative distribution of IPE on 2DHand dataset for varying input/heatmap target sizes of our SCN architecture. 89
7.4 Cumulative distribution of IPE on 2DHand dataset for a varying number of intermediate filter outputs of our SCN architecture. 89
7.5 Example projection images with all from the SCN predicted landmarks of all images from the 3DHand dataset. 91
7.6 Cumulative distribution of IPE on the 3DHand dataset. 92
7.7 Example images with all from the SCN predicted landmarks of all images from the 2DSkull dataset. 93
7.8 Example projection images with all from the SCN predicted landmarks of all images from the 3DSpine dataset. 95

7.9 Cumulative distribution of IPE on the 2DHand dataset for different fixed σ values and weighting factors α. 98
7.10 Relation of learned σ_i values to the PE_i for each landmark $\mathcal{L}_i$. 99
7.11 Progression of learned σ_i values during training for landmarks on the phalanges and metacarpals. 101
7.12 Progression of learned σ_i values during training for landmarks on ulna, radius, and carpals. 102
7.13 Cumulative distribution of IPE on 2DHand dataset for local appearance and spatial configuration components of our SCN. 103
7.14 Example input and result images from the 2DHand dataset. 105
7.15 Example input and result images from the 3DHand dataset visualized via maximum projection. 106
7.16 Example input and result images from the 2DSkull dataset. 107
7.17 Example input and result images from the 3DSpine dataset visualized via maximum projection. 108
7.18 Cumulative distribution of IPE on 2DHandReduced dataset for different numbers of training images. 110

8.1 Example images of the AFLW dataset for face point detection and the FLIC dataset for human pose estimation. 114
8.2 Cumulative distribution of IPE on the AFLW dataset. 116
8.3 Cumulative distribution of PE_{wrist} and PE_{elbows} on the FLIC dataset. . . . 117
8.4 Images showing the structures containing relevant information for age estimation. 119
8.5 Overview of our method for fully automatic age estimation. 120
8.6 Schematic representation of our CNN for age estimation using cropped regions around age-relevant structures as input. 121
8.7 Visualizations of the hand image regions delivering the most responses for generating the age predictions of a CNN grouped by age ranges. 125
8.8 The influences of the individual anatomical sites for all subjects over the whole predicted age range. 127
8.9 Overview of our fully automatic two-step multi-label whole heart segmentation pipeline. 129
8.10 Overview of the proposed coarse-to-fine fully automatic vertebrae localization, identification, and segmentation. 136
8.11 Schematic visualization of spine localization. 138
8.12 Schematic visualization of vertebrae localization. 139
8.13 Schematic visualization of vertebrae segmentation. 140

List of Tables

7.1 Three-fold cross-validation results for the 2DHand dataset. 87
7.2 Cross-validation results for the 3DHand dataset. 92
7.3 Results on the test images from 2DSkull dataset. 94
7.4 Two-fold cross-validation and test results on the 3DSpine dataset. 96
7.5 Localization results on the 2DHand dataset for different values of fixed σ and weighting factors α. 98
7.6 Individual final σ and mean PE values for the individual landmarks of the 2DHand dataset. 100
7.7 Localization results on the 2DHand dataset of the individual heatmap outputs. 103
7.8 Cross-validation results on 895 images from 2DHandReduced dataset, averaged from three random selections of training images. 110

8.1 Results of fully automatic age estimation methods for a four-fold cross-validation on hand MR images of 141 subjects younger than 18 years. . . . 124
8.2 Results for CNNs trained on all combinations of the anatomical sites for four-fold cross-validation on images of 322 subjects in the age range of 13 to 25 years. 126
8.3 Segmentation results of three-fold cross-validation on the MM-WHS 2017 challenge training set. 133
8.4 Ranked results on the CT test sets of the MM-WHS 2017 challenge. 134
8.5 Ranked results on the MRI test sets of the MM-WHS 2017 challenge. . . . 134
8.6 Results of a three-fold cross-validation on the VerSe 2019 challenge training set. 143
8.7 Results on the overall VerSe 2019 challenge test set. 144

1 Introduction

Contents

1.1 Motivation 1
1.2 Contributions 5
1.3 Structure of the Thesis 6

> You know I, I just think, that things have a way of working themselves out.
>
> *Walter White*

1.1 Motivation

In recent years, the areas of computer vision and medical image analysis have experienced a disruptive shift towards deep Convolutional Neural Networks (CNNs) for image-based classification [78], object localization [41], as well as semantic segmentation problems [88]. Despite the foundations for these deep learning approaches being known to the research community for at least 20–30 years [80], recently three factors revolutionized many computer vision and medical image analysis applications to even achieve beyond human expert performance. Firstly, improvements in network optimization, e.g., Rectified Linear Unit (ReLU) [49] and Adam [71], secondly, hardware improvements that enable network training on consumer GPUs [16, 18, 78], and thirdly, the availability of huge annotated training datasets with up to millions of images, e.g., ImageNet [119], has enabled stable and efficient training of these deep neural network architectures. While research in improving network optimization and more efficient hardware has enabled training of ever deeper and more powerful CNNs, reducing the requirement of deep network architectures for huge annotated training datasets remains challenging and an open research problem. Although

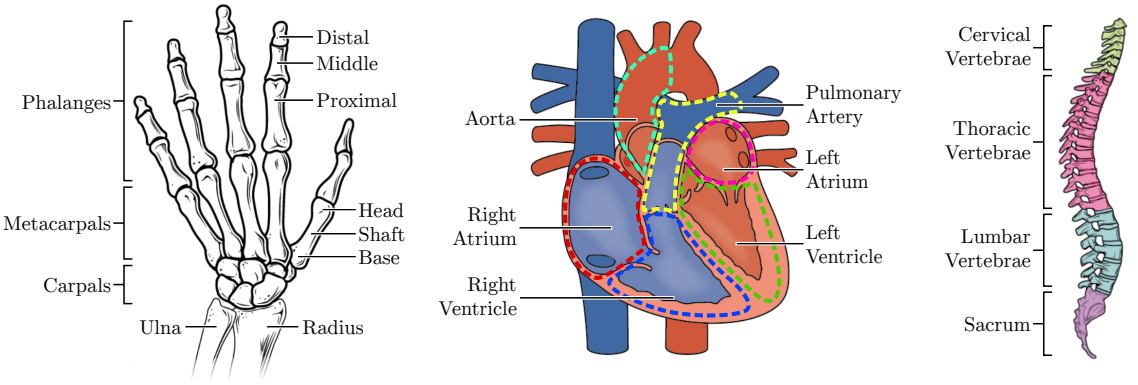
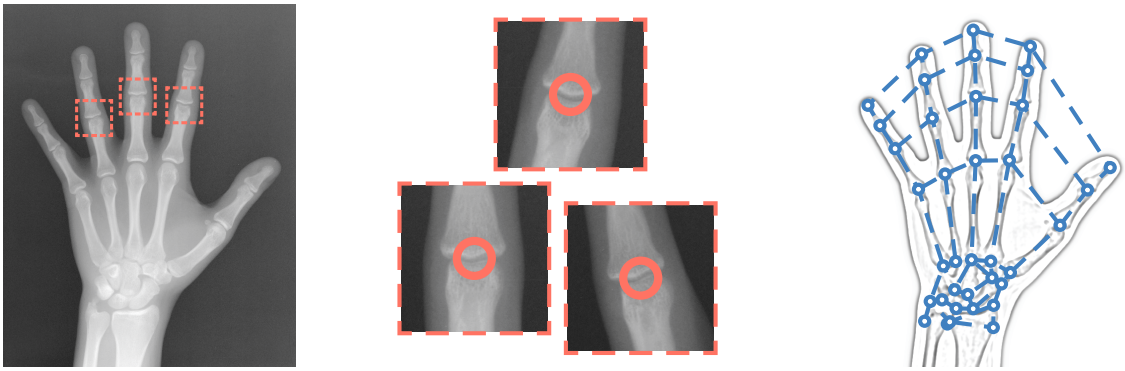


Figure 1.1: Examples of parts of the human body whose *spatial configuration* is restricted by anatomical constraints. The images of the body parts are adapted from [7]. The left hands are composed of carpals, metacarpals, and phalanges, which in turn are composed of distal, middle, and proximal phalanges. The heart can be split up into the left and right heart, which can be split up into the atria and ventricles. The left and right heart are connected via arteries and veins to the systemic and pulmonary circulatory system. The spinal column is composed of a certain number of vertebrae, which are separated into cervical, thoracic, and lumbar vertebrae, as well as the sacrum. While there exist variations due to pathologies, the basic structure of these body parts is required to be fixed, as otherwise, a correct functionality is not guaranteed.

it is generally demanding to create huge datasets with high quality ground-truth annotations, this is even more problematic in the medical imaging domain for two reasons: Firstly, ethical and financial concerns hinder large-scale acquisition and sharing of medical images. Secondly, trained specialists are needed to create ground-truth annotations, which is often difficult, costly, and time-consuming. Thus, compared to computer vision tasks, CNN-based approaches in many medical image analysis tasks have to be able to cope with significantly smaller quantities of annotated training data.

With only limited amounts of annotated training data, the available information in the data needs to be explored as comprehensively as possible. Due to their inherently flexible design, state-of-the-art CNNs are able to capture various layers of abstraction from the input data, ranging from low-level intensity to high-level context information. During the training process, the CNN identifies the information that optimizes a measure based on the discrepancy of ground-truth and predicted annotation. Unfortunately, although there is no restriction, there is also no guarantee which type of information a CNN is using to model the ground-truth annotations from the input data. When the training dataset is limited, the CNN could pick up spurious correlations specific to the few training images that lead to best results on the observed data, but to poor performance and generalization to unseen data. However, also in this limited dataset, there could be information present that generalizes well. In the medical imaging domain, a type of information that typically generalizes well is the geometric structure or *spatial configuration* of objects, as it is often restricted by anatomical and physical constraints (see Fig. 1.1). Hence, CNNs that explicitly incorporate the *spatial configuration* of objects should require a lower amount of



Input Local Appearance Spatial Configuration

Figure 1.2: Example images of *local appearance* and *spatial configuration*. The left image shows an example radiograph of a left hand. The images in the center show regions cropped around the bone joints between the proximal and middle phalanges for the index, middle, and ring finger. While this *local appearance* can be used to accurately localize each of the landmarks, it cannot be used to identify which cropped region corresponds to which finger. However, on radiographs of the left hand, the ring finger is always left of the middle finger, which in turn is always left of the index finger. Hence, as can be seen on the image on the right side, the *spatial configuration* of landmarks can be used to restrict the locations to only feasible solutions, which allows distinguishing of locally ambiguous landmarks.

annotated training data, by more effectively using the available information of the input images.

We explore using the *spatial configuration* of anatomical structures, while focusing on anatomical landmark localization. Anatomical landmark localization is an important task in medical image analysis, and has a vast amount of applications.

Frequently, landmark localization is used as a preprocessing step for segmentation [6, 53, 62, 104]. When landmarks are densely localized on the border around the structure of interest, e.g., as in [62] for the lungs, the landmarks can be connected to get the outline of the structure, resulting in the final segmentation. Other methods use sparse but distinct landmarks around the structure of interest to initialize more sophisticated segmentation methods that benefit from better initialization, e.g., multi-atlas-based segmentation for heart substructures as in [104]. Similarly, landmark localization is also used as preprocessing for coarse-to-fine segmentation by locating at first the structure of interest, e.g., centroids of individual vertebrae [74, 132]. Here, after localizing the individual vertebrae, a deformable vertebrae model is fit to the located vertebra, resulting in the final segmentation. Additionally, landmark localization may be used for identifying specific structures within vascular graphs, e.g., the plumonary artery [72], to analyze connections for separating arteries and veins. Anatomical landmarks are also used for image registration [65, 148], where in both moving and fixed image the same landmarks are first localized and registered to give a better initialization for the subsequent feature-based image registration.

Analyzing the relative positions of localized landmarks has also direct diagnostic pur-

poses. For example, landmarks on vertebrae are used to assess deformations of the spine, i.e., kyphosis and scoliosis [11]. Other diagnostic applications include identifying developmental dysplasia of the hip [144] and anatomically abnormal positions of teeth and jaws [152]. Additional forensic applications of landmark localization include bone age estimation [134], where the biological age of a person is estimated by focusing on the epiphyseal gaps around landmarks located on bone joints of the hand.

Due to its many applications, anatomical landmark localization has received much attention in the medical imaging community. Unfortunately, various difficulties exist for developing general methods that solve the tasks satisfactorily. Due to different appearances of landmarks located anywhere across the the human body, localization methods that work for general landmarks are predominantly using machine learning. These machine learning-based methods usually require lots of annotated training data to work well. However, as annotations are typically provided by medical experts, in the medical imaging domain annotations are costly to obtain, which results in a usually small annotated datasets. Additionally to the usually low number of annotated training images, locally similar structures often introduce difficulties due to ambiguity in anatomical landmark localization. Such ambiguities make it hard to achieve a low landmark localization error, defined as both high robustness towards landmark misidentification as well as high accuracy locally at each identified landmark.

As anatomical landmarks often have a strong *spatial configuration*, machine learning-based approaches frequently combine local landmark predictions with explicit handcrafted graphical models, aiming to restrict predictions to feasible spatial configurations as seen in the training data. Thus, the landmark localization problem is simplified by separating the task into two successive steps. The first step is dedicated to locally accurate but potentially ambiguous candidate predictions, while in the second step graphical models [22, 34] eliminate ambiguities to improve robustness towards landmark misidentification (see Fig. 1.2). Although these methods effectively use the prior knowledge of the *spatial configuration* of landmarks to improve results, often both steps are optimized not simultaneously but subsequently, which reduces performance and restricts flexibility.

We hypothesize that while retaining the CNNs' powerful representation capability and flexibility, their need for a large amount of training data can be reduced by following the idea explored with handcrafted graphical models. Such graphical models incorporate prior knowledge that the feasible location of a landmark is not distributed uniformly in image space, but is constrained by the locations of other anatomical landmarks. Thus, the hypothesis is that this prior knowledge on the *spatial configuration* of landmarks could enable CNNs to simplify the modeling of the underlying distribution of feasible anatomical configurations and therefore less training data is required.

1.2 Contributions

In this thesis, we propose a novel network architecture, the SpatialConfiguration-Net (SCN), which directly integrates geometric information of the *spatial configuration* of anatomical structures. The two-component end-to-end trained SCN learns to dedicate one component to deliver locally accurate but potentially ambiguous candidate predictions based on the *local appearance*, and the other component to focus on incorporating the *spatial configuration* to improve robustness by eliminating ambiguities. This way, we combine the high-level knowledge of the relative positions of anatomical structures with the superior feature extraction capabilities of CNNs. While in principle our proposed method works for tasks with target objects having restricted relative positions, we focus mainly on the task of anatomical landmark localization.

We perform anatomical landmark localization by regressing heatmaps, which has been shown to deliver good performance in similar computer vision tasks, e.g., human pose estimation [114, 142]. We extend the heatmap regression framework by adapting the loss function to also allow learning of the size of the target Gaussian function representing a measure of landmark uncertainty. This enables learning narrower heatmaps for landmarks where the network is certain to make a smaller error and broader heatmaps for landmarks where the network is more uncertain.

We perform thorough evaluation of our proposed method on several datasets from the medical imaging domain, where our SCN outperforms other methods. Especially when the number of training images is drastically reduced, the SCN outperforms other CNN-based methods by a large margin. In-depth analysis of the SCN confirms that our SCN learns to dedicate one component to deliver locally accurate but potentially ambiguous candidate predictions based on the *local appearance*, and the other component to focus on incorporating the *spatial configuration* to improve robustness towards landmark misidentification by eliminating ambiguities. Additionally, evaluation of our proposed loss function shows that the loss models a measure of uncertainty as the average landmark localization error correlates with the predicted heatmap size.

We also apply the SCN to datasets from the computer vision domain, as well as other applications with strong spatial constraints. Here, the SCN produces state-of-the-art results for face point detection and human pose estimation. We show results on the forensically relevant application of fully automatic age estimation from Magnetic Resonance (MR) images. When using landmark localization results of the SCN such that the age estimation network focuses on anatomical structures containing information relevant for age development, the predicted age is much more accurate than without focusing on these structures. We adapt the SCN for multi-label segmentation, where the *spatial configuration* of anatomical structures is also valuable. We apply the adapted SCN to a dataset of multi-label whole heart segmentation, where it outperforms other proposed methods. Finally, we apply the SCN for vertebrae localization and segmentation, where it also outperforms other proposed methods.

1.3 Structure of the Thesis

As our proposed methods are based on CNNs, we give an overview of the principles of Artificial Neural Networks (ANNs) in general and CNNs more specifically in Chapter 2. The related work of landmark localization is explored in Chapter 3. The heatmap regression framework for using CNNs in landmark localization, as well as our adaptions to also allow predicting the sizes of heatmaps, are explained in Chapter 4. In Chapter 5, we give details of the SCN for landmark localization and explain how it integrates the *local appearance* and *spatial configuration* of landmarks. Implementation details as well as several metrics used for evaluation are shown in Chapter 6. Results on several datasets for anatomical landmark localization are shown in Chapter 7, where we also perform in-depth evaluation of our proposed framework. We show additional applications in both computer vision and medical imaging in Chapter 8. Final discussions and conclusions are given in Chapter 9.

2 Deep Learning

Contents

2.1 Machine Learning 8
2.2 Artificial Neural Networks 13
2.3 Neural Networks Optimization 20
2.4 Convolutional Neural Networks 27
2.5 Network Regularization 34
2.6 Summary 38

> It's not the years, honey. It's the mileage.
>
> *Dr. Henry Walton "Indiana" Jones, Jr.*

In this chapter, we give an introduction to machine learning in general in Section 2.1, while in the rest of the chapter we focus on Artificial Neural Networks (ANNs), one of the recently most dominantly used machine learning techniques. The concepts and components of ANNs are introduced in Section 2.2, while we describe in Section 2.3, how ANNs are being trained and optimized. Building upon ANNs, in Section 2.4, we give an introduction to Convolutional Neural Networks (CNNs), which are the currently best-performing machine learning method. As CNNs require lots of data to perform well, various regularization techniques to improve the performance of CNNs with smaller amounts of training data are described in Section 2.5. A final short summary of the chapter is given in Section 2.6.

Most content of this chapter is general and describes the foundations of ANNs. The descriptions are based on my knowledge gained during courses about ANNs, as well as books from Bishop [9] and Goodfellow et al. [47]. To not clutter the individual sections, references to [9, 47] are reduced to a minimum.

2.1 Machine Learning

Machine learning deals with algorithms and models that teach a computer to solve a specific task without using explicit instructions, but by identifying patterns from data instead. A more formal definition of machine learning is as follows:

> "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E."
>
> — Tom M. Mitchell, *Machine Learning* [95].

Following this definition, a machine learning program learns to solve a specific task T. Common tasks include optical character recognition, speech recognition, image segmentation, estimating stock market prices, playing games, etc. The performance measure P, which declares how well the machine learning program solves the task T, has to be specifically defined for T. Some example performance measures are the misclassification rate (for optical character recognition, speech recognition, image segmentation), the mean absolute difference (for estimation of future stock market prices, or the chronological age of a person), and the outcome or score (for playing games). The machine learning program optimizes the performance measure P for task T by obtaining more and more experience E. Such experience E may be increased by longer *training* from more *data*, which allows the program better identifying patterns in the data to solve task T.

2.1.1 Machine Learning Categories

Machine learning programs solve tasks by identifying patterns from data. The dataset that is used for training the machine learning program is called the *training data* $\mathcal{D}^{\text{train}}$. The dataset $\mathcal{D}^{\text{train}}$ consists of N training examples, i.e., $\mathcal{D}^{\text{train}} = \{\mathbf{x}^{(1)} \dots \mathbf{x}^{(N)}\}$. As in this thesis we use only data that is representable as vectors, we define each data sample $\mathbf{x}^{(i)}$ as an n-dimensional vector, i.e., $\mathbf{x}^{(i)} \in \mathbb{R}^n$ for all $i = \{1 \dots N\}$. Depending on the target task and the annotation of the training examples, machine learning is commonly categorized into *supervised*, *unsupervised*, and *reinforcement learning* [9, 95] (see Fig. 2.1). In *supervised learning*, each training data sample $\mathbf{x}^{(i)}$ is annotated with a corresponding m-dimensional ground-truth label $\overset{*}{\mathbf{y}}{}^{(i)} \in \mathbb{R}^m$. The supervised machine learning program identifies patterns in the training data to be able to predict labels $\hat{\mathbf{y}}^{(i)} \in \mathbb{R}^m$ with a small error to the ground-truth labels $\overset{*}{\mathbf{y}}{}^{(i)}$. Contrary to *supervised learning*, in *unsupervised learning*, the training data lacks corresponding ground-truth annotations $\overset{*}{\mathbf{y}}{}^{(i)}$. The machine learning program is set up to identify underlying patterns in the training data to group the examples based on some similarity metric. Combining both *supervised* and *unsupervised* learning, in *semi-supervised learning* not all but only a subset of the training data is annotated with corresponding ground-truth labels. *Reinforcement learning* is different from the aforementioned categories, as it does not learn from annotated ground-truth labels or the structure of the data but from the outcome of a sequence of actions.

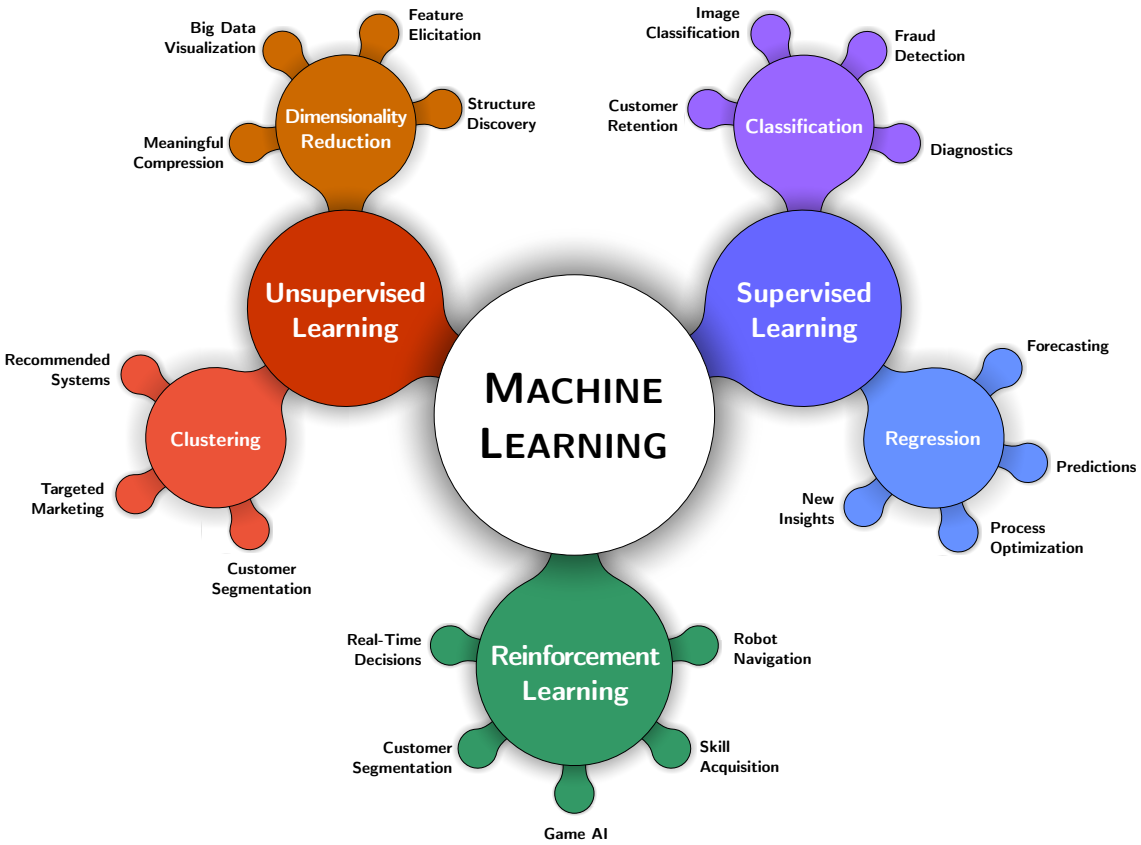


Figure 2.1: Basic categorization of machine learning with example tasks. Machine learning is often categorized into *unsupervised*, *supervised*, and *reinforcement learning*. While there exist more categories and many more tasks with possibly unclear categorization, this figure gives an overview of the large variety of tasks that can be solved with machine learning. The image is adapted from [125] with the (sub-)categories taken from [9].

The reinforcement learning program is optimized by identifying sequences of actions that improve a performance metric based on this outcome.

As we only deal with supervised learning throughout the thesis, we focus on machine learning techniques that perform supervised learning. Supervised learning techniques in general aim to model the *posterior probability* $p(\mathbf{y}^* \mid \mathbf{x})$, where $\mathbf{x} \in \mathbb{R}^n$ is an observation represented as an n-dimensional input vector, and $\mathbf{y}^* \in \mathbb{R}^m$ is the m-dimensional target. With Maximum Likelihood Estimation (MLE) or Maximum-A-Posteriori (MAP) estimation, a function modeling the *posterior probability* may be learned directly from samples of the labeled training dataset $\mathcal{D}^{\text{train}}$. This function may then be applied to unseen input examples to generate m-dimensional prediction outputs.

The two most prominent tasks in supervised learning are *classification* and *regression*. The major difference between these two methods is how the output and target vectors are being represented and interpreted. *Classification* deals with assigning classes, i.e., *discrete* variables, to data samples. Hence, the output of the machine learning methods for clas-

sification is a discrete label out of a set of possible labels. Machine learning techniques that are able to solve classification tasks include k-Nearest Neighbors (k-NN), Linear Discriminant Analysis (LDA), Support Vector Machines (SVMs), Random Forests (RFs), and ANNs. Differently to classification, *regression* assigns *continuous* variables to data samples. Here, the output is a real-valued vector of the output domain of the modeled function. Machine learning techniques for regression tasks include least-squares fitting, Support Vector Regressions (SVRs), Random Regression Forests (RRFs), and ANNs. In this thesis, we focus on ANNs (see Section 2.2) and their extensions to CNNs (see Section 2.4), due to their high performance, recently even outperforming humans in multiple tasks [79], e.g., playing games [127] and image classification [119].

2.1.2 Generalization

Typically, machine learning programs are optimized by minimizing an error on the training data $\mathcal{D}^{\text{train}}$, which is called the *training error*. However, the main goal of a machine learning program is not only to perform well on $\mathcal{D}^{\text{train}}$, but also on new unseen examples that come from the same data distribution as $\mathcal{D}^{\text{train}}$. Hence, we want the error on a new unseen example to be low as well. This expected error on unseen examples is called the *generalization error*, while the ability to perform well on previously unobserved examples is called *generalization*. In combination, two factors determine how well a machine learning algorithm performs: (1) its ability to minimize the *training error*, and (2) its ability to minimize the *generalization gap*, which is a discrepancy between the training error and the generalization error.

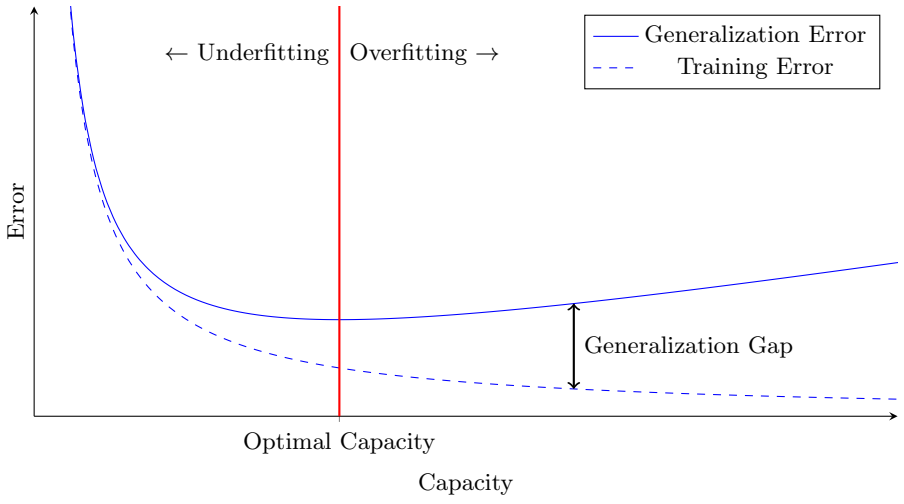
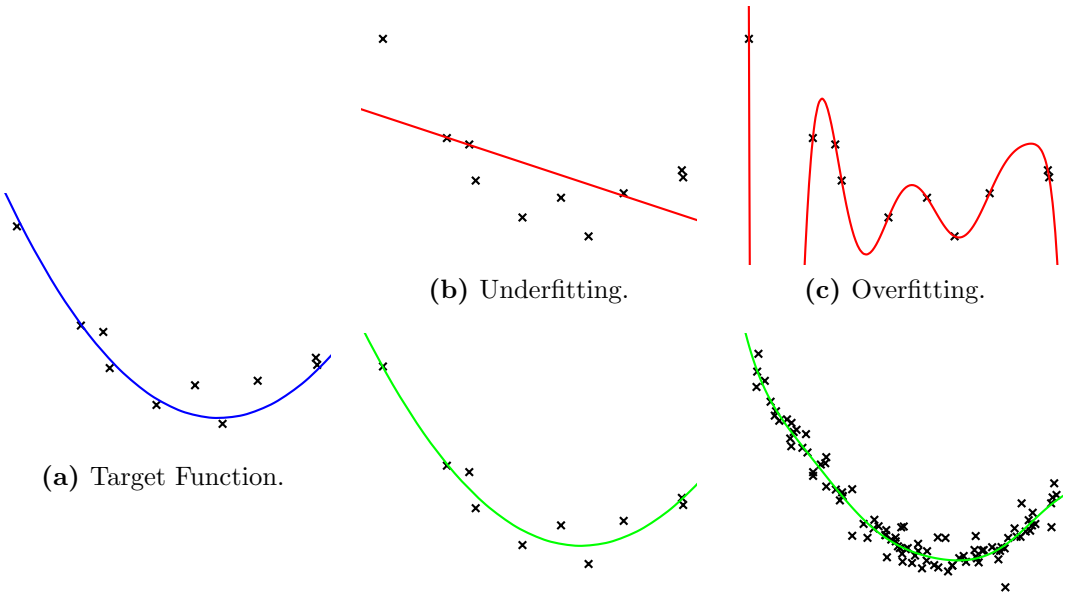


Figure 2.2: Visualization of the training error, generalization error, and the generalization gap. The graph shows the errors with respect to the model capacity. The model with the optimal capacity has the smallest generalization gap and is neither underfitting nor overfitting to the given training data. The figure is adapted from [9].

In machine learning, the *capacity* of a model defines its capability of fitting a large variety of functions. The *capacity* is not only dependent on hyperparameters of the machine learning model, e.g., the network architecture in ANNs, but also on how well the machine learning model is optimized to fit the training data. The relationship between model capacity, training error, and generalization error is depicted in Fig. 2.2. When the capacity of the model is too low, the model is not able to learn the task well enough and both training and generalization errors are high. The model is said to be *underfitting*. When the capacity of the model is too high, the model fits the training data too much, e.g., by memorizing the training data by heart. In this case, the training error is low, but the generalization error is high, leading to a large generalization gap. The model is said to be *overfitting*.



(a) Target Function. **(b)** Underfitting. **(c)** Overfitting. **(d)** Optimal capacity. **(e)** More training data.

Figure 2.3: Visualization of underfitting, overfitting, and optimal capacity. The graphs are adapted from [9] and show data points and best-fitted polynomials of varying degrees. The first graph (a) shows the quadratic target function and some sampled points with added random noise. The graph (b) shows a linear function, which is incapable of fitting the data points appropriately, i.e., it is *underfitting*. A fitted polynomial of degree nine, which is *overfitting* to the data points, is shown in (c). The fitted quadratic function, which has the optimal capacity and the smallest generalization gap for fitting the underlying function, is shown in (d). A very similar performance can also be achieved with a polynomial of degree nine and much more training data, see (e).

Example graphs for fitting polynomials of varying degrees to different amounts of training data are shown in Fig. 2.3. The target function, as well as sampled points with additional noise, are shown in Fig. 2.3a. We see from the graphs that the polynomials with too low or too high degrees are unable to model the underlying function for the given low amount of data points. The model with only a linear function has a large training

error, i.e., is *underfitting* (see Fig. 2.3b). Although the model with a polynomial degree of nine was able to fit the data points exactly, which leads to a training error of zero, the underlying function is not represented well resulting in a high generalization error, i.e., the model is *overfitting* (see Fig. 2.3c). The quadratic model, which has the same degree as the underlying function, is able to fit the underlying function well (see Fig. 2.3d). However, when using more training data, also the polynomial with degree nine was able to fit the underlying function well.

This leads to two observations: First, among competing hypotheses that explain known observations equally well, one should choose the simplest one (Occam's razor). Second, the generalization gap is bounded from above by a quantity that grows as the model capacity grows, but shrinks as the number of training examples increases [149]. Combining these two observations, we should (1) try to fit the simplest model that can capture the data, as well as (2) try to use as much data as possible for training.

Note that, due to the difficult optimization of machine learning methods (see Section 2.3), which often fails to find the globally optimal model parameters, a typical choice is to combine large amounts of training data with more complex models (see Section 2.4), where adequate model parameters can be found more easily [47, 50].

2.1.3 Training, Validation, and Test Sets

In practice, we are not able to calculate the exact generalization error, i.e., the expected error on all unseen examples, because exact calculation would require to know all possible unseen examples in advance. Hence, the generalization error is typically approximated with the *test error*, which is calculated on data from a fixed *test set* $\mathcal{D}^{\text{test}}$. The data from the *test set* is required to be disjoint from the training data $\mathcal{D}^{\text{train}}$, while both training and test data need to be independent and identically distributed.

To achieve the best performance for a model, its optimal capacity needs to be determined. As discussed in the previous section, the model capacity is not only dependent on the hyperparameters of the model, but also on the optimization of the model itself, i.e., how well the training algorithm managed to optimize the model to fit the training data. Thus, we need to determine the hyperparameters of both the model and the training algorithm that lead to the lowest *generalization error*. However, we may not use the test set for determining these hyperparameters, as determining these hyperparameters is part of the training. If we would use the test set for tuning any component of the machine learning model and training algorithm, we would distort the *test error* and it would consequently be a bad approximation of the *generalization error*. For this reason, the hyperparameters of the model and training algorithm are determined with the *validation set* $\mathcal{D}^{\text{val}}$, which is usually taken out of the training set $\mathcal{D}^{\text{train}}$. The remaining data from the training set is used to update the model parameters to minimize the *training error*, while the *validation set* is used to calculate the *validation error*. Based on the *validation error* of various models trained with different hyperparameters, the final set of hyperparameters is chosen.

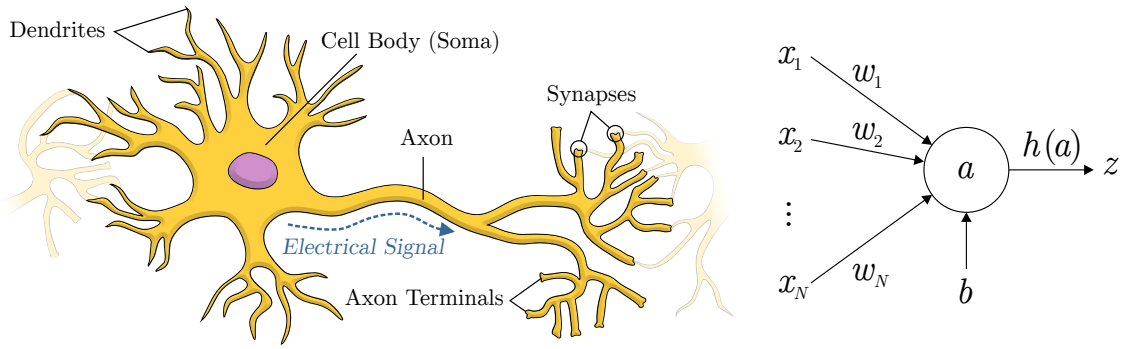
A drawback when splitting the training data into training and validation set is that the data of the validation set is used only to determine the hyperparameters, but not to optimize the parameters of the machine learning model. However, as machine learning methods learn to identify patterns from data, training a model with more data usually leads to better performance. To mitigate that the validation data was not used for optimizing the model parameters, after determining the final set of hyperparameters, the model may be retrained with both training and validation data. A drawback of this procedure is that the *validation error* cannot be tracked anymore.

As the training set is typically small, the choice of the validation set is critical and may not be representative of the whole training set. This could lead to suboptimal hyperparameters of the machine learning model. For this reason, the hyperparameters are often determined via k-fold *cross-validation*. In k-fold cross-validation, the training set is split into k equally sized sets. One set is then declared as the validation set, while the remaining sets are used as the training set. This procedure is repeated until every set has been used once for validation. This way, every training data sample will appear exactly once in the validation set, which allows a better estimation of the hyperparameters of the machine learning method. The drawback of k-fold cross-validation is that k models need to be trained for estimating the overall validation error.

2.2 Artificial Neural Networks

ANNs are currently the most dominantly used machine learning technique. In general, an ANN with parameters $\boldsymbol{\theta}$ maps an n-dimensional data sample $\mathbf{x} \in \mathbb{R}^n$ to an m-dimensional output $\hat{\mathbf{y}} \in \mathbb{R}^m$, i.e., $\hat{\mathbf{y}} = f(\mathbf{x}; \boldsymbol{\theta})$ with $f : \mathbb{R}^n \to \mathbb{R}^m$. While the mathematical foundations of neural networks date back to the 1940s [93], they became practically usable with the invention of the backpropagation algorithm [118], which allowed efficient training of ANNs. While ANNs were popular in the 1990s, in the early 2000s lot of research was focusing more on other machine learning techniques, e.g., SVMs [23] and RFs [10], due to their back then superior performance as compared to ANNs. However, architectural improvements in ANNs like convolution layers, pooling layers, the Rectified Linear Unit (ReLU) activation function, and better optimizers, as well as large annotated datasets and faster hardware, made training *deep* ANNs with lots of layers feasible.

The current deep learning boom started in 2012 with the AlexNet from Krizhevsky et al. [78]. The authors used modern Graphics Processing Units (GPUs) to train a deep CNN, which was able to beat other machine learning techniques by a large margin. Their AlexNet achieved an error of 15.3% for image classification in the ImageNet Large Scale Visual Recognition Challenge 2012 (ILSVRC2012), which was at that time more than 10.8 percentage points lower than the error of the second-best.



(a) A biological neuron. Adapted from [7].

(b) An artificial neuron [9].

Figure 2.4: Schematic representations of a biological and an artificial neuron. The biological neuron receives an electrical signal from previous neurons at its dendrites. When enough electrical signal has been accumulated within the neuron, it is activated and fires an electrical signal to its axon terminals, which serve as input to other neurons. Similarly, the artificial neuron receives a signal from previous neurons at its N inputs x_i, which are multiplied with the weighting factors w_i and added with a bias b. The accumulated value a is given to the activation function h, which calculates the activation value of the neuron that is used as input to other connected neurons.

2.2.1 Artificial Neuron

The basic building blocks of every ANN are artificial neurons, which are loosely inspired by biological neurons (see Fig. 2.4). Overly simplified, a biological neuron works as follows [7]: A neuron is connected to others via synapses, where dendrites and axon terminals of different neurons are joined. At the dendrites, the neuron receives electrical signals from the axon terminals of previous neurons. After a certain amount of electrical signal has accumulated inside the cell body, the neuron is *activated* and starts to *fire* electrical signals with a certain frequency out of its axon. The electrical signal at the axon terminals is then transferred to the next neurons' dendrites, and so on. Numerous connected biological neurons transfer electrical signals to form large biological neural networks.

An artificial neuron follows the same principles. It receives k input signals $x_1 \dots x_k \in \mathbb{R}$ from other neurons. These input signals are then multiplied with the corresponding weights $w_1 \dots w_k \in \mathbb{R}$. The weighted input signals plus an additional bias $b \in \mathbb{R}$ are summed up to the accumulated value $a \in \mathbb{R}$. Based on the value of a, a non-linear activation function $h : \mathbb{R} \to \mathbb{R}$ generates the output signal $z \in \mathbb{R}$ of the neuron. The non-linear activation function $h(a)$ is important to increase the modeling capacity of the artificial neuron, since, without this non-linearity, the neuron would only be able to model linear functions. Moreover, the non-linear activation function has a biological interpretation. Similar to biological neurons, dependent on the value of the accumulated input signals a and the non-linear activation function $h(a)$, an artificial neuron is either *activated* and sends a signal to its output z, or it is not. The output signal z serves either as an input for successive neurons or as an output of the whole ANN.

In mathematical terms, the accumulated value $a \in \mathbb{R}$ and the output value $z \in \mathbb{R}$ of an artificial neuron are defined as

$$a = \sum_{i=1}^{k} w_i x_i + b, \tag{2.1}$$

$$z = h(a), \tag{2.2}$$

respectively, with $x_1 \dots x_k \in \mathbb{R}$ being the k inputs, $w_1 \dots w_k \in \mathbb{R}$ the k weights, $b \in \mathbb{R}$ the bias, and $h : \mathbb{R} \to \mathbb{R}$ the activation function of the neuron.

Note that there exist other notations that define neurons as vectors and weights as matrices, thus, using matrix multiplications instead of summing scalars.

2.2.2 Layered Neural Networks

While a single artificial neuron alone is limited in its modeling capacity, multiple artificial neurons may be connected to form large artificial neural networks, i.e., ANNs, which are able to model much more complex functions. The neurons of an ANN are divided into input neurons x_i, output neurons $\hat{y}_i$, and hidden neurons z_i. The input neurons do not receive signals from other neurons, but their values are set to the values of the input data sample $\mathbf{x} \in \mathbb{R}^n$. The output neurons do not propagate signals to other neurons but serve as the predicted output of the network, i.e., $\hat{\mathbf{y}} \in \mathbb{R}^m$. All other neurons that are neither input nor output neurons are called hidden neurons.

Depending on the structure of the neuron connections, the networks can be grouped into different network types. A Feed-forward Neural Network (FNN) is an ANN without loops, i.e., a neuron's output is neither used as its input directly nor indirectly via connections over other neurons. Accordingly, if the neuron connections form loops, the network is called a Recurrent Neural Network (RNN). An advantageous property originating from such loops is that RNNs are able to encode temporal information by memorizing previous data. However, as RNNs work best with temporal data, we are not using RNNs throughout the thesis.

To describe the specific architecture of ANNs more efficiently, the neurons of the network are typically grouped into layers. As visualized in Fig. 2.5, in these so-called Layered Neural Networks (LNNs), neurons of the same layer receive their inputs only from neurons of the previous layer, and propagate their outputs only to the neurons of the next layer. As a layered architecture does not restrict the computational capabilities of FNNs, while making their description much easier, almost all modern literature dealing with neural networks uses LNNs.

The general structure of LNNs is as follows: The first layer is called the *input layer*, which consists of the n input neurons. The values $x_1 \dots x_n \in \mathbb{R}$ are set to the input data sample $\mathbf{x} \in \mathbb{R}^n$. The last layer is called the *output layer*, which represents the m output neurons. The values $\hat{y}_1 \dots \hat{y}_m \in \mathbb{R}$ represent the network's predicted output $\hat{\mathbf{y}} \in \mathbb{R}^m$.

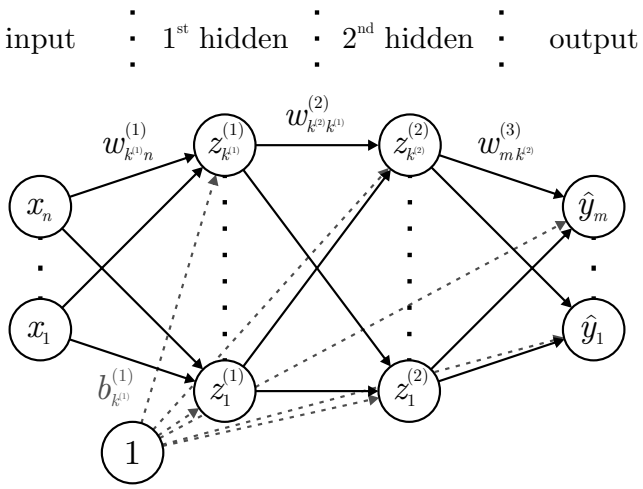


Figure 2.5: Schematic representations of a layered neural network. Neurons are shown as circle, while input neurons are denoted as x, hidden neurons as z, and output neurons as $\hat{y}$. Neuron connections are shown as black arrows, while each arrow has a corresponding weight w. The biases b are shown as dashed arrows. The images are adapted from [9, 47].

The intermediate layers are called *hidden layers*. For each hidden layer l (as well as the output layer), we calculate the $k^{(l)}$ accumulated values $a^{(l)}_1 \dots a^{(l)}_{k^{(l)}} \in \mathbb{R}$ and neuron outputs $z^{(l)}_1 \dots z^{(l)}_{k^{(l)}} \in \mathbb{R}$. The j^{th} accumulated value and neuron output of the l^{th} hidden layer are defined as,

$$a^{(l)}_j = \sum_{i=1}^{k^{(l-1)}} w^{(l)}_{ji} z^{(l-1)}_i + b^{(l)}_j, \tag{2.3}$$

$$z^{(l)}_j = h^{(l)}(a^{(l)}_j). \tag{2.4}$$

The i^{th} neuron output of the previous layer $l-1$ is denoted as $z^{(l-1)}_i$ and is connected with the j^{th} neuron of layer l via the weight $w^{(l)}_{ji}$; $b^{(l)}_j$ denotes the bias of the j^{th} neuron of layer l; $h^{(l)}$ denotes the activation function of layer l. The network parameters $\boldsymbol{\theta}$ are composed of all weights and biases. Following Eqs. (2.3) and (2.4), for a network with L layers the input layer can be considered as the 0^{th} hidden layer, i.e., $z^{(0)}_j = x_j$ with $k^{(0)} = n$, while the output layer can be considered as the L^{th} hidden layer, i.e., $z^{(L)}_j = \hat{y}_j$ with $k^{(L)} = m$.

LNNs also introduce some new terminology to neural networks. The number of layers of a neural network is called the network *depth* L, while the number of neurons within a layer is considered the network *width*. Networks with a small depth are called *shallow*, while networks with a larger depth are called *deep*. The first successful deep neural network, which started the deep learning boom, was the AlexNet [78]. While the AlexNet consisted of eight layers, recent improvements in optimizing neural networks allowed the training of very deep networks with more than a thousand layers [52].

It has been shown that neural networks with a single hidden layer and a large enough *width*, i.e., number of hidden neurons, can approximate any continuous function to any desired precision [57]. However, in practice, increasing the network *depth* instead of the *width* has shown to be more successful. In *deep* networks, the subsequent non-linear activation functions, as well as parameter sharing within a layer (see the convolutional operation in Section 2.4), drastically increase the modeling capacity of deep neural networks, while keeping the number of network parameters reasonable.

2.2.3 Activation Functions

Neuron activation functions incorporate non-linearities into artificial neurons, which is important for ANNs to achieve a high modeling capacity. Many works focused on proposing and analyzing activation functions, each with different properties, advantages, and disadvantages. As discussing this vast amount of different activation functions in detail is beyond the scope of this thesis, we will only list various commonly used activation functions that we also use in our networks. For each discussed activation function we will show the function definition, function graph, and derivative graph.

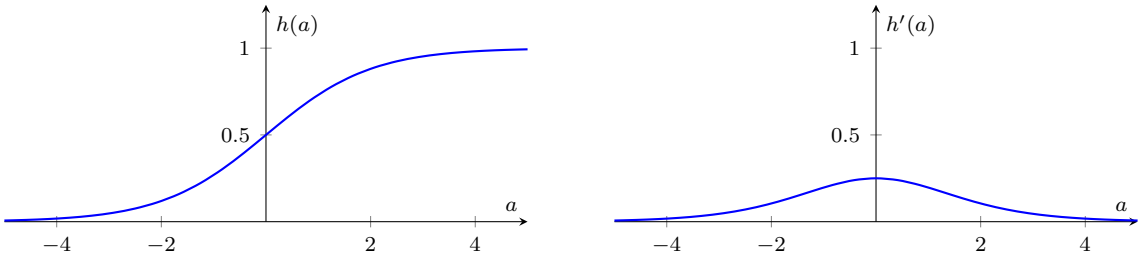


Figure 2.6: Sigmoid activation function. The plot on the left shows the function graph, the plot on the right its derivative.

The first discussed activation function, often used in earlier neural networks, is the sigmoid activation function (see Fig. 2.6). It is defined as

$$\sigma(a) = \frac{1}{1 + e^{-a}}. \tag{2.5}$$

The sigmoid function can be seen as a smoothed and continuous variant of the binary threshold activation function. Hence, in contrast to the binary threshold activation function, $\sigma(a)$ also has a smooth and continuous gradient for the whole input domain. The values of $\sigma(a)$ are bound between $(0, 1)$. However, $\sigma(a)$ is not symmetric around 0, i.e., $\sigma(0) = 0.5$. When the network layer weights are initialized with small random values, the output values of deeper layers would steadily increase, at some point reaching values close to 1. This leads to saturation in deeper layers resulting in little gradient information, which is disadvantageous for training neural networks via Backpropagation (BP) (see Section 2.3.2) and also known as the vanishing gradient problem [56].

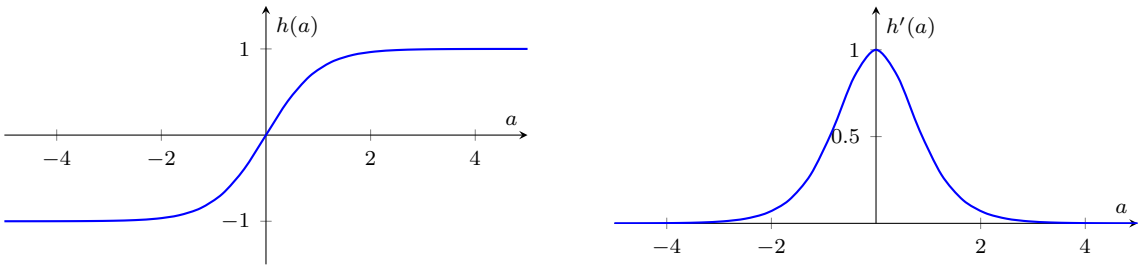


Figure 2.7: Hyperbolic tangent activation function. The plot on the left shows the function graph, the plot on the right its derivative.

Another activation function related to $\sigma(a)$ is the hyperbolic tangent (see Fig. 2.7), which is defined as

$$\tanh(a) = \frac{e^a - e^{-a}}{e^a + e^{-a}}. \tag{2.6}$$

Note that $\tanh(a)$ is a rescaled version of $\sigma(a)$, i.e., $\tanh(a) = 2\sigma(2a) - 1$. Its values are bound between $(-1, 1)$. As $\tanh(0) = 0$, also deeper layers do not saturate when the network weights are initialized with small values. This leads to better network training behavior as compared to $\sigma(a)$. However, supported by biological interpretation, artificial neural networks are supposed to perform better when neurons are only sparsely activated, i.e., $h(a) \approx 0$ for many neurons. The $\tanh(a)$ function is only deactivated when $a = 0$, which is hard to fulfill without further regularization.

Another disadvantagous property of the sigmoidal activation functions $\sigma(a)$ and $\tanh(a)$ is that the gradient is very small when the functions are saturated, i.e., $h'(a) \approx 0$ when $a \ll 0$ or $a \gg 0$. When using gradient-based optimization techniques, this leads to slow convergence.

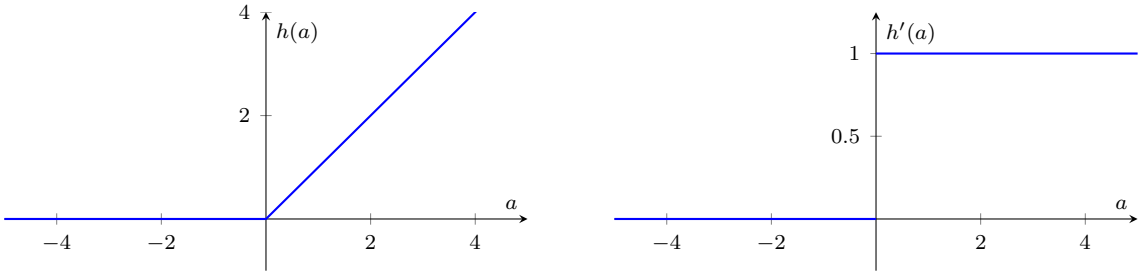


Figure 2.8: ReLU activation function. The plot on the left shows the function graph, the plot on the right its derivative.

An activation function that tackles drawbacks of sigmoidal activation functions is the ReLU function (see Fig. 2.8), which is considered to have contributed greatly to the recent success of deep neural networks. Being already introduced for shallow ANNs at the end of the 1990s [49], Glorot et al. [46], Nair and Hinton [97] showed that the ReLU activation function has many desired properties for training deep ANNs with many layers. The ReLU

function is defined as follows:

$$\text{relu}(a) = \max(0, a). \tag{2.7}$$

For all values $a \leq 0$, $\text{relu}(a) = 0$, hence, ReLU allows networks with sparsely activated neurons. While ReLU is bounded from below by 0, its maximum value is unbounded, which is biologically less plausible. However, this leads to good training characteristics for neural networks with many layers. In contrast to sigmoidal activation functions, when the network layer weights are initialized with values close to 0, the output of ReLU does not saturate when $a \gg 0$ and always provides strong gradient information.

However, when the layer weights are not initialized carefully or the training parameters are set incorrectly, there could exist neurons within the network that are deactivated, i.e., $\text{relu}(a) = 0$ for every training example. As there is also no gradient information in such cases (see Fig. 2.8), the output of the neurons will never change. The neuron will stay deactivated, reducing the overall computational capacity of the ANN.

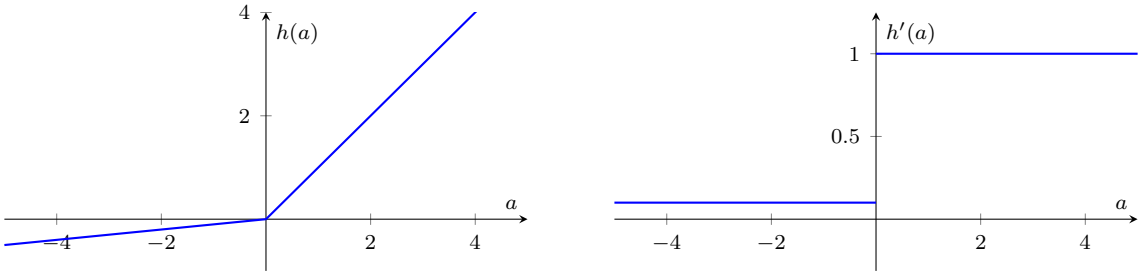


Figure 2.9: Leaky ReLU activation function. The plot on the left shows the function graph, the plot on the right its derivative.

An adaption of the ReLU activation function, where always deactivated neurons do not exist, is Leaky ReLU [90] (see Fig. 2.9). It is defined as

$$\text{lrelu}(a) = \max(l \cdot a, a), \tag{2.8}$$

with the negative slope $l < 1$. Although Leaky ReLU is unbounded with output values between $(-\infty, \infty)$, it has shown to have similar training characteristics as ReLU, without the issue of possibly always deactivated neurons.

While recently there have been proposed many different activation functions with various improvements or drawbacks for learning deep neural networks (e.g., PReLU [51], ELU [19], SELU [73]), deep ANNs typically work well when using ReLU or Leaky ReLU.

2.2.4 Loss Functions

ANNs with parameters $\boldsymbol{\theta}$ solve a specific task by mapping an n-dimensional data sample $\mathbf{x} \in \mathbb{R}^n$ to an m-dimensional target function $f(\mathbf{x}; \boldsymbol{\theta})$ with $f : \mathbb{R}^n \to \mathbb{R}^m$. To determine how well the network models $f(\mathbf{x}; \boldsymbol{\theta})$, a function that measures the performance has to be defined. This function is called the *loss function*.

Depending on the task to solve, a suitable loss function has to be specified. In literature, many different loss functions for many different tasks have been proposed. Since in this thesis, we only consider supervised learning, our utilized loss functions measure the error between the i^{th} data sample's ground-truth $\overset{*}{\mathbf{y}}^{(i)}$ and the network's prediction $\hat{\mathbf{y}}^{(i)} = f(\mathbf{x}^{(i)}; \boldsymbol{\theta})$. We employ both regression and classification tasks.

In regression tasks, the network is optimized to model a continuous function that maps an input $\mathbf{x}^{(i)} \in \mathbb{R}^n$ to its ground-truth annotation $\overset{*}{\mathbf{y}}^{(i)} \in \mathbb{R}^m$. During optimization, the network learns parameters $\boldsymbol{\theta}$ that minimize the error between $\overset{*}{\mathbf{y}}^{(i)}$ and $\hat{\mathbf{y}}^{(i)}$. A common loss function for regression that minimizes this error is the L_2 loss, i.e.,

$$L_2^{(i)}(\mathbf{x}^{(i)}, \overset{*}{\mathbf{y}}^{(i)}; \boldsymbol{\theta}) = \| f(\mathbf{x}^{(i)}; \boldsymbol{\theta}) - \overset{*}{\mathbf{y}}^{(i)} \|_2^2, \tag{2.9}$$

where $\|\cdot\|_2^2$ is the squared L_2 norm.

In classification tasks, the network is optimized to determine the class c of the input $\mathbf{x}^{(i)} \in \mathbb{R}^n$ out of k available classes. Typically, classification networks are optimized to model a function that maps an input $\mathbf{x} \in \mathbb{R}^n$ to a k-dimensional one-hot encoded class vector $\overset{*}{\mathbf{y}} \in \mathbb{N}^k$, i.e., $\overset{*}{y}_i = 1$ for $i = c$ and $\overset{*}{y}_i = 0$ for $i \neq c$. During optimization, the network learns parameters $\boldsymbol{\theta}$, which minimize the error between the network prediction $\hat{\mathbf{y}} = f(\mathbf{x}; \boldsymbol{\theta})$ representing a k-dimensional probability distribution, and the one-hot encoded target distribution $\overset{*}{\mathbf{y}}$. A common loss function for classification is the *cross-entropy* loss, i.e.,

$$L_{\text{CE}}^{(i)}(\mathbf{x}^{(i)}, \overset{*}{\mathbf{y}}^{(i)}; \boldsymbol{\theta}) = -\overset{*}{\mathbf{y}}^{(i)} \cdot \log f(\mathbf{x}^{(i)}; \boldsymbol{\theta}), \tag{2.10}$$

where $\cdot$ denotes the dot product of two vectors and the log function is applied element-wise to each entry of the k-dimensional network output $f(\mathbf{x}^{(i)}; \boldsymbol{\theta})$. As segmentation in images can be considered as pixel-wise classification, networks for segmentation often also use the cross-entropy loss and apply it per pixel.

The aforementioned loss functions define the loss for a single training example i. However, as we do not only want to calculate the loss for a single training sample but for the whole dataset $\mathcal{D}$ with N entries, i.e., $\{\langle \mathbf{x}^{(1)}, \overset{*}{\mathbf{y}}^{(1)} \rangle \dots \langle \mathbf{x}^{(N)}, \overset{*}{\mathbf{y}}^{(N)} \rangle\} \in \mathcal{D}$, we write

$$L(\mathcal{D}; \boldsymbol{\theta}) = \frac{1}{N} \sum_{i=1}^{N} L^{(i)}(\mathbf{x}^{(i)}, \overset{*}{\mathbf{y}}^{(i)}; \boldsymbol{\theta}). \tag{2.11}$$

In the following, when we address the loss function, for convenience we may omit the dataset $\mathcal{D}$, the sample index (i), as well as the sample input $\mathbf{x}^{(i)}$ and target $\overset{*}{\mathbf{y}}^{(i)}$, and just write $L^{(i)}(\boldsymbol{\theta})$ or $L(\boldsymbol{\theta})$, respectively.

2.3 Neural Networks Optimization

For the network to learn the target function, the parameters $\boldsymbol{\theta}$ of the network that minimize the loss function have to be determined. In (local) minima of the loss function $L(\boldsymbol{\theta})$

the gradient with respect to the parameters $\boldsymbol{\theta}$ is 0, i.e., $\nabla_{\boldsymbol{\theta}} L(\boldsymbol{\theta}) \widehat{=} 0$. The gradient is a vector-valued function $\nabla_{\boldsymbol{\theta}} L(\boldsymbol{\theta}) : \mathbb{R}^n \to \mathbb{R}^n$ whose value at a point $\boldsymbol{\theta}$ is the vector whose components are the partial derivatives for the individual network parameters of L at θ:

$$\nabla_{\boldsymbol{\theta}} L(\boldsymbol{\theta}) = \begin{bmatrix} \frac{\partial L}{\partial \theta_1}(\boldsymbol{\theta}) \\ \vdots \\ \frac{\partial L}{\partial \theta_n}(\boldsymbol{\theta}) \end{bmatrix}. \tag{2.12}$$

Here, $\theta_1 \dots \theta_n$ denote all learnable network parameters, which are typically the weights $\mathbf{w}$ and biases $\mathbf{b}$ of the all network layers.

Due to the non-linear activation functions of hidden neurons resulting in non-linear network outputs with non-convex loss functions, an analytic solution of the minimum of the loss function is intractable. Hence, neural networks are optimized via gradient-based optimization techniques, which update step-by-step the network parameters to decrease the value of the loss function until reaching a minimum where $\nabla_{\boldsymbol{\theta}} L(\boldsymbol{\theta}) \widehat{=} 0$. Generally, in one Gradient Descent (GD) step, the network parameters are moved towards the direction of their negative gradient, as this determines the largest decrease of the loss value. One step is defined as

$$\begin{aligned} \mathbf{g}^{(\tau+1)} &= \nabla_{\boldsymbol{\theta}} L(\boldsymbol{\theta}^{(\tau)}), \\ \boldsymbol{\theta}^{(\tau+1)} &= \boldsymbol{\theta}^{(\tau)} - \gamma\, \mathbf{g}^{(\tau+1)}. \end{aligned} \tag{2.13}$$

where the weight update $\mathbf{g}^{(\tau+1)}$ at timestep $\tau+1$ is set to $\nabla_{\boldsymbol{\theta}} L(\boldsymbol{\theta}^{(\tau)})$, which is the gradient of $L(\boldsymbol{\theta}^{(\tau)})$ with respect to $\boldsymbol{\theta}^{(\tau)}$. The updated network parameters $\boldsymbol{\theta}^{(\tau+1)}$ at timestep $\tau+1$ are set to previous parameters $\boldsymbol{\theta}^{(\tau)}$ at timestep τ, minus the current weight update $\mathbf{g}^{(\tau+1)}$ multiplied with a factor γ, which is called the *step size* or *learning rate*.

As it usually takes thousands of gradient descent steps for the loss function to converge to a minimum, while the gradient needs to be recalculated after each step, evaluating the gradient for the whole dataset of N training examples is impracticable. However, loss functions typically have the form $L(\boldsymbol{\theta}) = \frac{1}{N} \sum_{i=1}^{N} L^{(i)}(\boldsymbol{\theta})$, where $L^{(i)}(\boldsymbol{\theta})$ is the error of the i^{th} sample. Hence, we can approximate the gradient of the whole dataset $\mathcal{D}$ of size N with the gradient of a randomly drawn subsample or *mini-batch* $\mathcal{B} \subset \mathcal{D}$ of size N', i.e.,

$$\nabla_{\boldsymbol{\theta}} L(\boldsymbol{\theta}) \approx \frac{1}{N'} \sum_{i=1}^{N'} \nabla_{\boldsymbol{\theta}} L^{(i)}(\boldsymbol{\theta}). \tag{2.14}$$

For every calculation of this approximated gradient, a different *mini-batch* $\mathcal{B}$ is randomly drawn from the dataset $\mathcal{D}$. When using a mini-batch $\mathcal{B}$ instead of the whole dataset $\mathcal{D}$ to calculate the gradient and update the network parameters, the optimization technique is

called Stochastic Gradient Descent (SGD). The update step is defined as

$$\begin{aligned} \mathbf{g}^{(\tau+1)} &= \frac{1}{N'} \sum_{i=1}^{N'} \nabla_{\boldsymbol{\theta}} L^{(i)}(\boldsymbol{\theta}^{(\tau)}), \\ \boldsymbol{\theta}^{(\tau+1)} &= \boldsymbol{\theta}^{(\tau)} - \gamma\, \mathbf{g}^{(\tau+1)}. \end{aligned} \tag{2.15}$$

SGD has various advantages as compared to GD. In contrast to GD that needs to calculate the gradient on the whole training dataset, the complexity of a gradient update step in SGD is independent of the number of training images N, but only depends on the size of the mini-batch N'. Moreover, as each mini-batch contains different training samples, gradients of different mini-batches point towards different local minima. Thus, during optimization, the chances of getting stuck in a local minimum are reduced in SGD. As a drawback, the noisy gradients and parameter updates of SGD may hinder convergence towards a minimum, especially when training is already advanced. A technique to prevent that is to decay the learning rate during training, e.g., to start with a large learning rate and decrease it after a while. In our proposed methods, we often start training with a fixed learning rate and after a certain number of iterations, we reduce the learning rate by a fixed factor and continue training until convergence.

2.3.1 Variants of Stochastic Gradient Descent Optimizers

Although SGD is capable of finding (local) minima of the loss function, it is typically not applied without modifications, as it has several drawbacks. First, due to the use of mini-batches, subsequent gradients may be noisy and point towards opposite directions, which results in subsequent parameter updates that cancel themselves. Furthermore, in plateaus of the loss function without a steep gradient direction, learning may be very slow.

A method to overcome canceling parameter updates and plateaus is to incorporate momentum into the parameter update step. Momentum can be interpreted as giving the parameter updates a mass by accumulating an exponentially decaying moving average of past gradients. Subsequent parameter updates are smoothed to go towards a dominant direction while being prevented to cancel themselves completely. Thus, the parameter update step depends on how well previous gradients are aligned. The update step for SGD with momentum is defined as

$$\begin{aligned} \mathbf{g}^{(\tau+1)} &= \frac{1}{N'} \sum_{i=1}^{N'} \nabla_{\boldsymbol{\theta}} L^{(i)}(\boldsymbol{\theta}^{(\tau)}), \\ \mathbf{v}^{(\tau+1)} &= \alpha\, \mathbf{v}^{(\tau)} - \gamma\, \mathbf{g}^{(\tau+1)}, \\ \boldsymbol{\theta}^{(\tau+1)} &= \boldsymbol{\theta}^{(\tau)} + \mathbf{v}^{(\tau+1)}. \end{aligned} \tag{2.16}$$

Different to Eq. (2.15), in Eq. (2.16) the parameter update is modified to be a *velocity*, denoted as $\mathbf{v}^{(\tau+1)}$ and $\mathbf{v}^{(\tau)}$ at timestep $\tau+1$ and τ, respectively. The momentum parameter

α handles how much of the previous parameter update $\mathbf{v}^{(\tau)}$ is incorporated for the current parameter update $\mathbf{v}^{(\tau+1)}$, hence, modeling an exponentially decaying moving average of past updates.

A modification of the SGD with momentum is to incorporate Nesterov's accelerated gradient method [98]. With this method, the parameter update will not be evaluated at the location of the current parameters, but after taking a step towards the current gradient direction. Due to this correction factor of the gradient, SGD with Nesterov's momentum shows faster convergence properties as compared to using normal momentum. The update step for SGD with Nesterov's momentum is defined as

$$\begin{aligned} \mathbf{g}^{(\tau+1)} &= \frac{1}{N'} \sum_{i=1}^{N'} \nabla_{\boldsymbol{\theta}} L^{(i)}(\boldsymbol{\theta}^{(\tau)} + \alpha\, \mathbf{v}^{(\tau)}) \\ \mathbf{v}^{(\tau+1)} &= \alpha\, \mathbf{v}^{(\tau)} - \gamma\, \mathbf{g}^{(\tau+1)} \\ \boldsymbol{\theta}^{(\tau+1)} &= \boldsymbol{\theta}^{(\tau)} + \mathbf{v}^{(\tau+1)} \end{aligned} \tag{2.17}$$

Although SGD with (Nesterov's) momentum may be used to locate robust minima of the loss function of a large variety of tasks, it has some disadvantages. For example, in SGD with (Nesterov's) momentum it is not possible to find *the* best learning rate that fits any given task. Different learning rates have to be carefully examined, to enable the optimizer to find a satisfactory minimum that leads to good performance for the given task. Moreover, each network parameter has the same learning rate, which leads to large steps for parameters with steep gradients, while parameters with gradients on plateaus may be updated with only small steps, possibly never reaching an optimal value.

Optimizers with adaptive learning rates circumvent these problems by adapting the step widths for each parameter individually, based on their previous updates. This allows lowering the step widths for parameters with too large gradients, which would otherwise cause the loss function to diverge, while it also allows increasing the step widths for parameters with small gradients on plateaus, which would otherwise not converge satisfactorily. Multiple optimizers with adaptive learning rates have been proposed e.g., Adagrad [29], RMSprop [55], Adam [71]. As the principle of these optimizers is very similar, we will only describe Adam [71] in more detail, as we also use it in this thesis and it additionally removes some drawbacks of the other optimizers.

Similar to SGD with momentum, Adam accumulates an exponentially decaying moving average over past gradient updates

$$\begin{aligned} \mathbf{g}^{(\tau+1)} &= \frac{1}{N'} \sum_{i=1}^{N'} \nabla_{\boldsymbol{\theta}} L^{(i)}(\boldsymbol{\theta}^{(\tau)}), \\ \mathbf{m}_1^{(\tau+1)} &= \beta_1\, \mathbf{m}_1^{(\tau)} + (1-\beta_1)\, \mathbf{g}^{(\tau+1)}, \\ \mathbf{m}_2^{(\tau+1)} &= \beta_2\, \mathbf{m}_2^{(\tau)} + (1-\beta_2)\, \mathbf{g}^{(\tau+1)} \odot \mathbf{g}^{(\tau+1)}, \end{aligned} \tag{2.18}$$

while $\mathbf{m}_1^{(\tau)}$ and $\mathbf{m}_2^{(\tau)}$ are moving averages of the first and second moments of the gradients at timestep τ, β_1 and β_2 are the decay rates for the moment estimates and $\odot$ is the hadamard or element-wise product. As the moments are biased towards their initialization, Kingma and Ba [71] proposed to use their bias-corrected estimates

$$\begin{aligned} \widehat{\mathbf{m}}_1^{(\tau+1)} &= \frac{\mathbf{m}_1^{(\tau+1)}}{1-\beta_1^\tau}, \\ \widehat{\mathbf{m}}_2^{(\tau+1)} &= \frac{\mathbf{m}_2^{(\tau+1)}}{1-\beta_2^\tau}. \end{aligned} \tag{2.19}$$

Note that in Eq. (2.19) the τ in the numerators β_1^τ and β_2^τ are not β_1 and β_2 at timestep τ, but denote the exponent. The final parameter update step for Adam is then defined as

$$\boldsymbol{\theta}^{(\tau+1)} = \boldsymbol{\theta}^{(\tau)} - \gamma \frac{\widehat{\mathbf{m}}_1^{(\tau+1)}}{\sqrt{\widehat{\mathbf{m}}_2^{(\tau+1)}} + \epsilon}, \tag{2.20}$$

where γ is the learning rate, and ϵ is needed to prevent divisions by zero.

2.3.2 Backpropagation

All gradient descent based optimizers described in the previous section need to calculate the gradient of the loss function with respect to the individual parameters, i.e., $\nabla_{\boldsymbol{\theta}} L(\boldsymbol{\theta})$ (see Eq. (2.12)), to update the network parameters. The gradients of the individual parameters are calculated with the BP algorithm by applying the chain rule of calculus. While the principles of BP date back to the 1960s, Werbos [156] mentioned to apply it to neural networks, while Rumelhart et al. [118] showed experimentally that BP can generate useful internal representations of data in hidden layers of FNNs.

The BP algorithm calculates the gradient values with two passes through the network: The *forward pass* proceeds from first to the last layer and calculates the outputs of all network layers. The *backward pass* proceeds from last to the first layer and uses an error signal based on the network loss to calculate the derivatives of the individual network parameters.

For the *forward pass*, we calculate the outputs for layer l as defined in Eqs. (2.3) and (2.4), i.e.,

$$a_j^{(l)} = \sum_{i=1}^{k^{(l-1)}} w_{ji}^{(l)} z_i^{(l-1)} + b_j^{(l)}, \tag{2.21}$$

$$z_j^{(l)} = h^{(l)}(a_j^{(l)}). \tag{2.22}$$

From these recursive equations we can see that only the outputs of layer $l-1$ are required for calculating the outputs of layer l. Thus, starting from the input layer, we proceed

forward from each layer to the next one until reaching the final output layer, which serves as the overall network output with which we can calculate the loss function L.

For the *backward pass*, we calculate the partial derivatives of the loss L with respect to each network weight and bias. From Eqs. (2.21) and (2.22) we can derive

$$\frac{\partial L}{\partial w_{ji}^{(l)}} = \frac{\partial L}{\partial a_j^{(l)}} z_i^{(l-1)} \quad \text{and} \quad \frac{\partial L}{\partial b_j^{(l)}} = \frac{\partial L}{\partial a_j^{(l)}} \cdot 1. \tag{2.23}$$

The common term of both equations is called the *error signal* $\delta_j^{(l)}$, which is defined as

$$\delta_j^{(l)} = \frac{\partial L}{\partial a_j^{(l)}} = \left(\sum_{k=1}^{k^{(l+1)}} \delta_k^{(l+1)} w_{kj}^{(l)} \right) h'^{(l)}(a_j^{(l)}). \tag{2.24}$$

With this recursive equation only the error signals from layer $l+1$ are required for calculating the error signal of layer l. Hence, to apply this equation, we first evaluate the error signal of the loss function. Then, we proceed *backward* from each layer to the previous one and calculate the layer's error signals and update its weights and biases, until reaching the input layer.

2.3.3 Weight Initialization

For applying BP to determine the network parameters that solve the task optimally, the network parameters $\boldsymbol{\theta}$ need to have initial values from where the optimization procedure is starting. Proper initialization of the layer weights and biases is crucial for the BP algorithm to converge to a reasonable minimum, or even to converge at all.

There are some characteristics that a proper weight initialization needs to fulfill. First, the weights of the same layer need to be different, as otherwise, the network will produce the same output for every neuron in the layer, drastically reducing the computational power of the ANNs. With the same weights, the neuron outputs would always remain to be the same, as every neuron would receive the same gradient information. Hence, a proper initialization method needs to ensure that the weights of the same layer are different, i.e., by incorporating randomness. Second, the weights should be scaled such that the signal of the activation function has strong gradient information to ensure fast convergence. Moreover, also after the multiplication of weights of subsequent layers, the layer outputs should be in a regime, where the gradient information is reasonable.

A well-established weight initialization strategy is to draw values from either a uniform or normal distribution around zero. This strategy was not only used by many successful papers in the pre-CNN era but also in the CNN-era [78]. Here for example, the weights of every layer are initialized with values drawn from a normal distribution with zero mean and standard deviation of 0.001. The network biases are initialized with zeroes, except for some layers that are initialized with ones to increase convergence speed. However, a weight

initialization where the weights are drawn from the same uniform distribution for every layer is problematic and can reduce convergence speed for deep networks [128]. A possible reason for this is the vanishing gradient problem in BP [56], which states that the error signal (see Eq. (2.24)) is strong in the deeper layers closer to the network output where the loss is calculated, but almost vanishes for the first layers. A method to overcome this problem and increase performance for very deep networks is to apply the loss function not only in the last layer but also in intermediate layers, which then provides a strong error signal in all network layers, e.g., [137]. Another method to provide a strong error signal throughout all network layers to use residual connections [52] or dense connections [58]. However, such methods only prevent the symptoms but do not the cause of improper weight initialization.

As noted by He et al. [51], improper weight initializing leads to either reducing or magnifying the magnitudes of input signals exponentially throughout the network, which induces either vanishing gradients, or network divergence, respectively. Unfortunately, when initializing the network weights of each layer with always the same uniform or normal distribution, most of the time the input signals are either reduced or magnified exponentially. To overcome this, Glorot and Bengio [45] proposed to initialize each network layer with values from different distributions. They propose to draw each weight of a layer from a uniform distribution having a range that is dependent on the number of input/output connections for the neurons of a layer. Although their method provides improved performance, it is based on the assumption that the layer activations are linear, which is not the case for typical activation functions. To overcome this issue, [51] proposed to initialize the weights of the layers from normal distributions, where the standard deviation is based on the number of input and/or output connections having a different scaling factor as compared to Glorot and Bengio [45]. They propose to sample weights $w_{ji}^{(l)}$ of layer l as follows:

$$w_{ji}^{(l)} \sim \mathcal{N}(0, {}^{2}/_{\#\text{in}^{(l)}}), \tag{2.25}$$

where the normal distribution $\mathcal{N}$ has a mean $\mu = 0$ and a variance $\sigma^2 = {}^{2}/_{\#\text{in}^{(l)}}$ with $\#\text{in}^{(l)}$ being the number of input connections of a neuron of layer (l). The authors of [51] proved that their initialization method is sufficient to prevent reducing or magnifying the magnitudes of input signals exponentially throughout the network.

When using ReLU and related activation functions, initializing the network weights with Eq. (2.25) typically provides fast convergence and achieves good performance.

2.3.4 Normalization

For achieving optimal performance of neural networks, the input features should be normalized to have a mean of zero and a unit variance [81, 157]. This so-called *whitening* has shown to improve convergence speed for the training of neural networks, or even to make training possible in the first place. Although the property of zero mean and unit variance for features in other hidden layers is also expected to improve network convergence, this

property is difficult to maintain without additional normalization techniques, despite the improvements in weight initialization (see Section 2.3.3).

Many methods for normalizing the outputs of hidden layers have been proposed. One of the most influential normalization techniques is *batch normalization* [63]. In Batch normalization, the feature outputs of a hidden layer are normalized such that every feature has a mean of zero and a variance of one across the mini-batch. As these mini-batch statistics can not be calculated during inference with a mini-batch size of one, a running mean and variance of the mini-batch statistics are calculated during training, which are then used for normalizing the layer output during inference. Batch normalization not only reduces the number of needed training iterations for the network to converge but also works as a regularization to reduce overfitting. However, it has the major disadvantage that during training the mini-batch statistics may be noisy for small mini-batch sizes. This is especially problematic in the medical imaging domain, where CNNs have to be trained with small mini-batch sizes due to high-resolution volumetric data requiring lots of memory.

There exist multiple other normalization techniques that have advantages, but also disadvantages as compared to batch normalization. *Layer normalization* [5] and *instance normalization* [146] also work on the feature outputs of hidden layers. In contrast to batch normalization, they do not normalize across the mini-batches, but other feature dimensions. *Weight normalization* [120] normalizes not the features, but the weights of the hidden layers. *Self normalizing networks* [73] go in a different direction. By carefully choosing the activation function (the *scaled exponential linear unit* – SELU) and the weight initialization technique, self normalizing networks are able to retain the optimal feature statistics throughout the whole network and the whole training procedure.

Unfortunately, there does not exist *the* normalization technique that works best for every task. Thus, one has to decide which normalization technique to use or whether to use normalization at all for each task and network architecture independently.

2.4 Convolutional Neural Networks

A major drawback of standard LNNs is their *dense* connections between layers, also known as *fully connected* layers. Every neuron in such layers uses every neuron of the previous layer as its input. As there is a learned weight for every connection, *deep* or *wide* networks have many parameters and are prone to overfitting. CNNs drastically reduce the number of network parameters by replacing most *dense* layers with *convolution* and *pooling* layers [80]. This allows training networks with lots of hidden layers leading to high computational complexity while keeping the number of trainable parameters reasonable.

Despite the advantages of CNNs as compared to standard LNNs, the requirement of large training datasets, as well as high computational demands of training such networks, hindered their early breakthrough at the end of the 1990s [80]. CNNs had only achieved their breakthrough as late as 2012 [18, 78]. The rise of faster hardware and larger datasets

made the training of CNNs possible, starting the current deep learning boom.

Being inspired by the visual system [60], CNNs are especially useful for images. Similar to *simple cells* in the visual system, the convolution layer fires at structures with a specific orientation, regardless of the position on the image. Stacking convolution and pooling layers allows the modeling of complex interactions, which is similar to the mechanism of *complex cells* in the visual system. Hence, such deep CNNs have extraordinary image feature extraction capabilities, which are superior to other methods' capabilities relying on handcrafted features, e.g., RFs.

In the following subsections, we will give an overview of CNNs, their properties, building blocks, and novel terminology.

2.4.1 Convolution Layers

In spatially structured data, e.g., images, convolution layers leverage three ideas to improve machine learning systems, i.e., *spares interactions*, *parameter sharing*, and *translation equivariance* [47]. The discrete convolution operation of $\text{conv}(\mathbf{x}; I, K) : \mathbb{R}^d \to \mathbb{R}$ of a d-dimensional image I and kernel K is defined as:

$$\text{conv}(\mathbf{x}; I, K) = (I * K)(\mathbf{x}) = \sum_{\mathbf{y} \in \Omega_K} I(\mathbf{x} - \mathbf{y}) K(\mathbf{y}), \tag{2.26}$$

where $\mathbf{x}$ and $\mathbf{y}$ define discrete positions on a regular grid, e.g., pixels coordinates for images, and Ω_K defines the support of kernel K, i.e., $\Omega_K = \{\mathbf{x} \in K \,|\, K(\mathbf{x}) \neq 0\}$. An example of a kernel of size 3×3 being convolved over an image of size 4×4 is shown in Fig. 2.10. Note that in order to maintain the image size after convolution, in this example the pixels on the border are padded with zeroes. More information on padding is given in Section 2.4.1.2.

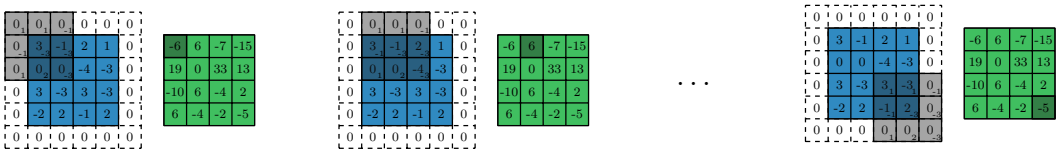


Figure 2.10: Example of calculating output values of a discrete convolution for a 2-dimensional image. The blue square is the input image; the green image is the output image. The values of the pixels are written inside each pixel. The gray square in the input image shows the convolution kernel, while its values are shown in the bottom right corner of each pixel. The gray square in the output image shows the current output pixel for the current position. The 0 pixels on the border indicate zero padding, which is needed to keep the size of the output the same as the input. Images are generated and adapted from [30].

In the convolution operation, the channel or *feature* dimension of the image I serves as an additional spatial dimension, e.g., a two-dimensional multi-channel input image turns into a three-dimensional volume. The support of the convolution kernel K is typically small for the width and height dimension of the input image, while the kernel covers the whole feature dimension. Furthermore, a convolution layer typically does not only consist of one but multiple different kernels, where each one forms one entry of the feature dimension of the output image. A visualization of this is given in Fig. 2.11.

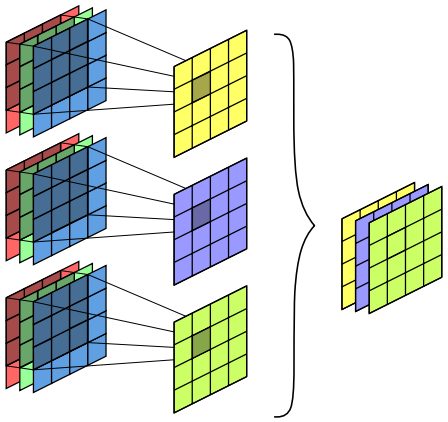


Figure 2.11: Visualization of a convolution layer with multiple input and output channels. The red, green, and blue images visualize the input *channels* or *features* for the convolution layer. The yellow, violet, and lime images visualize the outputs for the three kernels of this layer, which are combined to the output of this layer. The three different kernels cover the whole feature dimension, but not the whole width and height of the input images.

2.4.1.1 Properties of Convolution Layers

Differently to dense layers, neurons in convolution layers are not connected to all neurons of the previous layer, but only to the neurons that are within the support Ω_K of the kernel K, i.e., a subset of neurons that is spatially close to the neuron's position. While this *sparse interaction* between subsequent layers drastically reduces the number of network parameters, the layer's *receptive field* is restricted by the kernel size, e.g., in the case of images, a neuron does not see all pixels of the previous layer anymore. As in consecutive convolution layers, the support of the kernels is summed up, deep CNNs still obtain large receptive fields. Therefore, deep CNNs obtain a large receptive field from consecutive convolution layers as the effective support of the kernels is summed up.

In addition to the *sparse interaction*, the weights on the connections are shared among all neurons in the same layer, which reduces the network parameters as compared to dense layers even more. This particular form of *parameter sharing* of the kernel K of the convolution operation allows interpreting the output of a convolution layer as a local image feature, e.g., an edge or blob. Deep CNNs are able to model more global image features by consecutive convolution layers.

The image features represented in a convolution layer are independent of the position on the image, i.e., they are *translation equivariant*. This means that when the input is translated by a certain number of pixels, the output of the convolution layer is translated in the same way. Furthermore, the number of parameters of a convolution layer only depends on the size of the kernel, hence, it is independent of the input image size.

2.4.1.2 Terminology

The convolution layer introduces new terminology to neural networks, which we describe in the following.

In dense layers, the *number of features* denotes the number of neurons. In convolution layers, however, the *number of features* denotes the size of the channel dimension. For example, for a two-dimensional image of size 128×128 and 32 channels, the number of features is 32, although the number of neurons is $128 \times 128 \times 32 = 524,288$.

When describing the *kernel size*, the size of the kernel's channel dimension is usually not mentioned, as it is implicitly defined by the number of channels of the layer's input. For example for a convolution layer that has an image with 64 channels as input, 16 channels as output, and a kernel size of 3×3, the input images will be convolved with 16 different kernels, each having a size of $3 \times 3 \times 64$. This convolution layer will have in total $3 \times 3 \times 64 \times 16 = 9,216$ weights. Note that the number of weights in independent from the width and height of the image.

When convolving an image with a discrete kernel of a specific size as in Eq. (2.26), the output of pixels on the border is not defined, as the kernel would cover pixels outsize the input region. There are multiple techniques for handling the pixels on the border. The first technique is to convolve the kernel only over the defined input region. Although this would cover all existing pixel values, it would also reduce the image size, e.g., for input size 128×128 with a kernel size 7×7 the output size would be 122×122. The reduced image size is problematic for image-to-image networks, e.g., used for segmentation, as the pixels on the image border would not be labeled. A technique that prevents reducing the image size is *padding*. Using the same example as before, the input of size 128×128 would be padded to have a size of 134×134 before convolving it with a 7×7 kernel, such that the output would still be 128×128. A comparison of a convolution layer with and without padding is visualized in Fig. 2.12. There exist multiple padding techniques, e.g., *reflecting* the image, or *duplicating* the last and first row or column on the border, while the most dominantly used padding is *zero* padding, which sets the padded values to zeroes.

Another term that is important for convolution layers is the *stride*. The stride defines the number of pixels the kernel is translated between on the input image neighboring pixels of the output image. Hence, the stride can be used to reduce the image size by an integer factor. As an example Fig. 2.13 shows a convolution of size 2×2 with a stride 2×2 that halves the input width and height.

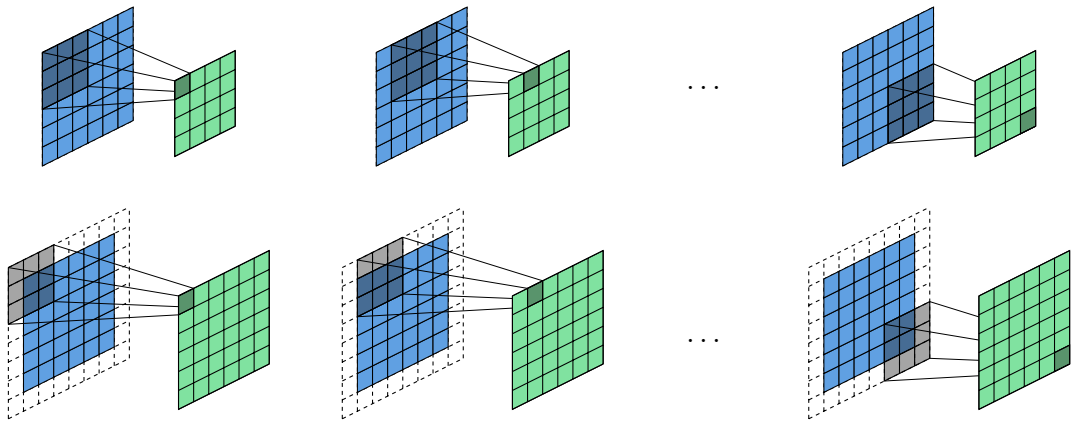


Figure 2.12: Visualization of a convolution operation without padding and with padding. The top row shows a convolution without padding, where the size of the output image is reduced; the bottom row shows a convolution with padding, where the size of the output image stays the same.

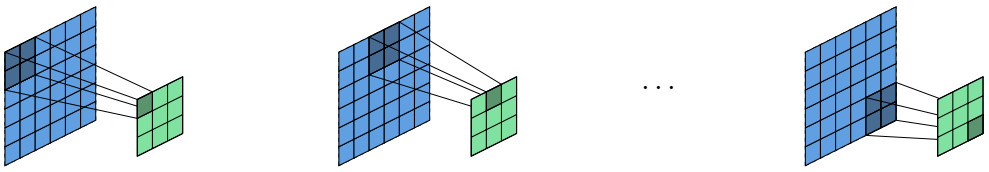


Figure 2.13: Visualization of a convolution operation with a stride of two. As not only the kernel size but also the stride is 2×2, the image size is halved after applying the convolution operation.

2.4.2 Pooling Layers

Pooling layers have the same spatial properties of convolution layers, but in contrast of convolving the input image with learned weights of kernel K as in Eq. (2.26), the pooling operation $\text{pool}(\mathbf{x}; I, K)$ that maps $\text{pool} : \mathbb{R}^d \to \mathbb{R}$ applies a function f to the pixels that are within the support of kernel K, i.e.,

$$\text{pool}(\mathbf{x}; I, K) = f(\{I(\mathbf{x} - \mathbf{y}) \,|\, \mathbf{y} \in \Omega_K\}), \tag{2.27}$$

The most commonly used functions are *average* and *maximum* [170], which set each output pixel to the average or maximum of the input pixels inside the kernel (see Fig. 2.14). These functions are parameter-free, hence, they do not increase the number of network parameters.

In contrast to convolution layers, where a kernel K covers the whole input channel dimension, pooling layers are applied to each input channel separately and independently. For example, a pooling layer with kernel size 2×2 and stride 2×2 with an input layer of size 64×64 with 16 features will create an output image of size 32×32 with also 16 features. Also note that while in convolution layers the channel size of K is typically not

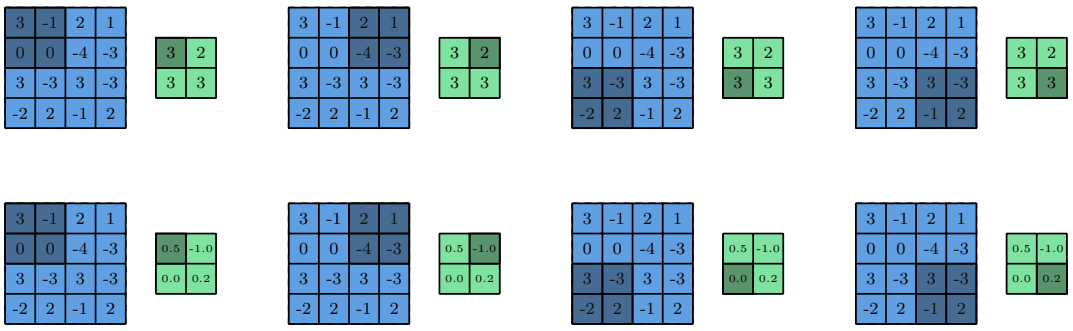


Figure 2.14: Computing the output values of a discrete pooling operation. The top row shows a *max* pooling, whereas the bottom row shows *average* pooling. The pooling layers have a kernel of size 2×2, while the stride is also set to 2×2 to reduce the image size.

mentioned but implicitly defined by the number of input channels, in pooling operations the channel size of K is not mentioned as it is one.

Pooling layers are mostly used to reduce the image size within the network. Thus, they are typically used with a stride larger than one. When using *max* pooling and setting the stride equal to the pooling size, the layer introduces slight translation, rotation, and scale invariances. As it does not matter for the *output* of the pooling operation at which position of the kernel the *input* has the maximum value, slight translations, rotations and scalings produce the same result.

2.4.3 Fully Connected Layers

In the context of CNNs, a *fully connected* or *dense* layer is the same as the basic layer of an ANN, i.e., every input neuron is connected to every other output neuron. Such layers are used in classification and regression networks to transform the intermediate output images of convolution or pooling layers to the final network output. Hence, such layers model functions that transform image-based features into high-level concepts like class probabilities and regression values.

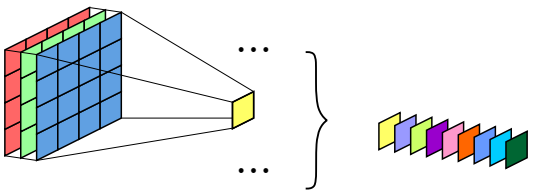


Figure 2.15: Visualization of a fully connected layer. A single output neuron (e.g., the yellow neuron) of a fully connected layer is connected to all input neurons, thus, introducing lots of weights and network parameters. Multiple output neurons form the output of the fully connected layer.

While these layers are necessary for certain tasks, they impose major drawbacks to CNNs. First, in contrast to convolution layers, where the number of parameters is dependent on the kernel size, the number of parameters of fully connected layers is dependent on the number of input neurons. Another implication of that is that CNNs with fully connected layers are not independent of the size of the input images anymore, thus, can only be applied to images of the same size. Second, such layers often account for the majority of network parameters, which could easily lead to overfitting [128]. A lot of research focused on reducing drawbacks of fully connected layers, e.g., by replacing them with other layers like global average pooling [84], by putting more computational capacity into the convolution layers of the network [52], or by regularizing the network parameters to prevent overfitting (see Section 2.5).

2.4.4 Transposed Convolution Layers

The *transposed convolution* layer is also called *backward convolution*, *fractionally strided convolution*, or *deconvolution* [88, 117, 126]. The name *transposed convolution* originates from the implementation. When implementing convolutions with matrix operations, the internal matrix that is used in the calculations of the forward and backward passes is transposed as compared to the normal convolution layer. Hence, the layer can easily be implemented by exchanging the forward and backward passes of a standard convolution layer. The *transposed convolution* layer is able to model the same functions as the normal convolution layer [47]. However, it has different properties when being applied with stride or padding (see Fig. 2.16). As the layer can be seen as a convolution in the *backward* direction, e.g., a stride of 2 does not *halve*, but it *doubles* the image size. Thus, also the name *fractionally strided* convolution, because a transposed convolution layer with stride 2 can be interpreted as a convolution layer with a stride of $1/2$.

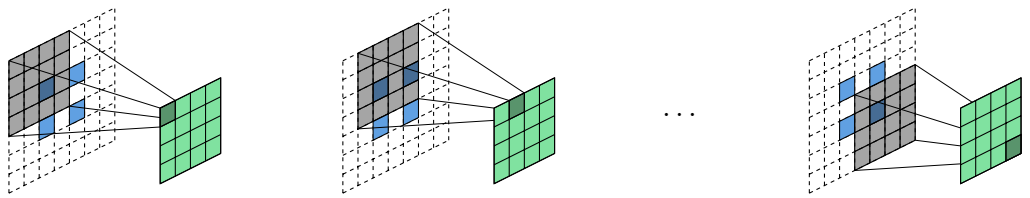


Figure 2.16: Computing the outputs of a transposed convolution layer. The shown transposed convolution has a kernel size of 4×4, padding of 4×4, and a stride of 2×2. This example shows that transposed convolution operations may be used to increase the image size and learn upsampling functions.

Due to the possibility of increasing the image size, the *transposed convolution* layer is mainly used for upsampling intermediate images in fully convolutional networks [88, 117, 126]. In such applications, the layer learns an upsampling function. However, when not used carefully with wrong hyperparameters, the layer causes image artifacts [102]. In our

networks, we observed that using a fixed upsampling function, e.g., nearest neighbor or linear upsampling works better than learning the upsampling function with a *transposed convolution* layer. Moreover, when using fixed upsampling layers that do not incorporate learned parameters, the number of network parameters is smaller as compared to using *transposed convolution* layers.

2.4.5 Other Convolution Layers

There exist more modifications to the convolution layer, which change how the neurons and weights are connected between layers. For example, dilated convolution [13, 166] make the convolution kernel sparse, thus, they increase the layer's receptive field while keeping the kernel parameters low. Another example is grouped convolutions [78, 159], which change how the convolution kernel connects the feature channels among layers. This can reduce the number of network parameters while only marginally reducing the computational capacity of the network. As in our networks, we do not use such convolution layers, we do not discuss them in more detail, but refer to overviews as, e.g., in [47].

2.5 Network Regularization

Determining an appropriate capacity of a machine learning model is essential for high performance (see Section 2.1.2). This is especially difficult for ANNs due to their flexible structure leading to countless possible architectures as well as possible training strategies, resulting in numerous hyperparameters. Furthermore, due to their high number of parameters, ANNs are prone to overfitting.

To evaluate which hyperparameters result in the best performance, the validation error needs to be tracked and observed during training. For ANNs it is especially critical that they are not optimized too excessively to the training data, as otherwise, they are likely to start overfitting. Hence, it is also important to appropriately set the network solver's hyperparameters, e.g., mini-batch size, learning rate, and the number of training iterations (see Section 2.3.1).

So far, we only discussed that we can change the model complexity in terms of network architecture and the learning algorithm's hyperparameters to reduce the generalization error. However, we can also give a learning algorithm preference for *simple* hypotheses, i.e., solutions. A complex hypothesis will be chosen only if it fits the training data significantly better. Any modification to a learning algorithm that is intended to reduce its generalization error but not its training error is called *regularization*.

2.5.1 Weight Regularization

A basic regularization technique that is frequently used in various machine learning methods is *weight regularization*. As large weights in machine learning models often indicate overfitting, the idea of weight regularization is to prefer models with small weights. This

is in line with Occam's razor, which states that when two hypotheses (machine learning models in our case) provide the same solution, prefer the *simpler* one (i.e., the one with less or smaller weights).

For ANNs, weight regularization adds a term penalizing large network weights $\mathbf{w}$ to the loss function L. Here, $\mathbf{w}$ is a vector consisting of all weights of all layers of the network. When adding the squared magnitude of $\mathbf{w}$, i.e., the squared L_2 norm, of the weights multiplied with a factor λ to L, the total loss is defined as,

$$L = L + \lambda \|\mathbf{w}\|_2^2. \tag{2.28}$$

This L_2 weight regularization is also called *weight decay*. Due to the squared magnitude, large weights are penalized much and small weights are preferred.

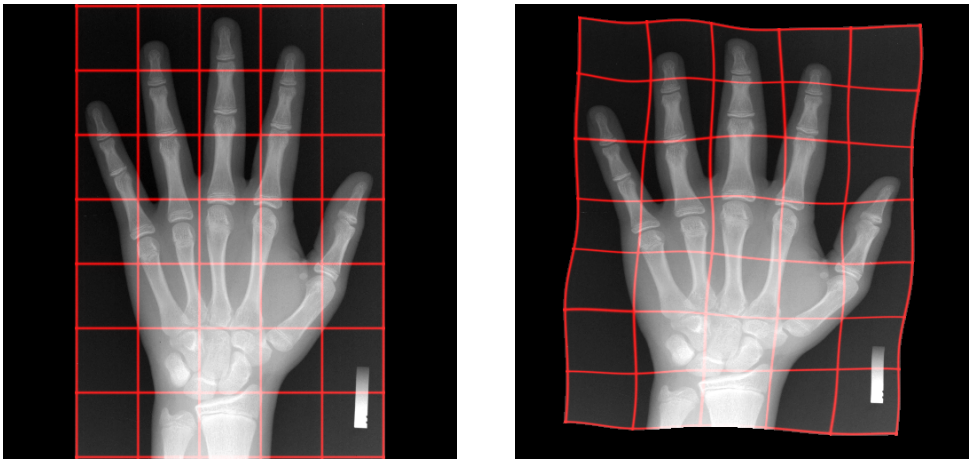
In deep CNNs, weight regularization is frequently used due to its potential for reducing overfitting. However, the factor λ has to be chosen carefully. If λ is too large, the optimizer focuses too much on minimizing the norm of the weights (**??** and Eq. (2.28)), and not on minimizing the actual loss of the task. Nevertheless, a reasonably small weight regularization factor λ does not hinder the optimizer from minimizing the loss of the task, while it reduces out-of-scale network weights and overfitting.

2.5.2 Dropout

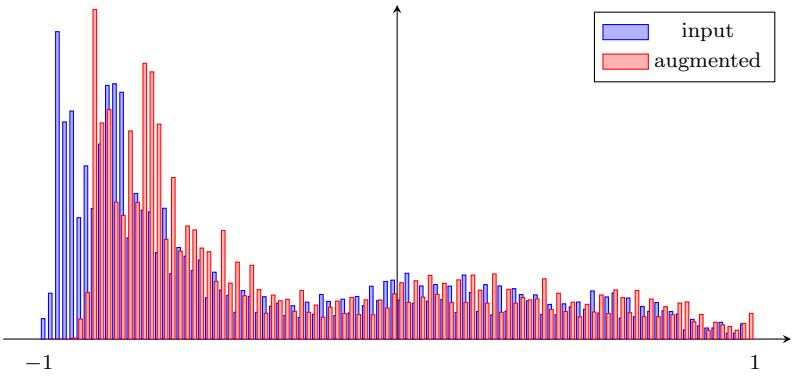
Dropout is a recent and highly effective regularization technique for ANNs [129]. The principle of dropout is to randomly deactivate neurons during training, by setting the neurons' activation value to zero with a certain probability p. Therefore, dropout may be interpreted as a way of regularizing a neural network by adding noise to its hidden units. During inference, the features are not randomly deactivated any more, but the activation values are multiplied with the factor p to bring them into scale.

As for every training iteration, different neurons are randomly deactivated, dropout has some beneficial effects that reduce overfitting of ANNs to the training data. First, due to these unreliable feature activations, dropout enforces the layer to not only rely on a specific feature but to combine information from multiple features. The layer needs to incorporate multiple sources of information to activate a feature, which reduces overfitting and leads to more diverse features. Second, the random feature deactivation can be considered as sampling multiple networks with a similar structure. For every training iteration, a different network that shares weights with other similar networks is being optimized. During inference, all sampled networks are combined into a single neural network, which represents an ensemble of all trained networks.

Interestingly, networks that use dropout extensively often have a training loss that is larger than the validation loss. The reason for that is that the individually sampled training networks have a smaller capacity than the ensemble that is used during inference. Hence, when using dropout, the number of intermediate features might need to be increased for the network to have the same modeling capacity as the network without dropout.



(a) Input image. **(b)** Augmented image.



(c) Intensity histograms.

Figure 2.17: Example of the employed data augmentation techniques. The images on the top show an example input and its augmented version, where random translation, scaling, rotation, and elastic deformations are applied. Furthermore, the intensity values were scaled and shifted, as seen in the intensity histograms at the bottom

2.5.3 Data Augmentation

Another regularization technique for deep neural networks of utmost importance in the medical imaging domain is data augmentation [17, 117]. With data augmentation, the training examples are modified by random transformations to better cover the theoretical input data distribution of the machine learning task. Although each synthetic training example correlates with its original unmodified training example, data augmentation still drastically increases the number of different examples that the machine learning program observes during training.

Different from the computer vision domain, in the medical imaging domain, many ways of augmenting data can be used to modify the training examples, such that the augmented training examples still look realistic. An example of an augmented hand radiograph is

given in Fig. 2.17. We mainly use two input data augmentation techniques in this thesis: intensity transformations and spatial transformations.

Intensity transformations directly modify the pixel/voxel values of the image/volume. Transformations include shifting (i.e., addition) and scaling (i.e., multiplication) with random values from a certain range, as well as adding pixel-wise noise. Such intensity augmentations are especially useful when dealing with imaging modalities, where the intensity value range is not fixed either due to fluctuating illumination (e.g., in microscopy) or unknown scale (e.g., in Magnetic Resonance Imaging (MRI)).

Spatial transformations modify the orientation and shape of the object that is visible in the image/volume. Transformations include translation, scaling, rotation, as well as elastic deformations. Although CNNs have properties to be invariant to translations, as well as slight scalings and rotations, we found spatial transformations crucial for preventing overfitting. This is due to the fact that random spatial transformations combined with resampling may modify the images sub-pixel-wise. When translating an image by an integer value and using it as an input for a convolution layer, the output of the convolution layer will be exactly the same for both images but translated by the same integer value. As this also leads to the same gradient for the convolution layer (if we are neglecting the pixels on the border), this is not useful for augmenting the data to prevent overfitting. When translating an image by a float value (i.e., in sub-pixel space) however, the output of a convolution layer will be slightly different for both images. This leads to different gradients, which is beneficial for preventing overfitting. We also found elastic deformations to be very beneficial. While they are not used in the computer vision domain, as they would produce images that are unrealistic, in the medical imaging domain elastic deformations produce images that still look realistic, especially for volumetric images showing regions of soft tissue. We implement elastic deformations by randomly translating points on a regular grid on the input image and interpolating the image intensity values with third-order B-splines. These elastic deformations transform the image inhomogeneously, i.e., different regions in an image are transformed differently.

In practice, when working with a new dataset, we initially examine the training data and perform experiments to estimate the data augmentation hyperparameters. We found that slightly exaggerated data augmentation hyperparameters that generate *slightly* unrealistic images are beneficial for training CNNs. With such images, the networks learn to solve a more difficult task during training, while the validation or testing data are more likely to be within the data distribution that has been observed during training. Due to the typically large number of data augmentation hyperparameters, when we found good hyperparameters that cover well the training data distribution, we fix them for this dataset and do not modify them for subsequent experiments anymore.

Although data augmentation may also be used during inference by merging differently augmented predictions, in our experiments, we only apply data augmentation during training. For this, we implemented a CNNs training framework that performs on-the-fly training data augmentation with hyperparameters that are randomly sampled from specified

ranges (see Chapter 6). Due to this on-the-fly generation of training images, the network practically never observes exactly the same training examples, which hinders the network from learning the training examples by heart and drastically reduces overfitting.

2.6 Summary

In this chapter, we have given an introduction to machine learning in general. As our proposed methods are based on ANNs, we have explained the basic framework for training ANNs in more detail. We have introduced terminology that we will use throughout the thesis, which is required for defining neural networks, as well as for optimizing them for solving various tasks. Moreover, we have introduced CNNs and their building blocks extending basic ANNs. Finally, we have shown regularization techniques utilized within this thesis, which improve the performance of CNNs to generalize better to unseen data.

3

Related Work on Landmark Localization

Contents

3.1 Landmark Localization 39
3.2 Graphical Models for Landmark Localization 40
3.3 Image Feature Responses for Landmark Localization 46
3.4 CNNs in Anatomical Landmark Localization 53
3.5 Our Contributions to Landmark Localization 55

> Nobody expects the Spanish Inquisition! Our chief weapon is surprise...surprise and fear...fear and surprise.... Our *two* weapons are fear and surprise...and ruthless efficiency.... Our *three* weapons are fear, surprise, and ruthless efficiency...and an almost fanatical devotion to the Pope.... Our *four*...no... *Amongst* our weapons.... Amongst our weaponry...are such elements as fear, surprise.... I'll come in again.
>
> *Cardinal Ximénez*

In this chapter, we will give an overview of the task of landmark localization in general. In Section 3.1, several related tasks and common terminology are defined. In Section 3.2, we show how to incorporate graphical models into landmark localization, while in Section 3.3 we list some methods of how to extract image features. Finally, in Section 3.4, we give a short overview of how Convolutional Neural Networks (CNNs) are being used specifically for the task of anatomical landmark localization, while we present our specific contributions to the research field in Section 3.5.

3.1 Landmark Localization

Landmark localization in general is the task of both identifying and determining the coordinates of (multiple) specific landmarks in images. Many different tasks in the computer

vision as well as the medical imaging domain deal with landmark localization. In computer vision, the tasks range from hand pose estimation [33], over face alignment [85], to human pose estimation [145]. In medical imaging, the tasks range from localizing landmarks inside organs like heart [104] or brain [169] for subsequent registration, over localizing landmarks on the border of organs like the lungs [61] for segmentation, to localizing landmarks on lateral cephalograms [152] or spine Computed Tomography (CT) volumes [44] for diagnostic purposes. All the aforementioned tasks in both computer vision and medical imaging have in common that they use a d-dimensional image $I : \Omega_I \subset \mathbb{R}^d \rightarrow \mathbb{R}$ to determine the coordinates $\mathbf{x}_i \in \Omega_I$ of N specific landmarks $\mathcal{L}_i$ with $i = 1 \dots N$. Note that some previous works use the term landmark localization for the task of localizing landmarks without their *identification* or *naming*, e.g., localizing all centroids of vertebrae without determining their type or label. Throughout the thesis, when using the term landmark localization, we always refer to the task of determining both the coordinate and the label of specific landmarks.

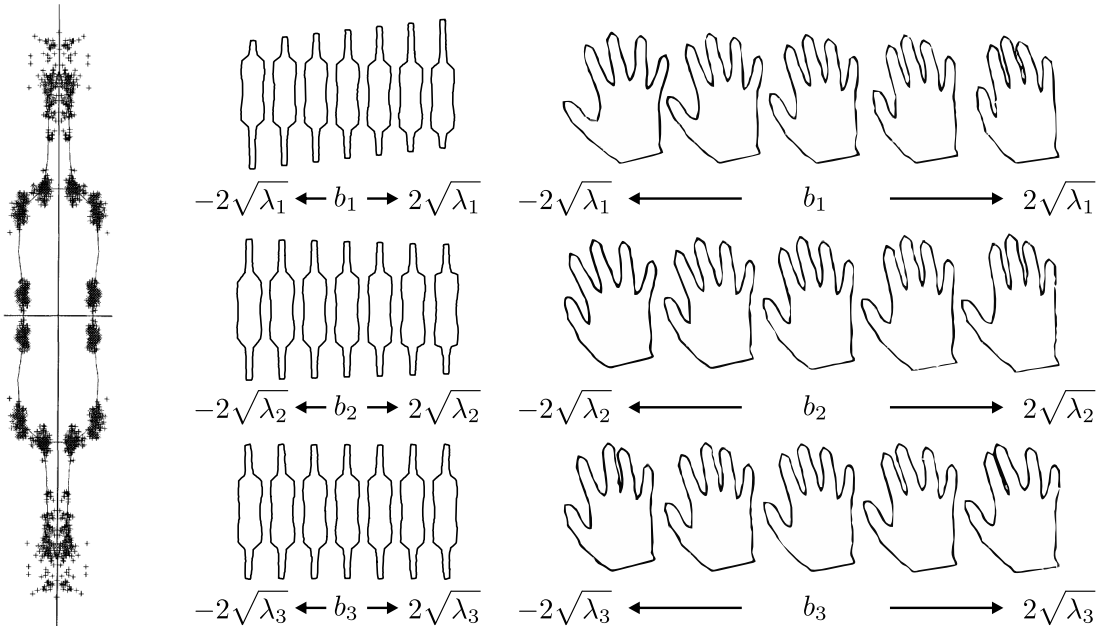
Due to the variable visual appearance and shape of landmarks on images, modern methods for landmark localization are based on machine learning, often combining local feature responses with handcrafted graphical models encoding the global spatial configuration of landmarks. In the following, we will describe various graphical models and feature extraction methods for landmark localization.

3.2 Graphical Models for Landmark Localization

Since in earlier methods the image feature extraction techniques were not as advanced as nowadays with Random Forests (RFs) or CNNs, strong models of the spatial configuration of landmarks were even more important back then. Especially in the medical imaging domain, approaches incorporating models of the spatial configuration of landmarks are extensively used, since they efficiently capture the anatomical variation, which can already be observed on a small number of training images. There are several different techniques of how to use graphical models of the spatial configuration of landmarks. In the following, we will give an overview of the most important ones.

3.2.1 Active Shape Models

Frequently used in tasks of the medical imaging community, the seminal work of Cootes et al. [22] introduced Active Shape Models (ASMs) that model points on borders to segment structures like hands or the heart ventricle. These active shape models represent the mean shape as well as the shape variation learned from a set of training shapes. See Fig. 3.1 for an example for the shapes of a resistor, as well as shape variations of the resistor model and a hand model. After aligning all M training shapes with the generalized Procrustes algorithm, with $\mathbf{x}_i = (x_1, y_1, \dots x_N, y_N)^T$ being a vector of all x and y coordinates of the



(a) Resistor. **(b)** Resistor variation. **(c)** Hand variation.

Figure 3.1: Example of *active shape models*. The left image shows the aligned points of various shapes of a resistor with the mean shape as an overlay. The image in the center shows how the three most important parameters with the largest eigenvalues change the shape of the resistor. The image on the right shows how the parameters change the shape of a hand model. The figures are taken from [22].

N landmarks, the mean shape is defined as

$$\bar{\mathbf{x}} = \frac{1}{M}\sum_{i=1}^{M}\mathbf{x}_i. \tag{3.1}$$

With this, the covariance is calculated as

$$S = \frac{1}{M}\sum_{i=1}^{M}(\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^T. \tag{3.2}$$

With Principal Component Analysis (PCA), S is decomposed into M eigenvalues λ_i and eigenvectors p_i, while only the t eigenvectors with the largest eigenvalues are kept. These t first eigenvectors then form the matrix $\mathbf{P}$, which is used to define the allowable shape domain as

$$\mathbf{x} = \bar{\mathbf{x}} + \mathbf{P}\mathbf{b}, \tag{3.3}$$

where $\mathbf{b}$ is a vector of weights. Given a new input image, the mean shape is put to an initial position and is iteratively refined by moving each landmark to the closest edge of the image, while restricting the shape to the allowable shape domain.

A drawback in the originally proposed active shape models is that they only take edge-based features of the local surroundings of a landmark into account. Various works improved on that drawback by also taking other appearance features into account, while still using the *active shape model* as the landmark model. Cootes et al. also introduced *active appearance models* [20] that deal with this drawback by not only modeling the average shape of the target structure but also its average appearance. However, a drawback of these *active appearance models* is the pixel-wise matching of the appearance information of the whole model, which may be too restrictive. Cristinacce and Cootes [25] proposed to use *constrained local models* that learn the appearance only locally around the landmarks of the shape model. This approach showed to be more robust than *active appearance models*, while also allowing more descriptive local appearance as compared to ASMs. In [21] and [85], Cootes et al. further improved upon the *constrained local models* by using Random Regression Forests (RRFs) as the local appearance model around each landmark. Here, an RRF takes a local patch around the landmark location as input and votes for the most probable location of the landmark.

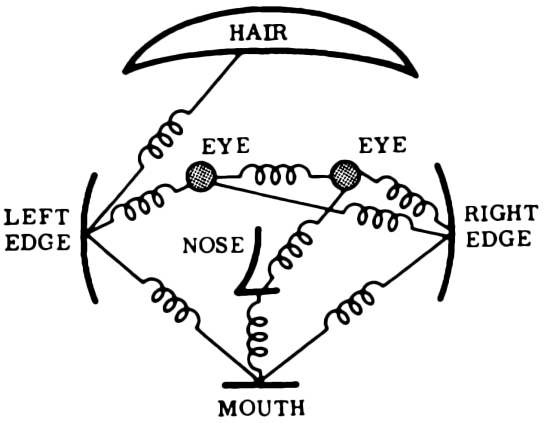
3.2.2 Pictorial Structures

Other shape-based models that have been used frequently in the computer vision community for human pose estimation and face point localization are Pictorial Structures (PS) (see Fig. 3.2). The theory on the PS dates back to the 1970s when it was proposed by Fischler and Elschlager [35]. The individual parts of the deformable PS model, e.g., eyes, mouth, and nose, are connected via spring-like connections, which allows modeling a variety of different spatial configurations while dismissing infeasible ones (see Fig. 3.2a).

In general, matching a pictorial structure to an image I is performed with energy minimization, while the energy depends on how well the individual part locations match the content of the image, as well as how well the individual parts agree with the deformable model. The deformable model is described as a graph $G = (V, E)$, where the vertices $V = v_1, \ldots, v_N$ are the N landmarks or *parts*, and if part v_i is connected with part v_j, there exists an edge with $(v_i, v_j) \in E$. An instance of a model configuration is defined as $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$, where $\mathbf{x}_i$ specifies the location of part v_i. With $m_i(\mathbf{x}_i)$ being a function measuring the degree of mismatch for part v_i with the image, and $d_{ij}(\mathbf{x}_i, \mathbf{x}_j)$ being a function measuring the degree of deformation of the model when part v_i is located at $\mathbf{x}_i$ and v_j at $\mathbf{x}_j$, the optimal match $\overset{*}{X}$ of the model to the image is defined as

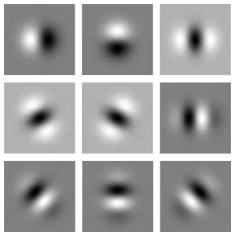
$$\overset{*}{X} = \arg\min_{X} \left(\sum_{v_i \in V} m_i(\mathbf{x}_i) + \sum_{(v_i, v_j) \in E} d_{ij}(\mathbf{x}_i, \mathbf{x}_j) \right). \tag{3.4}$$

In the eminent work of Felzenszwalb and Huttenlocher [34], the authors formulated an efficient statistical framework for both modeling and matching of PS. In this framework, the aim is to estimate the distribution $p(X \mid I, \theta)$ which defines the probability that

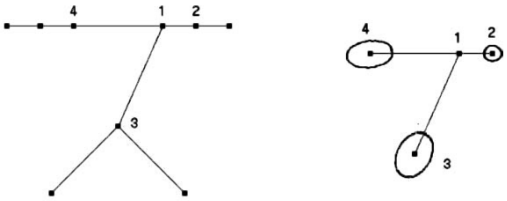


(a) Pictorial structure of a face.

(b) Face example.

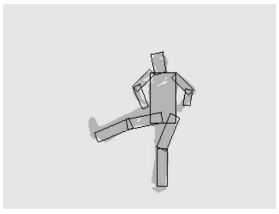


(c) Appearance.

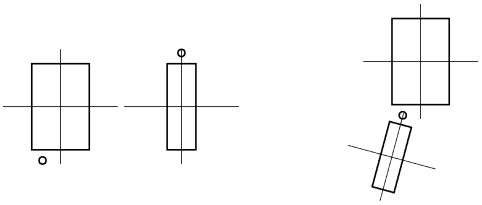


(d) Connections.

(e) Pose example.



(f) Appearance.



(g) Connections.

Figure 3.2: Example of *pictorial structures*. The second row shows an example, appearance features, and connection features for the face point model, the third row for the human pose model of [34], respectively. The image on the top is taken from [35], while the other images are taken from [34].

the configuration of the model is X, given the image I and model parameters θ. With $p(I \mid X, \theta)$ being the probability of seeing a particular image given that the object is at some location, and $p(X \mid \theta)$ being the probability that an object is at a particular location, applying Bayes' rule leads to

$$p(X \mid I, \theta) \propto p(I \mid X, \theta) p(X \mid \theta). \tag{3.5}$$

Felzenszwalb and Huttenlocher [34] parametrize their PS model with $\theta = (u, E, c)$, where $u = \{u_1, \ldots, u_N\}$ are the appearance parameters of the parts, E defines which parts are

connected, and $c = \{c_{ij} \,|\, (v_i, v_j) \in E\}$ are the connection parameters. When applying Maximum-A-Posteriori (MAP) estimation this leads to

$$p(X \,|\, I, \theta) \propto \left(\prod_{v_i \in V} p(I \,|\, \mathbf{x}_i, u_i) \prod_{(v_i, v_j) \in E} p(\mathbf{x}_i, \mathbf{x}_j, |\, c_{ij}) \right). \tag{3.6}$$

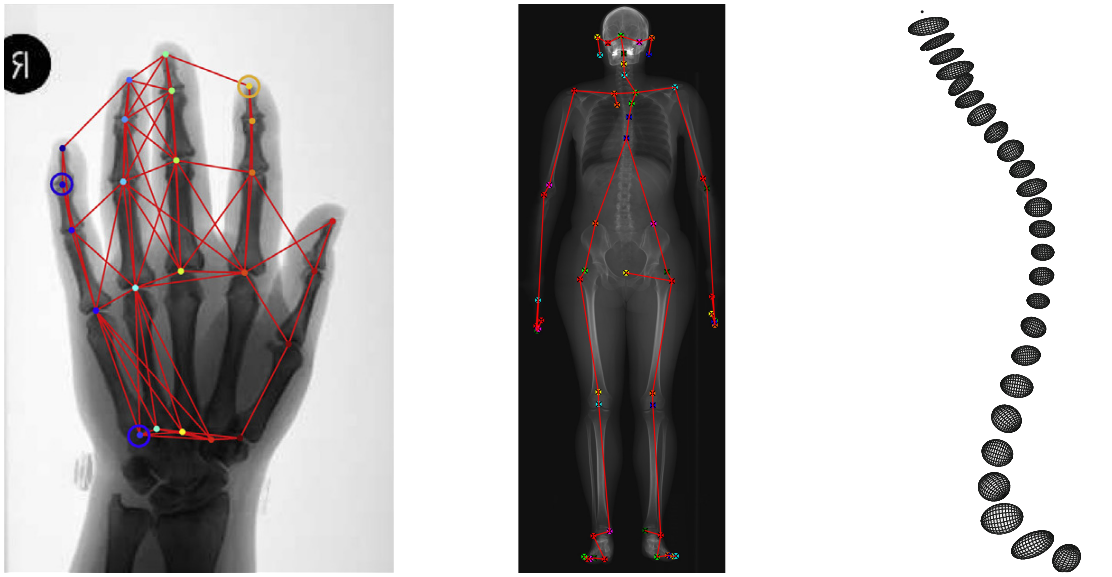
With this framework, there are multiple possibilities of representing objects with both appearance features u_i and connection features c_{ij}. Felzenszwalb and Huttenlocher [34] show two examples of matching face points and human poses. For face points, they set the appearance features to be mixtures of Gaussian derivative functions of the image, and connection features as the mean and covariance of relative positions of parts. For human poses, they set the appearance features to depend on the number foreground and background pixels within rectangles surrounding the body parts, and the connection features to force parts being connected at locations of joints.

Similar to the original formulation of ASMs, the original formulation of PS had shortcomings in the definition of the appearance and connection features. Several works improved upon them by applying better image feature extraction methods, e.g., by transferring appearance models between body parts [32] or by using discriminative features with AdaBoost [3]. Another research direction in the PS framework was to enrich the connection features by allowing higher-order features [140] or more complex joint relationships [161].

3.2.3 Markov Random Fields and Hidden Markov Models

Other frequently used graphical models in medical image analysis are based on MRFs or HMMs. Such models have been successfully used to localize landmarks in many applications, like lung segmentation [61], brain registration [141], and analysis of whole-body CT scans [116, 147], spine CT scans [42, 74], and hand radiographs [28, 130]. Similar to PS, MRFs and HMMs are typically used to model both unary and pairwise terms in a graph of landmarks. Hence, they also combine responses from image features as the unary terms with the relative positions among landmarks as pairwise terms. However, differently to PS, which are often restricted to tree-like graphs, methods using MRFs or HMMs typically allow more general graphs. Moreover, they are often applied on top of the unary local predictions of individual landmarks, while the unary predictions are independent. Usually, in a refinement step after predicting multiple possible solutions from image features as the unary terms, such models then identify the most probable solution out of the initial ones based on the pairwise terms.

While in general MRFs and HMM are able to model also higher-order terms, typically they are used with unary and pairwise terms. Same as in PS, the MRF or HMM is described as a graph $G = (V, E)$, where the vertices $V = v_1, \ldots, v_N$ are the N landmarks, and if landmark v_i is connected with part v_j, there exists an edge with $(v_i, v_j) \in E$. The



(a) Hand graph. **(b)** Body graph. **(c)** Spine positions.

Figure 3.3: Example graphs for *Markov random fields* and *hidden Markov models*. The image on the left from [28] shows a graph of landmarks of the hand that was calculated automatically and used to refine predictions with a Markov Random Field (MRF). The image in the middle from [147] shows a manually created graph for landmarks of a whole-body CT scan. The image on the right from [42] shows the variation of landmarks around individual vertebrae of the Hidden Markov Model (HMM) of CT images of the spine.

models are optimized via an energy minimization function, which is defined as

$$\overset{*}{X} = \arg\min_{X} \left(\sum_{v_i \in V} u_i(\mathbf{x}_i) + \sum_{(v_i, v_j) \in E} p_{ij}(\mathbf{x}_i, \mathbf{x}_j) \right), \tag{3.7}$$

where u_i represents the unary terms, and p_{ij} the pairwise terms of the landmarks.

While MRFs and HMMs are very powerful in terms of modeling capacity, they may be too restrictive. For example, Glocker et al. used an HMM-based refinement step in their work on vertebrae localization [42], while refraining from this refinement step in the subsequent work [44], as they observed that HMMs were too restrictive, leading to worse solutions in pathological cases. Thus, especially for cases being far away from the healthy norm, which are underrepresented in the training data, the HMMs was not able to model the prior of the landmark distribution satisfactorily from the observed data. Similarly, in works of our research group, we also restricted results with MRFs in [130], while in the follow-up work [147], we directly included a geometric landmark model into an iterative method, leading to better results.

3.2.4 Other Graphical Models

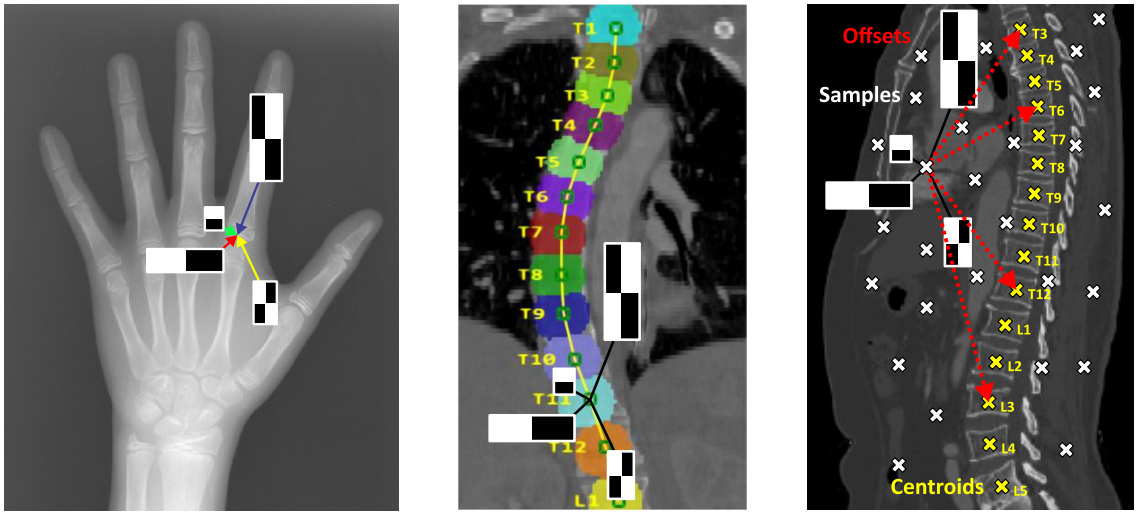
There are also other ways of integrating graphical models into landmark localization. For example, search strategies on localizing the next landmark based on previous predictions [70, 87, 111]. In [70], at first the whole spine, then cervical/thoracic/lumbar vertebrae, and finally each vertebra individual are iteratively localized within smaller and smaller search spaces. The method of [87] successively restricts the search space of a landmark based on the relative positions of predictions for other landmarks. With such search space restrictions, the runtime can be greatly reduced, while also delivering better performance due to removing false-positive predictions that are outside of the search space.

Also, simpler models, specifically designed with the target task in mind are applied in landmark localization. For example in vertebrae localization, due to the specific shape of the spinal column, models of polynomial curves may be fitted through the landmarks in the center of the vertebrae [59, 113]. Due to all vertebrae being present on the line through the spinal column, such polynomials may be used to model both the order of the vertebrae as well as the shape of the spine.

3.3 Image Feature Responses for Landmark Localization

For an accurate landmark localization, not only graphical models of the relative positions of landmarks but also image feature extraction methods capable of observing the local surroundings of landmarks are important. In earlier methods for landmark localization, due to the lack of fast and powerful image feature extraction methods, mostly handcrafted features were used. For example in the original work of ASMs [22], the landmarks were required to be on strong edges. The image feature responses, in this case, were based on image edges, i.e., large magnitudes of image gradients. Another way of using local image information is to use templates of the surroundings of landmarks. For example in [25], a template based on an *active appearance model* for every landmark is learned and used for subsequent matching. Other frequently used image features are low-level image features. In [34] for example, the image features of the pose model included the ratio of foreground to background pixels in a box around the target structure. In their face point model, they used image features based on differential Gaussian functions.

While there exist many other methods for constructing handcrafted features for landmarks, nowadays mostly machine learning based feature extraction is used. Before the deep learning boom, RFs were predominantly used for feature extraction in anatomical landmark localization. In Section 3.3.1 we will explain how RFs perform feature extraction. In Section 3.3.1 we will then show how current CNNs are used for this task.



(a) Haar-like box features. **(b)** Classification. **(c)** Regression.

Figure 3.4: Example of features used for *classification* and *regression random forests*. The image on the left shows a hand radiograph with a visualization of Haar-like features. The image in the middle shows how such features are used in RFs performing *classification* of each pixel. The image on the right shows RFs performing *regression* by estimating offsets to each landmark from the observed features at each sample. The middle and right images are adapted from [43].

3.3.1 Random Forests

RFs are a supervised machine learning technique (see Section 2.1) that may be used for both classification and regression tasks [10]. In general, the task for RFs is to obtain a posterior distribution $p(\mathbf{y} \mid \mathbf{x})$, where $\mathbf{x} \in \mathbb{R}^n$ is an observation represented as an n-dimensional input vector, and $\overset{*}{\mathbf{y}} \in \mathbb{R}^m$ is the m-dimensional target. An RF consists of multiple decision trees that are combined to generate a more robust prediction, while the predictions $p^{(t)}(\mathbf{y} \mid \mathbf{x})$ of each of the T decision trees are combined by averaging

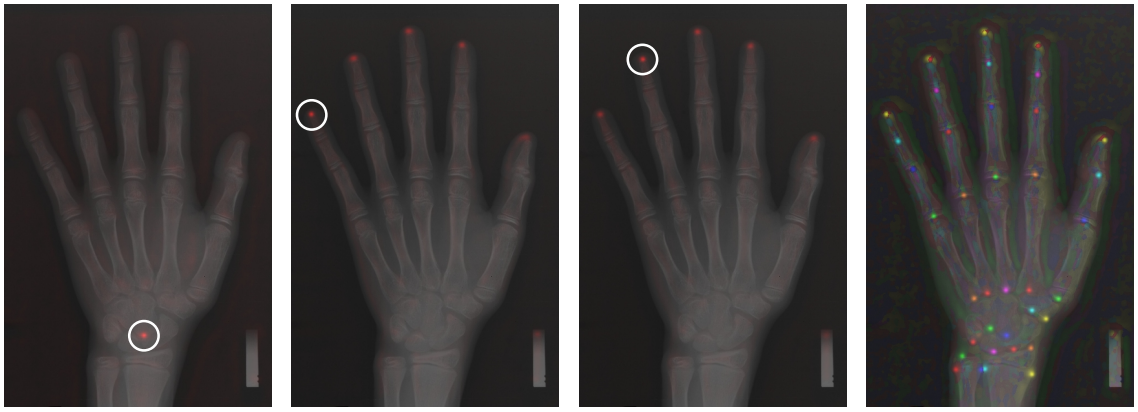
$$p(\mathbf{y} \mid \mathbf{x}) = \frac{1}{T} \sum_{i=1}^{T} p^{(t)}(\mathbf{y} \mid \mathbf{x}). \tag{3.8}$$

From a large set of possible features, the individual decision trees are trained to identify the features that maximize measures based on the *entropy* or the *information gain* with the target task in mind. When performing tasks on images, box features based on Haar-wavelets are often used in RFs (see Fig. 3.4a), since such features may be efficiently calculated [150].

Both classification and regression forests were used extensively for anatomical landmark localization [21, 24, 28, 31, 42–44, 85, 130, 147, 161].

In classification forests, the task is to assign to each pixel the label of its underlying structure, e.g., the underlying vertebrae in the task of vertebrae localization [44]. In this

example, the task is not directly a localization task, but more an (approximate) multi-label segmentation task (see Fig. 3.4b). The localization predictions are identified by taking, e.g., the center of mass of the predicted classification labels. A drawback of such classification forests is that the image region that is used for the prediction of a pixel is limited.



(a) Carpal. **(b)** Little finger. **(c)** Ring finger. **(d)** All landmarks.

Figure 3.5: Example heatmap outputs from *random regression forests*. The heatmaps are generated with the first stage of the method of Urschler et al. [147] and are shown as an overlay on top of the input image. In the first three images, the target landmarks are marked with a white circle. The last image shows the merged heatmap outputs for all landmarks.

In regression forests, the task is to estimate the position of the target landmark, based on the underlying structure of the current sample [21, 31, 42, 85, 130, 147]. Based on the features seen at the current sample, an offset to all landmarks is calculated (see Fig. 3.4c). This is different from classification forests, as in RRFs samples from potentially every position of the image may contribute to the prediction of each landmark. When performing inference of RRFs, for every pixel of the image, the location of the target landmarks can be estimated. All these predicted locations for each landmark i are accumulated and form a *heatmap* $H_i(\mathbf{x})$ with $H_i : \mathbb{R}^d \rightarrow \mathbb{R}$, which is an image representing a pseudo-probability of the landmark i being located at coordinate $\mathbf{x}$. The actual landmark coordinate may then be obtained by, e.g., taking the coordinate where the heatmap has its maximum value. Some example heatmaps of landmarks on hand radiographs obtained with the first stage of the method of Urschler et al. [147] are shown in Fig. 3.5. Here, in the first image showing the heatmap of a landmark located on a carpal bone, one can observe that the responses are concentrated near the correct position of the landmark. However, in the heatmaps of the fingertips of the little and ring finger, the responses are more ambiguous and similar for all fingertips. This shows the difficulty of standard RRFs to incorporate global context information.

Due to the high accuracy and robustness, RRFs were frequently used for landmark

localization. Several works dealt with the challenge of how to incorporate global context information into the prediction of RRFs. While many methods include global context information with graphical models, e.g., ASMs [21, 85], MRFs [28, 130], and HMMs [42], other methods aim to robustly combine both local and global features into the RRF framework. For example, [31] have adapted the seminal work of [24] on organ bounding box localization to first robustly restrict the predicted region based on global appearance features, followed by accurate localization based on local features. However, their performance is highly dependent on whether the first cascade stage delivers robust predictions since in the second stage only accuracy can be improved. Later, to eliminate false-positive predictions from local appearance feature responses, Urschler et al. [147] integrated spatial configuration of landmarks into a random forest that uses global appearance as well as geometric features. Thus, they mimicked an MRF within a single, unified RRF framework.

3.3.2 Convolutional Neural Networks

Due to their superior image feature extraction capabilities and with the help of large annotated training datasets like ImageNet [119], many tasks in computer vision have recently seen a disruptive shift towards CNNs [79], including landmark localization.

Figure 3.6: Coordinate regression for human pose estimation using CNNs with iterative refinement. The images show the architecture from [145]. The initial stage uses a crop around the person as input to generate the first initial predictions of the target landmarks. The subsequent stages use a smaller crop around the previous landmark predictions to generate coordinate offsets that refine the previous predictions.

The first CNNs for landmark localization directly regressed the coordinates of the landmarks, e.g., Sun et al. [136] for localizing face points and [115, 145] for human pose estimation. Here the outputs of the CNN are directly the coordinates of the landmarks (see Fig. 3.6), i.e., with a d-dimensional image $I : \Omega_I \subset \mathbb{R}^d \rightarrow \mathbb{R}$ as an input the CNN predicts for each of the N landmarks a d-dimensional coordinate vector $\mathbf{x}_i \in \Omega_I$ by modeling the function $f : I \rightarrow \mathbb{R}^{N \cdot d}$. The function is learned with a loss function incorporating a measure of the distance of the target coordinate $\overset{*}{\mathbf{x}}_i$ to the predicted coordinate $\hat{\mathbf{x}}_i$, e.g., the L_2 loss [115]. While these methods doing *coordinate regression* already made use of the powerful feature extraction capabilities of CNNs, directly regressing the coordinates involves a difficult to learn, highly nonlinear mapping from input images I to point coor-

dinates [114]. Hence, such coordinate regression CNNs often suffered from initially bad predictions and needed several refinement steps to make the predictions more accurate. For example, both Sun et al. [136] for face alignment and Toshev and Szegedy [145] for human pose estimation used cascades [27] of three to four carefully designed CNNs. In each step of the cascade, the predictions are iteratively refined based on local regions around the previous predictions (see Fig. 3.6). A disadvantage of such methods is that they highly depend on the success of predictions from all previous stages of the cascade since there is no mechanism to recover from prediction errors. To diminish this limitation, Chen and Yuille [15] proposed to combine coordinate predictions with a graphical model, which led to both more accurate and robust predictions.

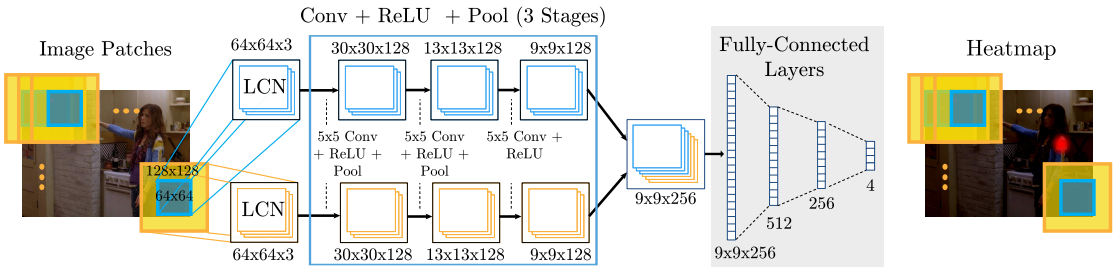
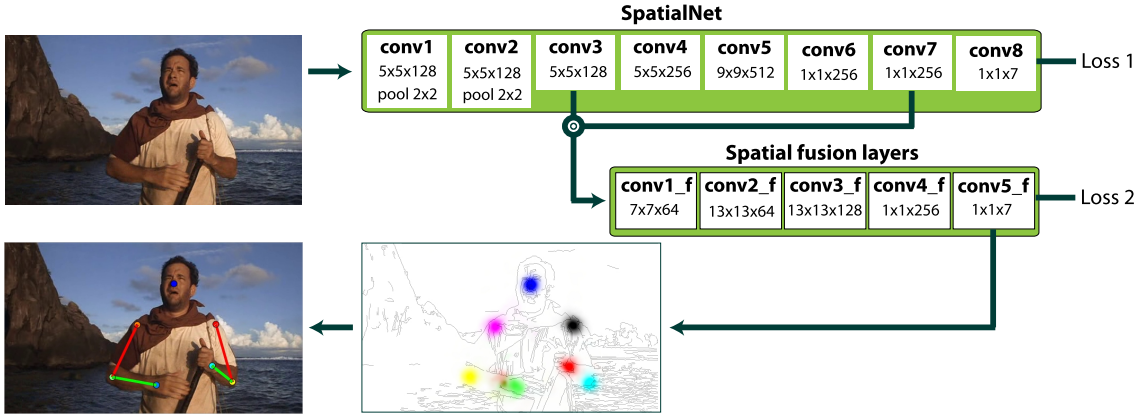


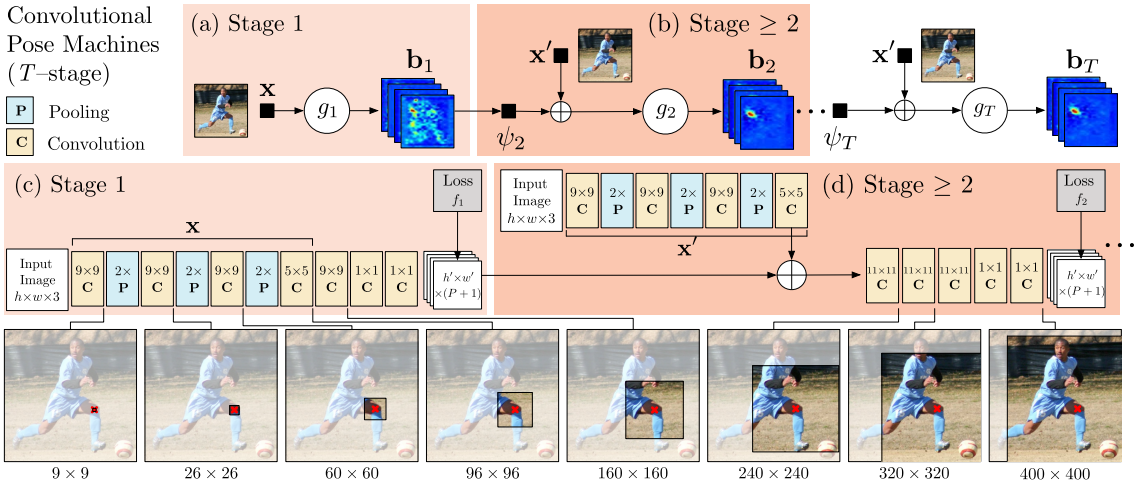
Figure 3.7: Heatmap regression for human pose estimation using CNNs. The images show the architecture from [142]. In a sliding window approach the center pixel of each window is regressed, representing a heatmap of the landmark visualized as a red overlay to the input. For pixels far away from the position of the target landmark, the heatmap values are zero; for pixels near the landmark, the values get higher, while the maximum value is at the exact location of the landmark.

Instead of regressing coordinates, [142] proposed an image-to-image mapping based on regressing *heatmap* images, which encode the pseudo-probability of a landmark being located at a certain pixel position. In [142], the CNN takes a cropped patch of the input image and regresses the pseudo-probability that the pixel on the center is the location of the sought for landmark, i.e., the CNN uses the d-dimensional cropped patch $C \subset I$ from image $I : \Omega_I \subset \mathbb{R}^d \rightarrow \mathbb{R}$ as an input to model the function $f : C \rightarrow \mathbb{R}$. In a sliding window approach, crops from all positions of the image are used to generate the full pseudo-probability heatmap $H : \Omega_I \subset \mathbb{R}^d \rightarrow \mathbb{R}$ of each of the N landmarks. Thus, their network for human pose estimation learns to generate high responses on locations close to the target landmarks, while responses on wrong locations are being suppressed (see Fig. 3.7). The work of Tompson et al. [142] also integrates the binary term of an MRF model inside the CNN architecture. However, they showed shortcomings in both the inefficient sliding window approach using parameter intensive fully-connected layers, as well as the separate training stages of predicting heatmaps and applying the graphical model.

With the rise of *fully convolutional* network architectures [88, 126] enabling efficient image-to-image modeling with CNNs, methods performing heatmap regression also made the switch from CNNs incorporating fully-connected layers to *fully convolutional* CNNs



(a) Network architecture from Pfister et al. [114].

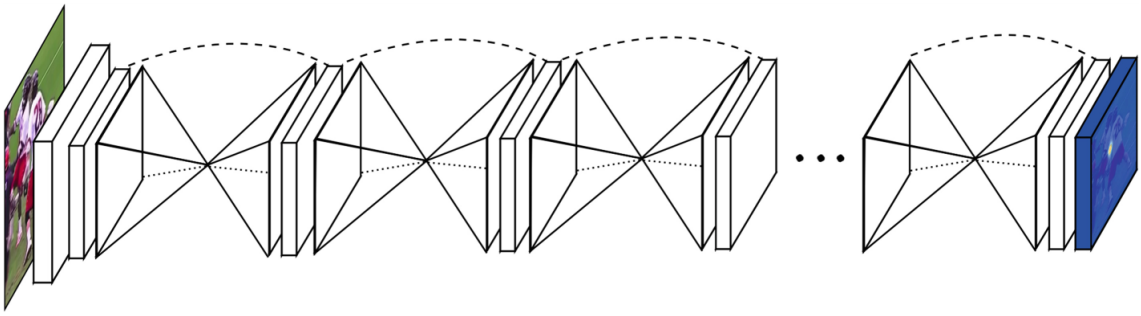


(b) Network architecture from Wei et al. [155].

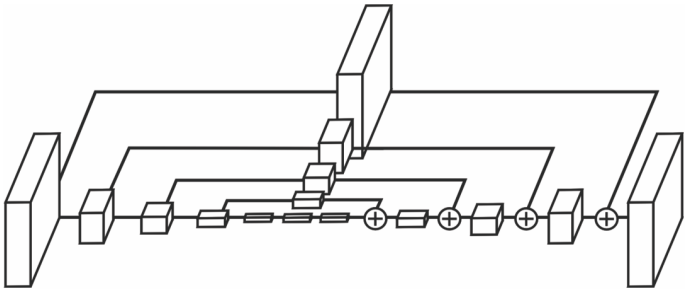
Figure 3.8: Heatmap regression for human pose estimation using fully convolutional CNNs. The images show the architectures from Pfister et al. [114] and Wei et al. [155]. Due to incorporating only two pooling layers, Pfister et al. [114] achieve a large receptive field by convolution operations with large kernel sizes. Wei et al. [155] include more pooling layers, however, both methods do not incorporate upsampling layers, hence reducing the network output resolution.

incorporating only convolution and pooling layers (see Fig. 3.8). In the *fully convolutional* heatmap regression framework, the CNN takes the full input image $I : \Omega_I \subset \mathbb{R}^d \rightarrow \mathbb{R}$ to model the function $f : I \rightarrow H^N$ that predict N heatmaps $H : \Omega_I \subset \mathbb{R}^d \rightarrow \mathbb{R}$ for all landmarks *simultaneously*. Pfister et al. [114] use a fully convolutional network architecture for human pose estimation, which incorporates many convolution layers but only few pooling layers (see Fig. 3.8a). Therefore, to achieve a large receptive field required for observing the whole human pose, they needed to reduce the network input resolution, which hinders high accuracy, while at the same time they needed to use convolution layers with large kernels, which introduces many network parameters. Moreover, they needed to incorporate

a second parallel path in their network with even larger kernel sizes to obtain valid pose predictions. Wei et al. [155] incorporated more pooling layers in their network, thus the input image does not need to be that drastically downsampled (see Fig. 3.8b). However, same as Pfister et al. [114], they also need to incorporate large kernel sizes, while the output resolution of the networks is reduced a lot. Furthermore, both methods of [114, 155] use network architectures that seem ad hoc and unstructured in terms of having different kernel sizes, number of features, and convolution layers.



(a) Schematic representation of stacked hourglass networks for heatmap regression.



(b) Internal representation of a single hourglass network.

Figure 3.9: Heatmap regression for human pose estimation using stacked hourglass networks. The images show the architecture from [100]. In contrast to [114, 155], the network architecture of Newell et al. [100] also incorporates upsampling layers and skip connections, which allow effective modeling of large receptive fields.

A much more elegant solution to enabling large receptive fields is the architecture from [100] based on stacked hourglass networks. Similar to the U-Net [117], the stacked hourglass network consists of several levels with different resolutions. The resolution changes of different levels are performed with downsampling with pooling layers and upsampling with transposed convolution layers, while same-resolution features are combined with skip connections. Therefore, the network input and output may be at the same high resolution, while the network has a large receptive field with a drastically reduced kernel size and number of network parameters. To improve predictions, Newell et al. [100] also incorporate the idea of cascaded pose regression by stacking multiple hourglass networks after each other, thus predicted heatmaps are iteratively refined through the cascade.

While heatmap regression with fully convolutional network architectures is currently

one of the best performing approaches for landmark localization, all aforementioned architectures lack the explicit incorporation of graphical models. However, graphical models could greatly improve the predictions, especially when dealing with limited amounts of training data, as has already been observed in earlier methods for landmark localization (see Section 3.2).

3.4 CNNs in Anatomical Landmark Localization

Despite the usually limited amount of annotated training images in the medical imaging domain, also in anatomical landmark localization, several works showed success in applying CNNs. However, earlier methods in anatomical landmark localization use CNNs not for feature extraction, but for classification. For vertebrae localization and identification, Chen et al. [12] proposed a three-stage framework combining a random forest used for coarse landmark localization, a shape model incorporating the information of neighboring landmarks for refining their positions, and CNNs for identification of landmarks. A drawback of their method is that they solely use two-dimensional CNNs, thus not benefiting from the full potential of volumetric information, which is possible with three-dimensional CNNs. Zhang et al. [169] detect thousands of anatomical landmarks simultaneously from a limited amount of MR volumes of the brain, by first registering all volumes to a template volume, and subsequently regressing all landmark coordinates with a single CNN. However, as discussed previously in Section 3.3.2, regressing the coordinates directly involves a highly nonlinear mapping from input images to point coordinates.

In our preliminary work [106] we introduced the CNN-based heatmap regression framework to anatomical landmark localization. We compared several basic network architectures incorporating only convolution layers, convolution and pooling layers, as well as a U-Net-like architecture incorporating convolution, pooling, and upsampling layers. Additionally, we proposed an initial version of our SpatialConfiguration-Net (SCN) architecture that integrates spatial information of landmarks directly into an end-to-end trained, fully convolutional network (see Chapter 5). There, we showed the potential to achieve good performance even in the presence of very limited amounts of training data. Building upon our proposed approach from [106], Yang et al. [160] used the heatmap regression framework to generate predictions for landmarks with missing responses, by incorporating a pre-trained model of neighboring landmarks into their CNN. However, in contrast to our method, which directly reduces false-positive responses on similar-looking landmarks, their method needs an additional postprocessing step to remove false-positive responses. A different approach to the heatmap regression framework was investigated by Sekuboyina et al. [123]. To reduce the amount of information that needs to be processed, Sekuboyina et al. [123] proposed to project the three-dimensional information of spine anatomy into two-dimensional sagittal and coronal views, and solely use these views as input for their two-dimensional CNN for vertebrae identification and localization. However, due to this projection, beneficial volumetric information may be lost. Bier et al. [8] used heatmap

regression for localizing landmarks on radiographs of the hips. To cope with the small number of images of their training dataset, they implement a sophisticated image synthesis framework based on projecting three-dimensional volumes to two-dimensional images and pretrain their networks with these synthetic images. As they use mostly the same architecture as proposed by Wei et al. [155] without intermediate upsampling layers, the resolution and thus accuracy of their heatmap regression outputs is limited. Mader et al. [91] also adapted the heatmap regression networks for predicting more than a hundred landmarks on a dataset of spinal CT volumes. However, due to memory restrictions resulting from a large number of landmarks, they also do not incorporate upsampling layers and use almost the same architecture as Wei et al. [155], thus reducing the accuracy of their heatmap regression outputs as well. To obtain state-of-the-art results, their method requires to put a separately computed graphical model on top of the network predictions.

While the heatmap regression framework delivers state-of-the-art results in anatomical landmark localization, there exist other recent and successful methods for anatomical landmark localization outside of the heatmap regression framework. For example, aiming for vertebrae identification and localization, Liao et al. [83] proposed an elaborate three-stage method combining several different CNNs. They pretrain a network to classify and localize vertebrae simultaneously, use the learned weights to generate responses with a fully convolutional network, and finally remove false-positive responses with a bidirectional recurrent neural network. Another different strategy for anatomical landmark localization was investigated in [39, 40]. They proposed the use of reinforcement learning to generate navigation trajectories that point towards the sought for landmarks. While their method delivers high landmark localization performance, limitations of their method are the computationally extensive training, especially in the multi-landmark case, as well as the requirement for large annotated datasets.

When looking for CNN-based methods that integrate anatomical constraints for tasks other than anatomical landmark localization, there exist some methods for semantic segmentation, e.g., Oktay et al. [103] regularize solutions with anatomical constraints, and Kamnitsas et al. [66] and Chen et al. [14] integrate pixel-wise graphical models. Such graphical models typically enforce smoothness and connectedness constraints on neighboring pixel output of the CNN, which is a valid constraint for most segmentation tasks, but does not work for anatomical landmark localization, where landmark locations are far apart. Thus, as their direct application to anatomical landmark localization is problematic, we do not intend on adapting these methods, but propose our own network architecture, which incorporates a graphical model that is tuned for anatomical landmark localization.

The aforementioned CNN-based methods for anatomical landmark localization either need lots of training data, sophisticated postprocessing or do not generalize to both two-dimensional and three-dimensional data. During the course of this thesis, we developed and extended a single end-to-end trained fully convolutional network architecture that works well in both two-dimensional and three-dimensional applications with limited amounts of training data, and does not need any dataset specific postprocessing [106–110].

3.5 Our Contributions to Landmark Localization

During the course of this thesis, we have made several contributions and proposed several improvements in the field anatomical landmark localization.

In our preliminary work [106], we adapted heatmap regression with CNNs as proposed in the computer vision community [142] and introduced it for anatomical landmark localization. While Tompson et al. [142] used CNNs intended for classification in a sliding window approach for heatmap regression, in [106], we adapted several fully convolutional network architectures, e.g., the U-Net [117], and evaluated their performance for anatomical landmark localization in both two-dimensional and three-dimensional datasets. In [106], we also proposed a preliminary version of our SCN, which directly includes a graphical model as convolution layers inside the CNN architecture. There, we already showed indications that our SCN can much better cope with a limited number of training images as compared to other fully convolutional architectures.

We extended our preliminary work in [108], where we evaluated our proposed architectures much more comprehensively. There, our improved version of the SCN outperformed other previously proposed methods that are not based on CNNs on all four evaluated two- and three-dimensional dataset, while also outperforming our own U-Net tuned for the landmark localization task. When drastically reducing the amount of training images, we confirmed that our SCN performs much better as compared to our U-Net. Furthermore, we showed that the two components of the SCN, i.e., the *local appearance* and *spatial configuration*, represent a local feature extractor and a graphical model, respectively. In this thesis, we analyze the SCN in even more detail to get an even better understanding of how the SCN performs landmark localization.

Additionally, in [108] we also introduced a loss function for heatmap regression, such that the target heatmap sizes adapt to the uncertainties of the network predictions for each landmark. While we showed in [108] that our loss function produces slightly better results without requiring to tune the heatmap sizes for each landmark individually, in this thesis, we extend the evaluation and show that the heatmap sizes correlate with the landmark localization error, effectively modeling a prediction uncertainty.

Adapted for various anatomical landmark localization tasks, i.e., hand bones [106, 108], cephalograms [108], heart substructures [107], and vertebrae [108, 109] (see Chapter 7), as well as other relevant applications, i.e., face alignment [110], human pose estimation [110], whole heart segmentation [107], and vertebrae segmentation [109] (see Chapter 8), we showed the general applicability of our method, frequently outperforming the previous state-of-the-art.

4 Using CNNs for Landmark Localization

Contents

4.1 CNNs for Landmark Localization 57
4.2 CNNs for Coordinate Regression 58
4.3 CNNs for Regressing Gaussian Heatmaps 59
4.4 Summary 64

> Roads? Where we're going we don't need roads.
>
> *Dr. Emmett Brown*

In this chapter, we explain in more detail the deep learning framework for landmark localization based on heatmap regression. A recap of landmark localization in general is given in Section 4.1. We explain the *coordinate regression* framework for landmark localization in Section 4.2. An overview of using *heatmap regression* with images of Gaussian functions centered at the landmarks' position is given in Section 4.3. In this section, we also introduce our proposed loss function that allows learning of the sigma values of the Gaussian functions, as well as example fully convolutional architectures for heatmap regression. A summary of this chapter is given in Section 4.4.

4.1 CNNs for Landmark Localization

Reiterating the previous chapter, we need to define several terms for landmark localization. At first, we define the input image I as

$$I : \Omega_I \subset \mathbb{R}^d \rightarrow \mathbb{R}, \tag{4.1}$$

with Ω_I being the *support* of the image, i.e., the coordinates on which the image is defined. As in our tasks landmarks may not be outside of the image region, the coordinates of the N specific landmarks $\mathcal{L}_1 \dots \mathcal{L}_N$ are defined as

$$\mathbf{x}_i \in \Omega_I \quad \text{for} \quad i = 1 \dots N. \tag{4.2}$$

The goal of landmark localization is now to determine the coordinates of the landmarks by observing the image I.

Due to their general applicability to various landmark localization tasks, as well as their superior feature extraction capabilities, most state-of-the-art methods of landmark localization use Convolutional Neural Networks (CNNs). Although there exist numerous techniques for landmark localization using CNNs (see Chapter 3), throughout this thesis, we focus on related work on CNNs performing *coordinate regression*, as well as evaluate several architectures and propose a new architecture for CNNs performing *heatmap regression*.

4.2 CNNs for Coordinate Regression

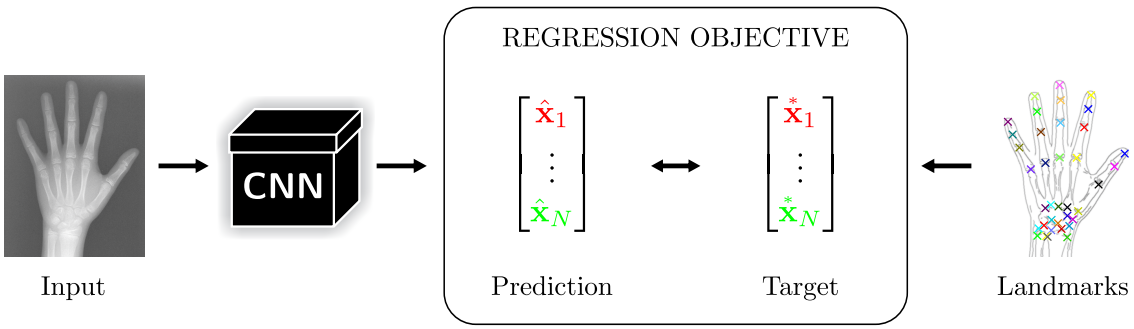


Figure 4.1: Overview of the *coordinate regression* framework for landmark localization. The CNN learns to predict coordinates $\hat{\mathbf{x}}_i$ that resemble the target coordinates $\overset{*}{\mathbf{x}}_i$ as close as possible for all landmarks $\mathcal{L}_i$ with $i = 1 \dots N$ simultaneously.

CNNs performing *coordinate regression* directly predict the N coordinates $\mathbf{x}_i \in \Omega_I$ of the N landmarks as network outputs (see Fig. 4.1). Thus, the CNN models the function $f : I \to \mathbb{R}^{N \cdot d}$, which can be split up into $f_i : I \to \mathbb{R}^d$ for all landmarks $\mathcal{L}_i$ with $i = 1 \dots N$. For a network with weights $\mathbf{w}$ and biases $\mathbf{b}$, the i^{th} network output is directly the predicted coordinate $\hat{\mathbf{x}}_i = f_i(I; \mathbf{w}, \mathbf{b})$. The loss function minimizes a measure of the distance of the target coordinate $\overset{*}{\mathbf{x}}_i$ to the predicted coordinate $\hat{\mathbf{x}}_i$. Typically the L_2 loss function is used within this minimization, which is defined as

$$L_2(I, \overset{*}{\mathbf{x}}_1, \dots, \overset{*}{\mathbf{x}}_N; \mathbf{w}, \mathbf{b}) = \sum_{i=1}^{N} \| f_i(I; \mathbf{w}, \mathbf{b}) - \overset{*}{\mathbf{x}}_i \|_2^2, \tag{4.3}$$

with I being the network input, i.e., input image, $\mathbf{w}$ the network weights, $\mathbf{b}$ the network biases, f_i the i^{th} d-dimensional network outputs corresponding to landmark $\mathcal{L}_i$, $\hat{\mathbf{x}}_i^*$ the corresponding target coordinates, and $\|\cdot\|_2^2$ the squared L_2 norm.

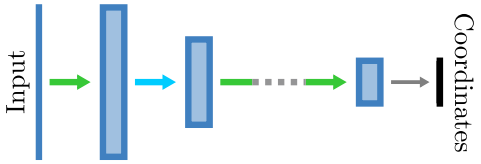


Figure 4.2: Schematic representations of the network architecture for *coordinate regression*. The architecture consists of alternating blocks of convolution and pooling layers, while final fully connected layers create the coordinate predictions. The input of the networks is input images, while the outputs are N coordinate vectors corresponding to the N landmarks. Blue boxes represent (intermediate) images; the black box represents the vector output; arrows represent connections, i.e., → convolution, → downsampling, → fully connected.

CoordReg-Net: For comparisons to other network architectures trained with our framework, in our later experiments we also evaluate a network architecture for coordinate regression. In line with network architectures replacing varying kernel sizes and number of filters [136, 145] with a more streamlined structure [128], our network for coordinate regression uses 3×3 kernels and the same number of feature outputs for every convolution layer. A schematic overview of the structure is shown in Fig. 4.2.

4.3 CNNs for Regressing Gaussian Heatmaps

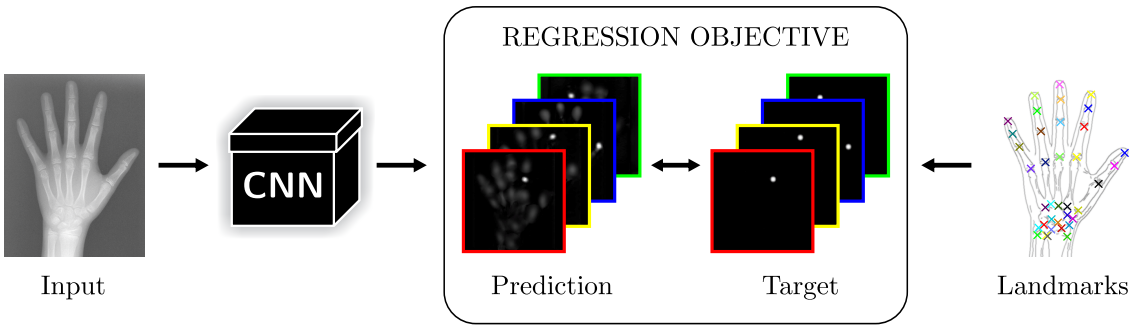


Figure 4.3: Overview of the *heatmap regression* framework for landmark localization. The CNN learns to predict heatmaps $\hat{H}_i$ that resemble the target heatmaps $\hat{H}_i$ as close as possible for all landmarks $\mathcal{L}_i$ with $i = 1 \dots N$ simultaneously.

In contrast to CNNs that directly regress landmark coordinates [145, 169] and require fully connected layers with many network parameters to model the highly nonlinear and difficult to learn image to coordinate mapping, our main method is based on regressing heatmap images [142] (see Fig. 4.3). By enabling an image to image mapping, we benefit

from the use of fully convolutional networks [88, 117, 126], since the number of network weights and thus computational complexity is reduced.

CNNs performing *heatmap regression* do not directly predict the N coordinates $\mathbf{x}_i \in \Omega_I$ of the N landmarks, but a set of N d-dimensional images of heatmaps H_i, i.e., $\mathbb{H} = \{H_i \mid 1 \dots N\}$. Each heatmap H_i for landmark $\mathcal{L}_i$ is defined on the whole region of the input image, i.e.,

$$H_i : \Omega_I \subset \mathbb{R}^d \to \mathbb{R} \quad \text{for} \quad i = 1 \dots N, \tag{4.4}$$

where Ω_I is the support of the image I. At each pixel position $\mathbf{x} \in \Omega_I$, this heatmap encodes a pseudo-probability of the landmark's coordinate $\mathbf{x}_i$ being located at $\mathbf{x}$, i.e.,

$$H_i(\mathbf{x}) = \tilde{p}(\mathbf{x} = \mathbf{x}_i \,|\, I) \quad \text{for} \quad \mathbf{x} \in \Omega_I \quad \text{and} \quad i = 1 \dots N, \tag{4.5}$$

where the pseudo-probability $\tilde{p}$ denotes that the values do not necessarily sum up to one.

Similar to other methods [100, 114, 142, 155] we represent a target heatmap $\overset{*}{H}_i$ with an image of a Gaussian function. For a target landmark $\mathcal{L}_i$ with ground-truth coordinate $\overset{*}{\mathbf{x}}_i$ and heatmap width σ_i, the function that calculates the values of the target heatmap image $\overset{*}{H}_i$ at coordinate $\mathbf{x}$ is defined as the d-dimensional Gaussian function

$$\overset{*}{H}_i(\mathbf{x}) = \frac{\gamma}{(2\pi)^{d/2}\sigma_i^d} \exp\left(-\frac{\|\mathbf{x} - \overset{*}{\mathbf{x}}_i\|_2^2}{2\sigma_i^2}\right) \quad \text{for} \quad \mathbf{x} \in \Omega_I \quad \text{and} \quad i = 1 \dots N. \tag{4.6}$$

Thus, heatmap pixels near the target coordinate $\overset{*}{\mathbf{x}}_i$ have high values, which smoothly but rapidly decrease farther away from $\overset{*}{\mathbf{x}}_i$. We introduce a scaling factor γ to avoid numerical instabilities during training due to otherwise too small values of the Gaussian function. Equal for each dimension d, the standard deviation σ_i defines the peak width of the Gaussian function in the heatmap image for landmark $\mathcal{L}_i$.

In the fully convolutional heatmap regression framework, the network learns functions $f : I \to H^N$ that predict N heatmaps *simultaneously*. Thus, the network outputs can be split up into $f_i : I \to H$ for all landmarks $\mathcal{L}_i$ with $i = 1 \dots N$. For a network with weights $\mathbf{w}$ and biases $\mathbf{b}$, the i^{th} network output is the predicted heatmap $\hat{H}_i = f_i(I; \mathbf{w}, \mathbf{b})$. The networks are trained to predict these N heatmaps by minimizing the pixel-wise differences between the predicted heatmaps $\hat{H}_i(\mathbf{x}) = f_i(I; \mathbf{w}, \mathbf{b})(\mathbf{x})$ and the corresponding target heatmaps $\overset{*}{H}_i(\mathbf{x})$ for all pixels $\mathbf{x} \in \Omega_I$ of image I and for all landmarks $\mathcal{L}_i$ with an L_2 loss:

$$L_{\mathbb{H}}(I, \overset{*}{\mathbf{x}}_1, \dots, \overset{*}{\mathbf{x}}_N; \mathbf{w}, \mathbf{b}) = \sum_{i=1}^{N} \sum_{\mathbf{x} \in \Omega_I} \|f_i(I; \mathbf{w}, \mathbf{b})(\mathbf{x}) - \overset{*}{H}_i(\mathbf{x})\|_2^2. \tag{4.7}$$

4.3.1 Learning Gaussian Heatmaps

Differently to other heatmap based methods (e.g., [142]), we modify the loss function of Eq. (4.7) to treat the heatmap peak widths $\boldsymbol{\sigma} = (\sigma_1, \dots, \sigma_N)$ not as predefined constants but as an unknown parameter of $\overset{*}{H}_i$ to allow $\boldsymbol{\sigma}$ being learned in addition to the network

weights $\mathbf{w}$ and biases $\mathbf{b}$ during training the CNN. Thus, we enable learning of the optimal heatmap peak width separately for each landmark, depending on the prediction confidences of the network. We define our novel objective function that is able to learn also the optimal heatmap peak widths as

$$L_{\mathbb{H}}(I, \overset{*}{\mathbf{x}}_1, \ldots, \overset{*}{\mathbf{x}}_N; \mathbf{w}, \mathbf{b}, \boldsymbol{\sigma}) = \sum_{i=1}^{N} \sum_{\mathbf{x} \in \Omega_I} \| f_i(I; \mathbf{w}, \mathbf{b})(\mathbf{x}) - \overset{*}{H}_i(\mathbf{x}; \sigma_i) \|_2^2 + \alpha \|\boldsymbol{\sigma}\|_2^2 + \lambda \|\mathbf{w}\|_2^2. \quad (4.8)$$

Here, the gradient is not only propagated back into the network output $f_i(I; \mathbf{w}, \mathbf{b})(\mathbf{x})$ but also into the target heatmap function $\overset{*}{H}_i(\mathbf{x}; \sigma_i)$, thus allowing updates of σ_i. The objective function also involves the L_2 norm of the heatmap peak widths $\boldsymbol{\sigma}$ scaled with factor α, as well as an L_2 regularization of the network weights $\mathbf{w}$ scaled with λ (see Section 2.5.1).

Due to first term in Eq. (4.8), i.e., the L_2 distance of $\hat{H}_i(\mathbf{x}) = f_i(I; \mathbf{w}, \mathbf{b})(\mathbf{x})$ and $\overset{*}{H}_i(\mathbf{x}; \sigma_i)$, not only the predicted heatmaps $\hat{H}_i$ aim to be close to the target heatmaps $\overset{*}{H}_i$ but also the target heatmaps $\overset{*}{H}_i$ aim to be close to the predicted heatmaps $\hat{H}_i$. For each landmark $\mathcal{L}_i$, $\hat{H}_i$ and $\overset{*}{H}_i$ receive feedback from each other during training (see Fig. 4.3). While the network parameters ($\mathbf{w}$ and $\mathbf{b}$) are updated to better model the shape of the target heatmap $\hat{H}_i$, at the same time, the peak width parameter (σ_i) of each target heatmap $\overset{*}{H}_i$ is updated to better model the shape of the predicted heatmap $\hat{H}_i$.

The second term in Eq. (4.8) avoids the trivial solution when $\sigma_i \to \infty$ leading to $\overset{*}{H}_i \approx 0$ everywhere, since $\boldsymbol{\sigma}$ are learnable network parameters. The factor α defines how strong the heatmap peak widths $\boldsymbol{\sigma}$ are being penalized.

The first and second term of Eq. (4.8) work against each other. To minimize Eq. (4.8), the former term prefers larger $\boldsymbol{\sigma}$, whereas the latter term urges $\boldsymbol{\sigma}$ to be as small as possible. The network predicts narrower heatmap peak widths for landmarks, where it is confident that the prediction is correct, and wider peak widths for more uncertain landmarks, originating, e.g., from landmarks that are hard to specify exactly. Thus, as shown in Fig. 4.4, the network learns the optimal tradeoff between large σ_i generating oversmoothed, potentially inaccurate predictions, and small σ_i leading to potentially highly accurate responses but with multiple peaks in close proximity. Note that each individual σ_i is learned only from single annotations per image, but from all annotated training images, thus, modeling the ambiguities of landmark annotations for the whole training dataset.

4.3.2 Obtaining the Landmark Coordinates

In inference, the final predicted coordinate $\hat{\mathbf{x}}_i \in \mathbb{R}^d$ of each landmark $\mathcal{L}_i$ needs to be obtained from the heatmap images predicted by the CNN. As the networks were trained to predict Gaussian images with the maximum being at the landmark's coordinate $\overset{*}{\mathbf{x}}_i \in \mathbb{R}^d$, we obtain the predicted coordinate $\hat{\mathbf{x}}_i \in \mathbb{R}^d$ of each landmark $\mathcal{L}_i$ by taking the coordinate

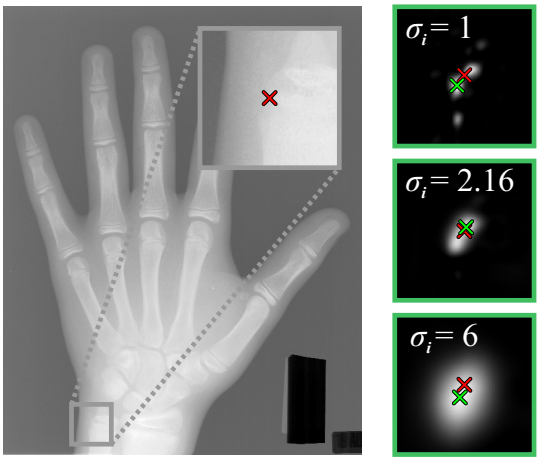


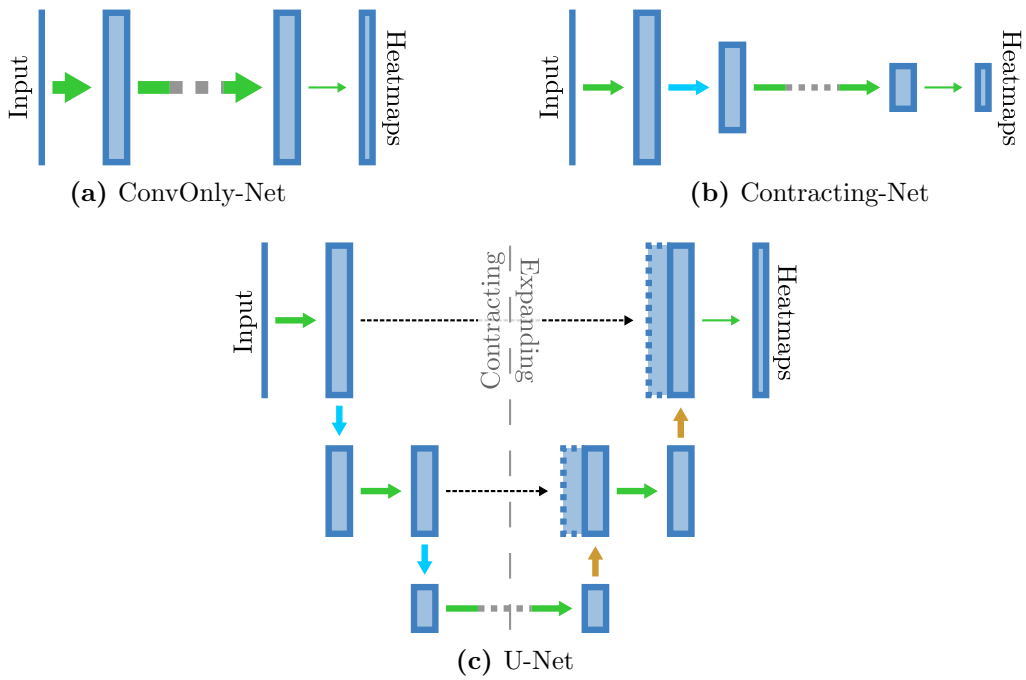
Figure 4.4: Example heatmap output for different σ_i for the zoomed region of landmark $\mathcal{L}_i$. The top image shows multiple peaks when choosing fixed σ_i too small; the middle image shows responses for learned $\sigma_i = 2.16$; the bottom image shows an oversmoothed response for too large fixed σ_i. The target coordinate $\mathring{\mathbf{x}}_i$ is depicted by $\times$, predicted coordinates $\hat{\mathbf{x}}_i$ by $\times$.

where the predicted heatmap has its highest value, i.e., the global maximum response:

$$\hat{\mathbf{x}}_i = \arg\max_{\mathbf{x}} \hat{H}_i(\mathbf{x}) \qquad \text{for} \quad i = 1 \dots N. \tag{4.9}$$

There also exist other methods for obtaining the predicted landmarks coordinate, e.g., taking the center of mass of the heatmap image. However, we found taking the maximum response to be both accurate and robust also in the case when outlier responses exist on the heatmap.

Note that the CNNs may predict maxima with small responses due to missing landmarks that are not present in the image region, or multiple local maxima due to ambiguous structures and annotations. In the first case, a simple postprocessing step that removes landmarks with too small heatmap responses may be used to remove false-positive predictions of landmarks that are not present in the image. In the second case, the postprocessing step is more complicated, as the correct landmark's coordinate is likely to be on either one of the local maxima, or in their vicinity. In previous works, graphical models like Active Shape Models (ASMs) [85] or Markov Random Fields (MRFs) [147] are often used to remove these ambiguities and to predict the most probable response based on statistics of the relative positions of landmarks. However, as described in Chapter 5, we propose to integrate these landmark statistics directly into the network architecture as a *spatial configuration* component. This way, we do not need to perform sophisticated postprocessing of the predicted heatmaps or landmark coordinates, but simply take the global maximum response as in Eq. (4.9) to get the predictions, as our end-to-end trained network for heatmap regression already incorporates a graphical model.



(a) ConvOnly-Net **(b)** Contracting-Net **(c)** U-Net

Figure 4.5: Schematic representations of the network architecture for *heatmap regression*. The architectures consist of only convolution, pooling, and upsampling layers, representing image-to-image networks. The input of the networks are input images, while the outputs are N heatmaps corresponding to the N landmarks. Blue boxes represent (intermediate) images; arrows represent connections, i.e., → convolution, → downsampling, → upsampling.

4.3.3 Fully Convolutional Network Architectures

Inspired by literature, we define several different fully convolutional network architectures for anatomical landmark localization with heatmap regression (see Fig. 4.5 for schematic representations). We defined the architectures such that all are capable of implicitly capturing spatial relationships between landmarks by allowing convolutional filter kernels to cover large image areas.

ConvOnly-Net: The first fully convolutional network architecture is similar to the one of Pfister et al. [114], however we do not incorporate downsampling of intermediate images, neither with pooling nor with strided convolution layers (see Fig. 4.5a). Thus, much larger kernels are needed for observing receptive fields that are large enough for localizing the landmarks, which largely increases the number of network parameters to optimize. Differently to Pfister et al. [114], we do not use arbitrarily different kernel sizes and numbers of feature outputs of the individual layers, but keep these hyperparameters the same for every layer (see Simonyan and Zisserman [128]). We found that using the same kernel size as well as numbers of intermediate outputs reaches the same predictive performance while being more streamlined and easier to follow.

Contracting-Net: This architecture (see Fig. 4.5b) uses alternating convolution and pooling layers, thus, it is similar to the work of [155]. Due to the involved downsampling, the architecture is capable of covering large receptive fields with small kernel sizes. As a drawback of the low resolution of the target heatmaps, lower accuracy in localization has to be expected. We call the layers having the same resolution as being at the same level. Each level consists of two consecutive convolution layers, followed by an average pooling layer with stride two (except the last one). Again, differently to Wei et al. [155], we do not use arbitrarily different kernel sizes and number of feature outputs of the individual layers, but keep these hyperparameters the same for every layer.

U-Net: This architecture is used to represent the group of fully convolutional networks for anatomical landmark localization that also incorporate upsampling layers and skip connections (e.g., [100]). For this purpose, we use a slightly adapted U-Net [117] (see Fig. 4.5c). In contrast to Ronneberger et al. [117], we do not crop intermediate convolution layer outputs, but employ zero padding to keep the input and output size of convolution layers the same. We call the layers having the same resolution as being in the same level. The first half of the U-Net is called the contracting path, as it reduces the resolution of the intermediate outputs. At each level, we employ two consecutive convolution layers, followed by average pooling with stride two to generate the inputs of the next lower level with lower resolution. Different to Ronneberger et al. [117], we replace maximum with average pooling. The second half of the U-Net is called the expanding path, as it increases the resolution of the intermediate outputs. At each level, we employ two consecutive convolution layers, followed by upsampling to generate the inputs of the next upper level with doubled resolution. Instead of learning the deconvolution kernels used for upsampling, we use fixed linear upsampling in the expanding path, thus obtaining a more symmetric architecture. Due to the contracting and expanding path, the network is able to grasp a large image area using small kernel sizes while still keeping a high output accuracy.

4.4 Summary

In this chapter, we have introduced the basic terminology for landmark localization with CNNs. We defined the coordinate regression framework and showed a general network architecture that works within this framework. Typically providing better results than coordinate regression, we also defined the heatmap regression framework, which we mainly use for landmark localization. In the heatmap regression framework, we have introduced our proposed loss function, which allows learning of optimal heatmap sizes. Finally, we have shown fully convolutional network architectures inspired by related work on landmark localization, while our main proposed network architecture, the SpatialConfiguration-Net (SCN), will be explained in the next chapter.

5 Integrating Spatial Configuration into CNNs

Contents

5.1 Fundamental Concept of the SpatialConfiguration-Net 65
5.2 SpatialConfiguration-Net for Heatmap Regression 70
5.3 Summary 72

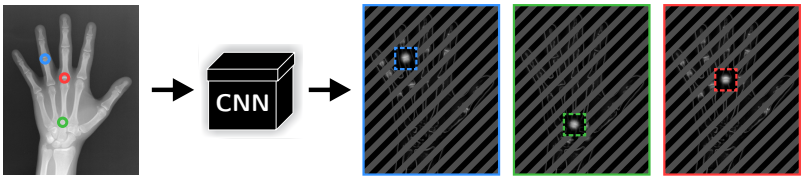
> Please excuse the crudity of this model. I didn't have time to build it to scale or to paint it.
>
> *Dr. Emmett Brown*

Aiming for low landmark localization error in the presence of limited training datasets, in this chapter, we introduce the SpatialConfiguration-Net (SCN), an end-to-end trainable network architecture that integrates a graphical model. The fundamental concept of the architecture that combines the *local appearance* with the *spatial configuration* of anatomical structures is explained in Section 5.1. The specific structure of the SCN for landmark localization tuned to the heatmap regression framework is given in Section 5.2. We summarize the chapter in Section 5.3.

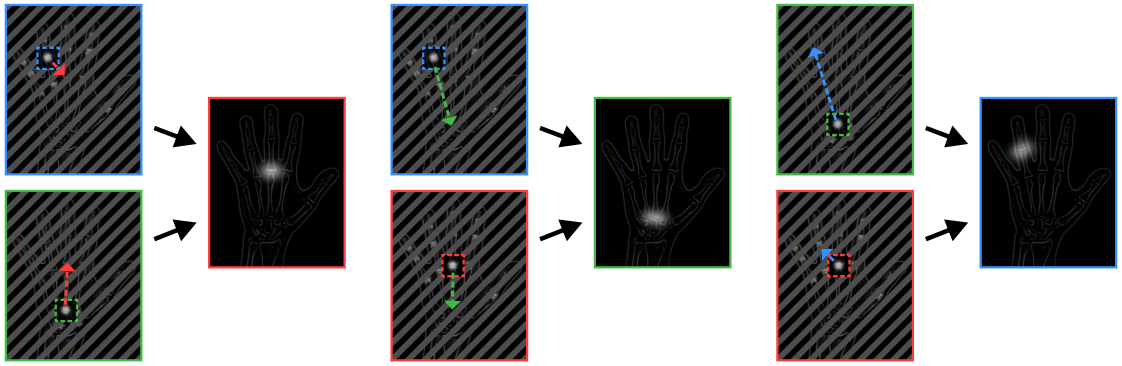
5.1 Fundamental Concept of the SpatialConfiguration-Net

When looking more closely at the individual layers of state-of-the-art deep Convolutional Neural Networks (CNNs) that subsequently apply non-linear filters on images, it can be observed that they model different levels of abstraction [79]. The first CNN layers model low-level pixel features like color, edges, and blobs, while deeper layers model higher-level features of the structure of objects visible in the image [167]. Due to this inherently flexible design, state-of-the-art deep CNNs learn to optimally dedicate each layer to a

specific level of abstraction, but only when trained with enough data. Unfortunately, in the case of small datasets, deep CNNs may not be able to identify both the low-level and high-level features that generalize well for the given task, but rather overfit to the training data. Moreover, a simple reduction of the number of network layers to reduce overfitting is only possible up to a certain extent, as too shallow CNNs may not have enough descriptive power to model the same levels of abstraction as deep CNNs, thus, solving the given task unsatisfactorily [167]. However, previous machine learning approaches have shown that objects with geometric structure can be modeled solely with two levels of abstraction, by combining both handcrafted low-level image feature extractors and handcrafted graphical models representing spatial constraints. We hypothesize that with our specially designed network architecture, the SCN, we can enforce the network to explicitly model these two levels of abstraction, effectively reducing the amount of required training data.



(a) Extract locally accurate heatmap predictions for each landmark, i.e., the *local appearance* heatmaps $\mathbb{H}^{\mathrm{LA}}$.



(b) Transform $\mathbb{H}^{\mathrm{LA}}$ to relative positions of other landmarks to generate the *spatial configuration* heatmaps $\mathbb{H}^{\mathrm{SC}}$.

Figure 5.1: Estimate positions of other landmarks by transforming local predictions. In the heatmap regression framework, a CNN generates local heatmap responses for every landmark. Due to the strong spatial configuration of anatomical landmarks, these local responses may be used to estimate the relative positions of other landmarks, which are indicated with colored arrows. The shaded regions indicate that the responses on the local heatmaps that are far away from the ground-truth positions are irrelevant for transformation.

5.1.1 Transforming Local Responses of Anatomical Structures

The fundamental concept of our SCN is based on the observation that local responses on specific target structures, e.g., heatmap responses for landmark localization, can be used

to estimate the position of other target structures. For example in anatomical landmark localization, as visualized in Fig. 5.1, the predictions of a CNN that performs heatmap regression can be used to estimate the positions of other landmarks. To achieve this, local heatmap responses from every landmark are *moved* or *transformed* to the estimated relative positions of every other landmark. Combining the transformed responses from each other landmark, the position of the target landmark can be approximated well. While such position estimates are not accurate as they are only approximations of the target landmarks' position without directly using input image intensity information, these estimates may be used to constrain the position of target structures to only feasible ones.

One way of modeling such transformations from one response to another is with convolution operations having a kernel that covers the distance of the landmarks. As visualized in Fig. 5.2, the convolution kernel has a response on the expected position of the target landmark, whereas individual estimates from all landmarks are combined via addition. While the convolution kernel may be fixed and calculated from statistics of the positions of the landmarks in the training dataset, the kernel may also be learned within the heatmap regression framework, as the convolution operation is a basic building block of CNNs (see Section 2.4).

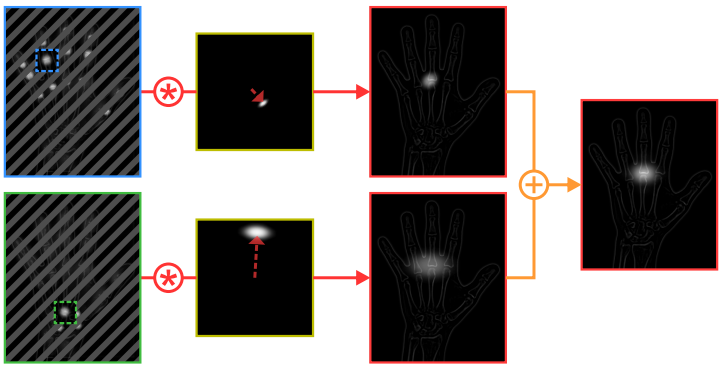


Figure 5.2: Estimate positions of other landmarks with convolutions. The transformations from local heatmap responses (images with green and blue borders) to other landmark positions may be modeled by performing convolutions (visualized as ⊛) with specific kernels (yellow borders). The final heatmap of the estimated position of the landmark is calculated by combining the individual estimations (red border) via addition (visualized as ⊕).

Both initial heatmap responses as well as the transformed heatmaps represent pseudo-probabilities of the landmarks' positions. We call the initial heatmaps the *local appearance* heatmaps $\mathbb{H}^{\mathrm{LA}} = \{H_i^{\mathrm{LA}} \mid i = 1 \dots N\}$, as they depend on the *local appearance* of the input image I, i.e.,

$$H_i^{\mathrm{LA}}(\mathbf{x}) = \tilde{p}(\mathbf{x} = \mathbf{x}_i \,|\, I) \quad \text{for} \quad \mathbf{x} \in \Omega_I \quad \text{and} \quad i = 1 \dots N. \tag{5.1}$$

We call the transformed heatmaps the *spatial configuration* heatmaps $\mathbb{H}^{\mathrm{SC}} = \{H_i^{\mathrm{SC}} \mid i = 1 \dots N\}$, as they encode the *spatial configuration* of landmarks. Since the *spatial configu-*

ration heatmaps do *not* use pixel information from the input image I, but from the *local appearance* heatmaps H_i^{LA} with $i = 1 \dots N$, they are (directly) dependent only on the *local appearance* heatmaps, i.e.,

$$H_i^{\text{SC}}(\mathbf{x}) = \tilde{p}(\mathbf{x} = \mathbf{x}_i \,|\, H_1^{\text{LA}}, \dots, H_N^{\text{LA}}) \quad \text{for} \quad \mathbf{x} \in \Omega_I \quad \text{and} \quad i = 1 \dots N. \tag{5.2}$$

Thus, similar to graphical models like Pictorial Structures (PS) or Markov Random Fields (MRFs) (see Section 3.2), our model based on the heatmap regression framework consists of unary and pairwise terms of landmarks, represented by the *local appearance* heatmaps H_i^{LA} and *spatial configuration* heatmaps H_i^{SC}, respectively. While general fully convolutional CNNs architectures, e.g., the U-Net, may also encode the *spatial configuration* of landmarks *implicitly* in their layers when trained with enough data, our proposed heatmap transformation method *explicitly* encodes the *spatial configuration* of landmarks.

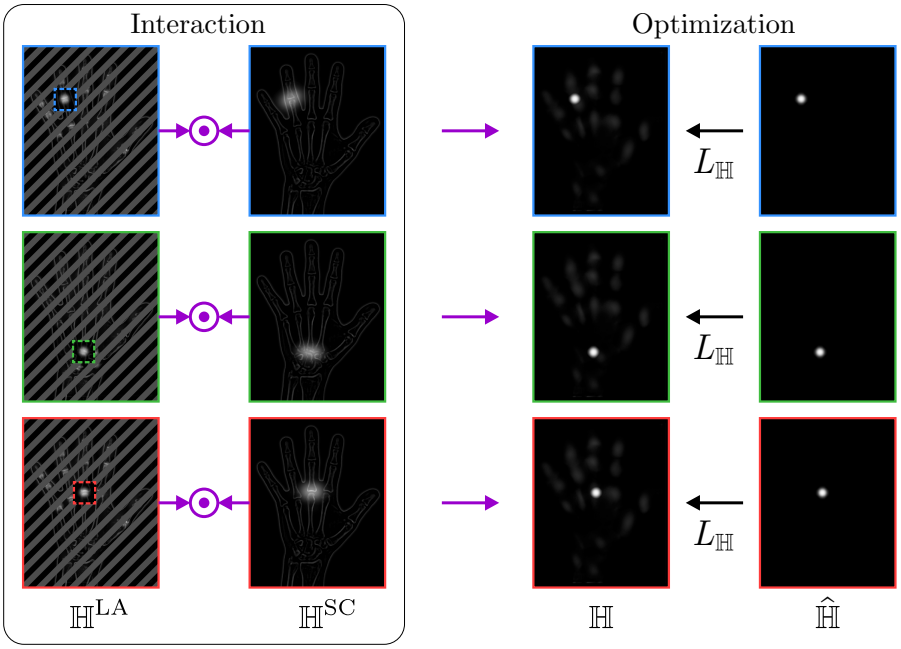


Figure 5.3: Interaction and optimization of the *local appearance* and *spatial configuration* components of the SCN. The loss $L_{\mathbb{H}}$ is only applied to the final heatmaps $\mathbb{H}$, while the *local appearance* and *spatial configuration* components are trained via Backpropagation (BP). Due to the multiplication of the *local appearance* heatmaps $\mathbb{H}^{\text{LA}}$ with the *spatial configuration* heatmaps $\mathbb{H}^{\text{SC}}$ (visualized as $\odot$), the gradient is backpropagated to both components. Thus, the *local appearance* component focuses on creating locally accurate but possibly ambiguous predictions, while the *spatial configuration* component focuses on removing these ambiguities with a graphical model. This interaction of both components leads to a simplification of the localization task into two simpler sub-problems.

5.1.2 Optimization of Local Appearance ⊙ Spatial Configuration

As illustrated in Fig. 5.3, for N landmarks, the set of predicted heatmaps $\mathbb{H} = \{H_i \mid i = 1 \dots N\}$ is obtained by element-wise multiplication $\odot$ of the corresponding heatmap outputs of the two components:

$$H_i = H_i^{\mathrm{LA}} \odot H_i^{\mathrm{SC}}. \tag{5.3}$$

The heatmaps H_i^{LA} and H_i^{SC} are the outputs of the *local appearance* and the *spatial configuration* components for each landmark $\mathcal{L}_i$, respectively.

The SCN is trained for an input image I with corresponding target heatmaps $\mathbb{H}$ in an end-to-end manner. The loss function $L_{\mathbb{H}}$ of Eq. (4.8) is not applied separately to the *local appearance* heatmaps $\mathbb{H}^{\mathrm{LA}}$ and the *spatial configuration* heatmaps $\mathbb{H}^{\mathrm{SC}}$, but only to the final heatmaps $\mathbb{H}$. There is no pretraining of the *local appearance* and *spatial configuration* components of the network, they are jointly optimized. As the final heatmaps are obtained by multiplication of both *local appearance* and *spatial configuration* heatmaps (see Eq. (5.3)), the loss applied to the final heatmaps is backpropagated through both components of the SCN. Thus, both components interact to learn predicting their respective heatmaps, which after multiplying model well the target heatmaps. Due to this interaction, the SCN learns to dedicate its *local appearance* component to deliver locally accurate but potentially ambiguous candidate predictions, and its *spatial configuration* component to focus on improving robustness towards landmark misidentification by eliminating ambiguities. This interaction results in low localization error, i.e., both high robustness towards landmark misidentification and high local accuracy at each identified landmark. Thus, the SCN splits up the localization task into two simpler sub-problems, i.e., generating locally accurate predictions, and removing infeasible solutions with a graphical model.

5.1.3 Interaction of Local Appearance ⇔ Spatial Configuration

The two components interact through the multiplication in Eq. (5.3). This multiplication is crucial for the SCN to learn the simplification of the localization task, as it forces both components to generate a response on the location of the target landmark coordinate $\overset{*}{\mathbf{x}}_i$, i.e., both $H_i^{\mathrm{LA}}(\mathbf{x})$ and $H_i^{\mathrm{SC}}(\mathbf{x})$ deliver responses > 0 for $\mathbf{x}$ close to $\overset{*}{\mathbf{x}}_i$, while on all other locations one component can have a response as long as the other one does not have one. Thus, as long as the *spatial configuration* component does not have a response on locations of locally similar structures, the *local appearance* component can concentrate on transforming the input image to a locally highly accurate response at the location of the target $\overset{*}{\mathbf{x}}_i$, without the need for suppressing locally similar structures. On the other hand, as long as the *local appearance* component generates a locally highly accurate response on $\overset{*}{\mathbf{x}}_i$, the *spatial configuration* component can focus on discriminating locally similar structures by eliminating false-positive responses from the outputs of the *local appearance* component, without requiring to be highly accurate at $\overset{*}{\mathbf{x}}_i$.

Note that also different techniques to multiplication allow interaction of the two components, e.g., addition or convolution. For example, previous works [100, 114, 142] use convolution operations to combine intermediate outputs in a cascade of subsequent CNNs to refine intermediate predictions. However, instead of splitting up the localization tasks as our SCN is doing, these methods aim for robustness towards landmark misidentification as well as local accuracy of the identified landmarks *in each network component simultaneously*. On the other hand, the multiplication of both components inside the SCN is sufficient to enable solutions where both components are facing simpler tasks that can be learned from smaller amounts of training data.

5.2 SpatialConfiguration-Net for Heatmap Regression

Although the two interacting components are in principle flexible regarding their architectures, in Sections 5.2.1 and 5.2.2, we describe our specifically proposed architectures for both components in the heatmap regression framework for landmark localization that we have used in most of the experiments of this thesis (see Chapter 7). Variants and extensions of the SCN used for other applications or tasks like multi-label segmentation are described in more detail in the experimental setup sections of the corresponding experiments in Chapter 8.

5.2.1 Local Appearance

Due to the multiplication in Eq. (5.3), the main focus of the *local appearance* component is to transform the input image I into a set of locally accurate but potentially ambiguous heatmaps $\mathbb{H}^{\text{LA}} = \{H_i^{\text{LA}} \mid i = 1 \dots N\}$. Thus, for each landmark $\mathcal{L}_i$ the *local appearance* component generates the heatmap output $H_i^{\text{LA}}(\mathbf{x})$, resembling the Gaussian target heatmap $\overset{*}{H}_i$ solely in the proximity of the landmark coordinate $\overset{*}{\mathbf{x}}_i$. This is achieved with a multi-level structure that is inspired by fully convolutional networks [117, 126] and the residual network [52]. As shown in the local appearance part of Fig. 5.4, each level consists of several consecutive convolution layers. In the multi-level structure, an average pooling layer connected before the last convolution layer of each level generates the input for the next lower level at half the resolution in our contracting path. In the expanding path, the outputs of each level are linearly upsampled to double the resolution. These upsampled outputs are added to the outputs of the final convolution layer from the next higher level. The outputs of each level represent a residual to the next lower levels, thus an intermediate heatmap is iteratively refined and at the same time resolution is increased until the original resolution is reached again. A last convolution layer at the highest level with the original resolution generates the set of local appearance heatmaps $\mathbb{H}^{\text{LA}}$.

Compared to other fully convolutional network architectures like [117], the expanding path of our *local appearance* component uses fewer parameters making the CNN faster to train.

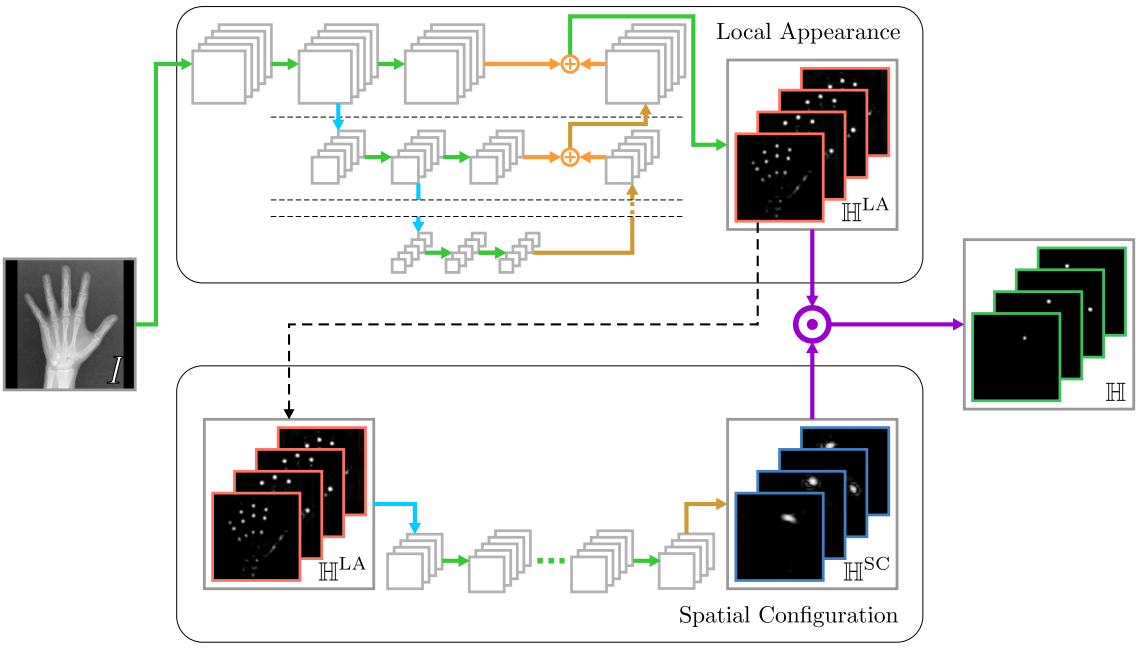


Figure 5.4: Schematic representation of our proposed SCN for landmark localization. In the *local appearance* component, the input image I is transformed into $\mathbb{H}^{\mathrm{LA}}$, representing *local appearance* heatmaps for each of the N landmarks. The dashed black line indicates that $\mathbb{H}^{\mathrm{LA}}$ is used as an input for the *spatial configuration* component, where $\mathbb{H}^{\mathrm{LA}}$ is transformed into the *spatial configuration* heatmaps $\mathbb{H}^{\mathrm{SC}}$. A multiplication of $\mathbb{H}^{\mathrm{LA}}$ and $\mathbb{H}^{\mathrm{SC}}$ results in the final heatmaps $\mathbb{H}$. Blank boxes represent intermediate images; arrows represent connections, i.e., → convolution, → downsampling, → upsampling; ⊕ represents pixel-wise addition, ⊙ pixel-wise multiplication.

5.2.2 Spatial Configuration

Due to the multiplication in Eq. (5.3), the main focus of the *spatial configuration* component is to disambiguate locally accurate but possibly ambiguous heatmaps $\mathbb{H}^{\mathrm{LA}}$ from the *local appearance* component, thus providing robustness towards landmark misidentification for localization. Using only local appearance heatmaps $\mathbb{H}^{\mathrm{LA}}$ as its input, the *spatial configuration* component implicitly incorporates a geometric model of the spatial configuration of landmarks by learning how to robustly predict the position of a single landmark from local position predictions $\mathbb{H}^{\mathrm{LA}}$ of all landmarks. By transforming the whole set of local appearance heatmaps $\mathbb{H}^{\mathrm{LA}}$ into a single heatmap $H_i^{\mathrm{SC}}(\mathbf{x})$ for each landmark $\mathcal{L}_i$, $H_i^{\mathrm{SC}}(\mathbf{x})$ delivers responses on coordinates $\mathbf{x}$ close to the target $\mathbf{x}_i^*$ and suppresses responses elsewhere. Thus, within our *spatial configuration* component, false-positive responses in $H_i^{\mathrm{LA}}(\mathbf{x})$ are suppressed by constraining responses to feasible landmark configurations.

As shown in the spatial configuration part of Fig. 5.4, we model the transformations from $\mathbb{H}^{\mathrm{LA}}$ to $\mathbb{H}^{\mathrm{SC}} = \{H_i^{\mathrm{SC}}(\mathbf{x}) \mid i = 1 \dots N\}$ with consecutive convolution layers. To cover the space among landmarks (see Section 5.1), these convolution layers need to have a large receptive field. As there is no need for high local accuracy in the *spatial configuration* component, the convolution layers can be calculated on a lower resolution compared to

the *local appearance* component, which additionally enables keeping the convolution kernel sizes and the computational complexity reasonably small. After downsampling $\mathbb{H}^{\mathrm{LA}}$, the consecutive convolution layers generate the downsampled version of the heatmap $\mathbb{H}^{\mathrm{SC}}$, which is resized to $\mathbb{H}^{\mathrm{SC}}$ with an upsampling layer. Thus, $\mathbb{H}^{\mathrm{LA}}$ and $\mathbb{H}^{\mathrm{SC}}$ have the same size to enable the element-wise multiplication given in (5.3).

In contrast to both our initially proposed SCN in [106] and the method of Tompson et al. [142], which only allow modeling of pairwise relationships as in the MRF model, multiple convolution layers are able to model more complex relationships between landmarks. This greatly increases the variability of landmark configurations representable by the network. Additionally, multiple layers capture the same receptive field with smaller kernel sizes, thus reducing the number of parameters needed to capture the spatial configuration of landmarks. Therefore, our consecutive convolution layers increase the potential representation capabilities of the network, while keeping network parameters and computational effort low.

5.3 Summary

In this chapter, we have explained the general principle of our proposed SCN architecture. We have described how *local appearance* responses can be transformed to relative positions of other structures by modeling their *spatial configuration*. This transformation can be modeled with convolution operations, thus, it can be directly integrated into a CNN. Our proposed SCN effectively integrates both *local appearance* and *spatial configuration* components into a single CNN, which can be trained in an end-to-end manner. Furthermore, we have explained how both stages interact during training via multiplication. Although the two interacting components of the SCN are in principle flexible regarding their architectures, we have shown specific details tuned for the landmark localization task. In the following chapters, we will give an overview of our used training framework, and perform in-depth analysis of our proposed SCN.

6 Experimental Setup

Contents

6.1 Overview of the Framework 73
6.2 Data Preprocessing and Augmentation 74
6.3 Dataset for Anatomical Landmark Localization 76
6.4 Networks for Anatomical Landmark Localization 79
6.5 Evaluation Metrics 82

> These go to eleven.
>
> *Nigel Tufnel*

In this chapter, we describe the framework and setup used for the experiments within the thesis. A general overview is given in Section 6.1. Our framework is capable of on-the-fly loading and preprocessing two- and three-dimensional images and corresponding annotations (Section 6.2). The evaluated datasets and network architectures for anatomical landmark localization are shown in Section 6.3 and Section 6.4, respectively. Several evaluated performance metrics are described in Section 6.5.

6.1 Overview of the Framework

During the writing of the thesis, we implemented a framework for on-the-fly data augmentation and network training and evaluation. We published the source code of the framework for reproducing the experiments on GitHub[1].

The framework performs on-the-fly data preprocessing and augmentation with SimpleITK [89, 165]. Before feeding an image to the network, it is preprocessed to a

[1] `https://github.com/christianpayer/MedicalDataAugmentationTool`

common representation and augmented with intensity and spatial transformations. For every image, new transformation parameters are randomly sampled from dataset-specific intervals. Thus, the networks practically never see the exactly same input image twice.

The training and testing of the networks presented in this thesis was done with Tensorflow [1]. Our framework allows defining network architectures and running cross-validation experiments. After a certain number of training iterations, the framework performs inference on the validation set to monitor the validation loss and metrics. These metrics are then used for obtaining the optimal hyperparameters for the specific experiment.

The experiments were conducted on several different workstations in parallel, while each workstation can train and evaluate a single network at a time. All workstations were running Arch Linux and used Intel i7 CPUs from generation 4 and 5. The amount of RAM ranged from 16 GB to 32 GB. As for graphics cards, we used NVIDIA GeForce Titan X, NVIDIA GeForce Titan Xp, and NVIDIA GeForce Titan V graphics cards with 12 GB memory. Training of the heatmap regression networks took approximately 4–6 hours for the 2D datasets, and up to 20 hours for the 3D datasets. Predicting landmarks for an unseen image takes from one second (2D) up to 20 seconds (3D) per image, while network inference takes less than a second, and most of the time is spent for image resampling as pre- and postprocessing, which is calculated on the CPU.

6.2 Data Preprocessing and Augmentation

To train Convolutional Neural Networks (CNNs) for specific tasks, both the input data and the corresponding target needs to be preprocessed to be used as input and output of the network. Furthermore, due to the typically small number of training images in medical imaging tasks, an effective data augmentation to increase the variation in the training dataset is crucial for obtaining good results with CNNs (see Section 2.5.3). For our evaluated datasets and tasks, we perform various preprocessing and data augmentation techniques as described in the following.

6.2.1 Intensity Transformations

As network architectures work best with input intensity values near zero, we preprocess the intensity values of the input images such that they are within the range where CNNs work best (see Section 2.3.4). While normalizing each pixel individually with the pixel's mean and standard deviation of all images of the dataset is a potential normalization strategy [114], except for our preliminary work on landmark localization [106], we refrained from using this normalization scheme for two reasons: First, for calculating the mean and standard deviation of the pixel's intensity value, all images that the network will see during training have to be known beforehand. Due to our on-the-fly training data augmentation technique, with our framework seeing all images beforehand is not possible without storing all generated images, which would induce large memory requirements.

Second, pixels in regions that are mostly zero, which is often the case near the border, the standard deviation of the pixel's intensity would be high, leading to large outlier values that hamper the performance of the CNN.

For the aforementioned reasons, we do not perform a pixel-wise normalization, but an image-wise normalization. Thus, we analyze the intensity values of all images of the dataset and process them such that the intensity values of each image lie within approximately $[-1, 1]$. For datasets of images with a physical meaning of the intensity values, e.g., Computed Tomography (CT) images, or where the intensity values are restricted within a certain range, e.g., radiographs of specific parts, we shift and scale the intensity of each pixel/voxel of an image with values derived from the minimum and maximum intensity of the whole dataset. For datasets where each image may be scaled differently, e.g., Magnetic Resonance (MR) images, we calculate a per-image minimum and maximum value for subsequent pixel-/voxel-wise intensity shift and scale. To be more robust in terms of intensity outliers, we do not use the overall minimum and maximum intensity values but take robust measures, e.g., the 1^{st} percentile of all intensities of the image as the minimum and the 99^{th} percentile as the maximum. Furthermore, as such outliers may be counterproductive for the network's performance, we clamp the intensity values to be within a minimum and maximum value.

For training data augmentation, we shift and scale each intensity value of an image by the same random values obtained from uniform distributions within intervals specific to the dataset. For our main anatomical landmark localization experiments, during training, the intensity values are randomly multiplied with $[0.75, 1.25]$ and shifted by $[-0.25, 0.25]$.

6.2.2 Spatial Transformations

Due to the memory requirements of CNNs, the feasible network input image size is restricted by hardware limitations, especially when using volumetric data. Thus, in all our networks, we do not use the input images in their original resolution but preprocess the input images by downsampling them to a size suitable for training the CNNs with the task in mind. For each evaluated dataset, the network input image and output image size in pixels are the same for every image. Input images with physical pixel spacing information are rescaled to a fixed pixel spacing for each image of the dataset, while images without spacing information are rescaled with fixed aspect-ratio to fit the image size of the network inputs. Network outputs are rescaled back to the original input image size to generate the final prediction in original image resolution. Both down- and upsampling utilize bi-/trilinear or bi-/tricubic interpolation.

There are several advantages of working with downsampled images: First, as all input images are resampled to have the same physical pixel size, the networks are trained to work best for this specific pixel spacing. Thus, the variation of the input images is reduced and the task is easier for the CNNs. The individual layers can rely on the always same pixel spacing of the input and do not need to generalize to other pixel spacings. Second,

since due to the resampling the pixels input images are not required to be exactly aligned to the pixels of the network input, many different ways of augmenting data in terms of spatial transformations can be applied. Sub-pixel-wise translation, as well as scaling, and rotation of a single image already leads to a much larger training data variation and networks that better generalize to such transformations. In the medical imaging domain also elastic deformations are of great value, as they model well anatomically and physically plausible deformations, which can further increase the network's generalization capabilities. Finally, downsampling to a lower resolution also reduces high-frequency structures and noise, which are often not required for solving the task in mind but are problematic in terms of overfitting. While many of our CNNs work with a drastically reduced pixel spacing as compared to the original spacing, our experiments show that they are still able to outperform many other methods (see Chapters 7 and 8). Overall, down- or resampling input images to a fixed physical pixel spacing not only reduces the memory requirements for CNNs, but also opens many possibilities for spatially transforming input images for data augmentation to increase generalization.

For data augmentation, we sample values defining translation, scaling, rotation, and elastic deformation from uniform distributions within dataset-specific intervals. The translation and scaling values define the amount of translation and scaling in each dimension, respectively. The rotation values define the amount of rotation around the axis of each dimension. The values for elastic deformation define the displacement in each dimension of each point of a regular grid on the image. The deformation field is then interpolated with 3^{rd} order B-splines. All spatial transformations are combined into a single transformation. Thus, only a single resampling operation is performed, which minimizes resampling artifacts and makes use of as much information of the original input image as possible.

For our main anatomical landmark localization experiments, during training, the images are randomly translated by $[-20, 20]$ pixels, rotated by $[-15°, 15°]$, and scaled by $[-0.6, 1.4]$ per dimension. We additionally employ elastic deformations by randomly moving points on a regular 12×12 (3D: $12 \times 12 \times 12$) pixel grid by 5 pixels and interpolating with 3^{rd} order B-splines. All augmentation operations sample randomly from a uniform distribution within the specified intervals.

6.3 Dataset for Anatomical Landmark Localization

In our main evaluations in Chapter 7, we compare our proposed SpatialConfiguration-Net (SCN) architecture to various state-of-the-art localization methods on four datasets from the medical imaging domain, i.e., radiographs of left hands (2DHand), volumetric MR scans of left hands (3DHand), lateral cephalograms (2DSkull), and volumetric CT scans of the spine (3DSpine). Fig. 6.1 shows representative examples for these datasets.

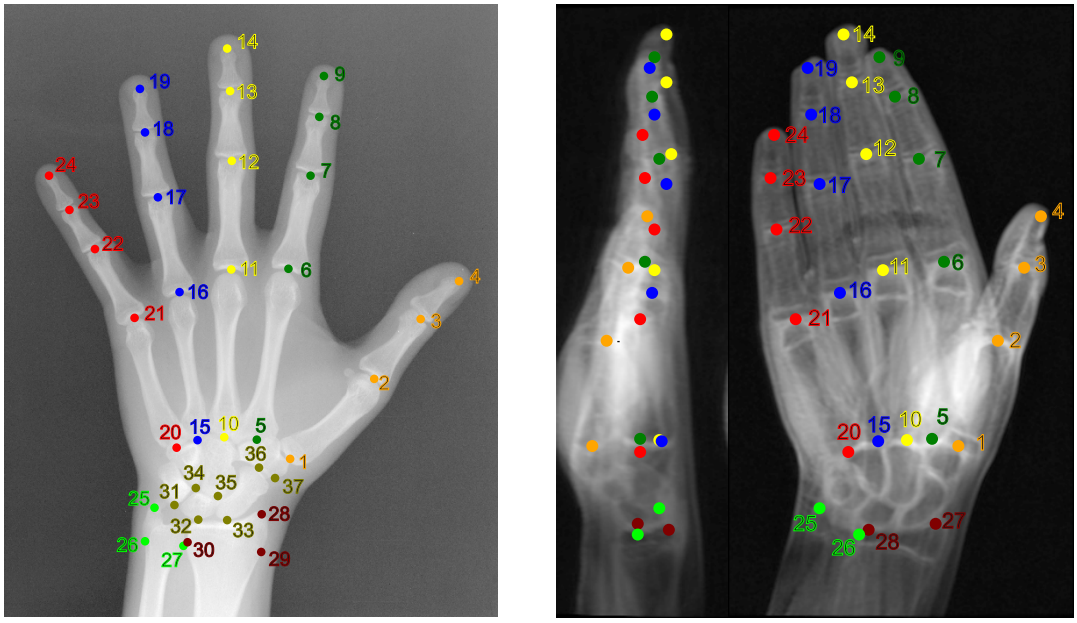
Hand Radiographs (2DHand): We use a publicly available dataset of hand radiographs of the Digital Hand Atlas Database System [38, 168] for comparison with state-of-the-art anatomical landmark localization methods and to investigate in detail several

hyperparameters of our SCN. The dataset consists of 895 radiographs of left hands with an average size of 1563 × 2169 pixels, acquired with different X-ray scanners. We performed a manual annotation of 37 characteristic landmarks on fingertips and bone joints. As the images lack information about physical pixel resolution, we calculate an image-specific normalization factor $s^{(j)}$. Using the normalization of [85, 130, 147], we assume a wrist width of 50 mm determined by two of the annotated landmarks at the wrist, i.e., $s^{(j)} = 50/\|\mathbf{x}^{*(j)}_{\text{l\_wrist}} - \mathbf{x}^{*(j)}_{\text{r\_wrist}}\|_2$. We use the same three-fold cross-validation setup as [130, 147], i.e., the 895 input images are split into three folds with an equal number of images, resulting in approximately 600 training and 300 testing images per fold. Moreover, we use the same preprocessing of the input images as [130], by performing histogram matching to a reference image solely inside the outline of the hand as computed by Otsu thresholding.

Hand MRIs (3DHand): To show the applicability of our SCN to volumetric MR data, we use an in-house dataset of 60 T1-weighted 3D gradient-echo hand Magnetic Resonance Imaging (MRI) scans with 28 annotated landmarks, which is intended for automatic forensic age estimation [134] (see Section 8.2). The average volume size is $294 \times 512 \times 72$ with a voxel resolution of $0.45 \times 0.45 \times 0.9$ mm^3. We use the same five-fold cross-validation setup as used in other works of our group [31, 106, 147], with each fold consisting of 43 training and 17 testing images. As we know the physical voxel resolution of each image, $s^{(j)}$ is set to 1.

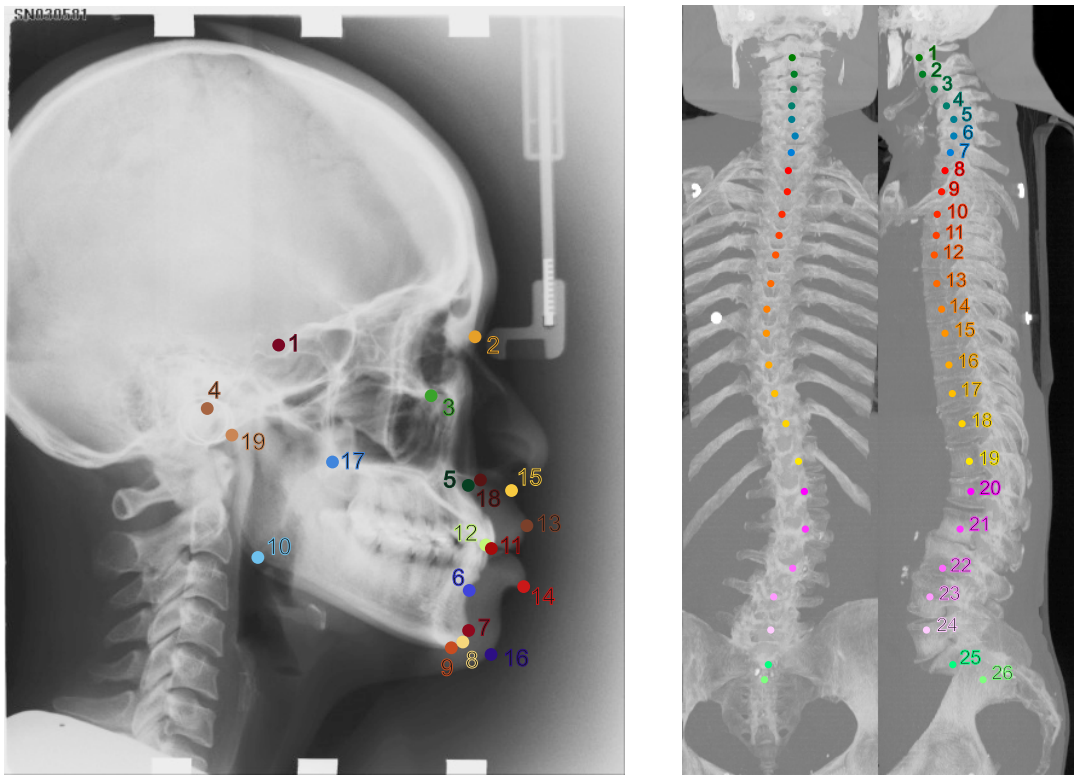
Lateral Cephalograms (2DSkull): To compare our proposed SCN in the context of a landmark localization challenge, we apply it to the publicly available dataset that was used for the ISBI 2015 Cephalometric X-ray Image Analysis Challenge [152]. It consists of 400 lateral cephalograms from 400 different subjects, with 19 annotated landmarks. The 2D images have a size of 1935 × 2400 with a physical resolution of 0.1×0.1 mm^2 per pixel. We use the train/test split (150 training, 150 *Set1* + 100 *Set2* testing images) and evaluation protocol described in [152]; $s^{(j)}$ is set to 1.

Spine CTs (3DSpine): To compare our proposed SCN with other recent methods, we also evaluate on a publicly available volumetric dataset of pathological spine CT scans [44] used for the MICCAI CSI 2014 Vertebrae Localization and Identification Challenge. This dataset includes various challenging cases such as scoliotic spines, vertebrae fractures, metal insertions causing severe image artifacts, and highly restrictive field of views. In line with previously reported results, we split the evaluation on this dataset into two sets: In *Set1*, the 224 CT scans of the challenge training set are evaluated with two-fold cross-validation. In *Set2*, the networks are trained on all 224 CT scans plus additional 18 CT scans and evaluated on the challenge test set, which consists of 60 CT scans. The average volume size is $512 \times 512 \times 160$ with a voxel resolution of $0.34 \times 0.34 \times 2.06$ mm^3. Due to known physical voxel resolution of each image, $s^{(j)} = 1$.



(a) Hand radiographs (2DHand)

(b) Hand MRI (3DHand)



(c) Lateral Cephalograms (2DSkull)

(d) Spine CT (3DSpine)

Figure 6.1: Example images of the evaluated datasets. Volumes are maximum-projected to 2D coronal and sagittal views for visualization. Circles and numbers indicate landmark annotations.

6.4 Networks for Anatomical Landmark Localization

In addition to comparisons with state-of-the-art methods on the previously described datasets, in Chapter 7, we evaluate several network architectures inspired from literature for anatomical landmark localization (see Sections 4.2 and 4.3).

We determine the number of solver iterations for each evaluated dataset by training the network on 80% of the training images and using the remaining 20% of the training images as a validation set to assess when the validation error has reached a plateau. For the datasets evaluated with cross-validation, we determine the number of training iterations with initial experiments by evaluating the validation loss on the first fold. All folds of the dataset are then trained for the determined number of iterations.

CoordReg-Net: For comparing a network performing coordinate regression trained with our framework to other networks performing heatmap regression, we set up a network architecture as described in Section 4.2. The network uses 256×256 images as input and it consists of four levels. Each level consists of two consecutive convolution layers with kernel size 3×3 and 128 outputs, followed by a 2×2 max pooling layer (except at the last level). Then, to model the non-linear mapping from image information to coordinates, two fully connected layers with 256 features and a fully connected layer with $d \cdot N$ outputs generate the d-dimensional coordinate prediction of all N landmarks. All convolution and fully connected layers, except the last one, use the Rectified Linear Unit (ReLU) activation function. The biases are initialized with zeroes, the weights are initialized with the method of [51]. For regressing the coordinates, they are normalized such that the coordinate 0×0 corresponds to the image coordinate 0×0, and that the coordinate 1×1 corresponds to the image coordinate 256×256. We train the network for 100,000 iterations with a mini-batch size of one and the Adam optimizer with default parameters [71] and a learning rate of 10^{-4}.

ConvOnly-Net: This is our first described network that performs heatmap regression (see Section 4.3.3). The network uses 128×128 images as input while it consists of five convolution layers with 11×11 kernels and 128 filter outputs, and an additional 11×11 convolution layer with N outputs to generate the predicted heatmaps for the N corresponding landmarks. All convolution layers use the ReLU activation function, except the last one, which uses no activation function. The biases are initialized with zeroes, the weights by drawing values from a Gaussian distribution with a standard deviation of 0.01. As this network was only used for initial experiments, we did not employ our proposed loss function of Eq. (4.8) that allows learning the optimal heatmap peak widths $\boldsymbol{\sigma}$, but the default L_2 loss function of Eq. (4.7) as also used by other methods [100, 114, 142, 155]. We train the network for 20,000 iterations with a mini-batch size of five and the Stochastic Gradient Descent (SGD) optimizer with Nesterov's momentum [98] of 0.99 and a learning rate of 10^{-5}.

Contracting-Net: The network uses 256×256 images as input while it consists of three levels. Each level consists of two consecutive convolution layers with kernel size 5×5

and 128 outputs, followed by a 2×2 average pooling layer (except at the last level). The very last convolution layer has N outputs to generate the predicted heatmaps for the N corresponding landmarks. The activation functions of the convolution layers, the initial weights, as well as the loss function, are set up the same as for the previous network. We train the network for 20,000 iterations with a mini-batch size of five and the SGD optimizer with Nesterov's momentum [98] of 0.99 and a learning rate of 10^{-5}.

U-Net (from Payer et al. [106]): The network uses 256×256 images as input while it consists of four levels. Both blocks in the contracting and the expanding path consist of two convolution layers with 3×3 kernels and 128 outputs, except the last one having N heatmap outputs for the N corresponding landmarks. In the contracting path, the next lower level is generated with 2×2 average pooling; in the expanding path, the next higher level is generated with bilinear interpolation. The activation functions of the convolution layers, the initial weights, as well as the loss function, are set up the same as for the previous networks. This U-Net is trained for 40,000 iterations with a mini-batch size of five and the SGD optimizer with Nesterov's momentum [98] of 0.99 and a learning rate of 10^{-6}.

SCN (from Payer et al. [106]): The network uses 256×256 images as input. The *local appearance* stage of the network consists of three convolution layers with 5×5 kernels followed by the *spatial configuration* stage with a downsampling factor of $1/8$. Differently to our later used SCN architectures, the *spatial configuration* block consists of only a single convolution layer with 15×15 kernels. Furthermore, the convolution kernel generating H_i^{SC} does not take H_i^{LA} but every other heatmap output from the *local appearance* stage as input, i.e., the input is the set $\{H_j^{\mathrm{LA}} \mid j = 1 \ldots N \wedge i \neq j\}$. The activation functions of the convolution layers, the initial weights, as well as the loss function, are set up the same as for the previous networks. This SCN is trained for 40,000 iterations with a mini-batch size of five and the SGD optimizer with Nesterov's momentum [98] of 0.99 and a learning rate of 10^{-6}.

U-Net (from Payer et al. [108]): This architecture is similar to the previously described U-Net as used in Payer et al. [106], but with minor differences. This U-Net uses larger images as input (specific sizes are mentioned in the individual experiments) and it is set up with 5 levels (3D: 4) and 128 outputs (3D: 64) for all intermediate convolution layers. Further increasing the number of levels and convolution outputs, i.e., the depth of the U-Net, showed no improvements. As proposed by Ronneberger et al. [117], dropout [129] is employed in the deepest two levels. The biases of the convolution layers are initialized with 0, the weights with the method described in [51], except the final convolution layer generating heatmaps, where the weights are drawn from a Gaussian distribution with a standard deviation of 0.001. These small weights are needed to generate initial heatmap responses close to 0, as otherwise, the network training would not converge. In contrast to the previously described networks for heatmap regression, we use the objective function with trainable heatmap peak widths $\boldsymbol{\sigma}$ (see Eq. (4.8)) with parameters $\gamma = 100$ (3D: $\gamma = 1000$), $\alpha = 20$ (3D: $\alpha = 1000$), and $\lambda = 0.0005$. These parameters were determined

as described in subsequent chapter. The network uses input image sizes of 512×512 for 2DHand and 2DSkull, $96 \times 128 \times 32$ for 3DHand, and $96 \times 96 \times 192$ for 3DSpine. The network is trained with a mini-batch size of one, and for 30,000 iterations in the 2D datasets, for 20,000 iterations for the 3DHand, and 40,000 iterations for the 3DSpine dataset. We employ the SGD optimizer with Nesterov's momentum [98] of 0.99 and a learning rate of 10^{-6}.

SCN (from Payer et al. [108]): Our main architecture, the SCN as described in Chapter 5, which is tuned to the heatmap regression task. The networks use input image sizes of 512×512 for 2DHand and 2DSkull, $96 \times 128 \times 32$ for 3DHand, and $96 \times 96 \times 192$ for 3DSpine. The *local appearance* component has four levels, each consisting of three consecutive 3×3 (3D: $3 \times 3 \times 3$) convolution layers having 128 outputs (3D: 64), while a 2×2 (3D: $2 \times 2 \times 2$) average pooling after the second convolution layer generates the level below. We include dropout [129] of 0.5 after the first convolution layer at each level to improve generalization. The *local appearance* heatmaps $\mathbb{H}^{\text{LA}}$ are generated from a 3×3 (3D: $3 \times 3 \times 3$) convolution layer having a number of outputs equal to the number of landmarks. The *spatial configuration* component is calculated at $1/16$ (3D: $1/4$) of the input resolution. It consists of three consecutive 11×11 (3D: $7 \times 7 \times 7$) convolution layers with 128 outputs and an additional 11×11 (3D: $7 \times 7 \times 7$) convolution layer having a number of outputs equal to the number of landmarks. These outputs are upsampled back to the input resolution with bi-/tricubic interpolation to generate $\mathbb{H}^{\text{SC}}$. Each intermediate convolution layer of the whole SCN has a Leaky ReLU [90] activation function with a negative slope of 0.1 to ease convergence at training. The convolution layer generating $\mathbb{H}^{\text{LA}}$ has no activation function; the convolution layer generating $\mathbb{H}^{\text{SC}}$ has a tanh activation function to restrict the outputs between -1 and 1. The weights are initialized with the method described in [51], except the layers generating heatmaps $\mathbb{H}^{\text{LA}}$ and $\mathbb{H}^{\text{SC}}$. Here, initial weights are drawn from a Gaussian distribution with a standard deviation of 0.001. These small weights are needed to generate initial heatmap responses close to 0, as otherwise, network training would not converge. Same as for the aforementioned U-Net, we use the objective function of Eq. (4.8) that allows learning the optimal heatmap peak widths $\boldsymbol{\sigma}$. We set up the loss function with parameters $\gamma = 100$ (3D: $\gamma = 1000$), and $\lambda = 0.0005$ that were empirically determined during initial experiments. To determine parameter $\alpha = 20$ we perform an experiment on our hand radiograph dataset that is described in Section 7.2. For the 3D datasets, we set $\alpha = 1000$, which we determined empirically. Same as for the U-Net, the SCN is trained with a mini-batch size of one, and for 30,000 iterations in the 2D datasets, for 20,000 iterations for the 3DHand, and 40,000 iterations for the 3DSpine dataset. We employ the SGD optimizer with Nesterov's momentum [98] of 0.99 and a learning rate of 10^{-6}.

6.5 Evaluation Metrics

The evaluated metrics for anatomical landmark localization as used in Chapters 7 and 8 and semantic segmentation as used in Chapter 8 are described in this section.

6.5.1 Landmark Localization Evaluation Metrics

The performance of landmark localization methods is evaluated with several commonly used metrics from the literature, describing localization error in terms of both local accuracy and robustness towards landmark misidentification.

The point-to-point error for each landmark $\mathcal{L}_i$ in image j is defined as the Euclidean distance between the target coordinate $\mathbf{x}_i^{*(j)} \in \mathbb{R}^d$ and the predicted coordinate $\hat{\mathbf{x}}_i^{(j)} \in \mathbb{R}^d$. To compensate for unknown or varying scale in the datasets, the point-to-point error may be multiplied with an image-specific normalization factor $s^{(j)}$, based on the Euclidean distances of specifically selected landmarks. For landmark $\mathcal{L}_i$ in an image j, the normalized point-to-point error is defined as

$$\mathrm{PE}_i^{(j)} = s^{(j)} \|\mathbf{x}_i^{*(j)} - \hat{\mathbf{x}}_i^{(j)}\|_2. \tag{6.1}$$

This allows calculation of sets of point-to-point errors, e.g., for all images of the dataset $\mathcal{D}$ over a specific landmark $\mathcal{L}_i$ only, i.e.,

$$\mathrm{PE}_i = \{\mathrm{PE}_i^{(j)} \mid j \in \mathcal{D}\}, \tag{6.2}$$

or over all landmarks L, i.e.,

$$\mathrm{PE}_{\mathrm{all}} = \{\mathrm{PE}_i^{(j)} \mid i \in \mathcal{L}, j \in \mathcal{D}\}, \tag{6.3}$$

on which several measures (e.g., mean, standard deviation, median) can be calculated.

Computed for all landmarks of a single image j, we define the image-specific point-to-point error $\mathrm{IPE}^{(j)}$ as

$$\mathrm{IPE}^{(j)} = \frac{1}{N} \sum_{i=1}^{N} \mathrm{PE}_i^{(j)}, \tag{6.4}$$

where N is the number of landmarks. We also present plots of the cumulative IPE distributions, which visualize the proportion of tested images that achieve a certain IPE.

We also report the number of predicted landmarks $\mathcal{L}_i$ that are outside a certain point-to-point error radius r for all images, i.e., number of outliers

$$\#\mathrm{O}_r = \left|\{(j,i) \mid \mathrm{PE}_i^{(j)} > r\}\right|. \tag{6.5}$$

To compare to other methods on the datasets of spine CT scans, we calculate the number of correctly identified landmarks. As defined by [44], a predicted landmark is correctly

identified, if the closest ground-truth landmark is the correct one, and the distance from predicted to ground-truth position is less than 20 mm. The number of correctly identified landmarks is defined as

$$\#\text{ID} = \left|\{(j,i) \mid \arg\min_k \|\mathbf{x}_k^{*(j)} - \hat{\mathbf{x}}_i^{(j)}\|_2 = i \ \wedge \ \text{PE}_i^{(j)} \leq 20\}\right|. \tag{6.6}$$

For comparison with other methods, the ID_{rate} is defined as the percentage of correctly identified landmarks over all landmarks.

6.5.2 Segmentation Evaluation Metrics

The segmentation performance is evaluated with several commonly used metrics from the literature, measuring the overlap, as well as the maximum surface distance between ground-truth and predict segmentation label.

A measure that is based on the overlap of ground-truth and prediction label is the Dice Similarity Coefficient (DSC). For a d-dimensional segmentation image $S : \Omega_S \subset \mathbb{R}^d \to \{0 \dots N\}$ with N labels, the set of pixels $\mathbf{x} \in \Omega_S$ of image S having segmentation label i is defined as $\mathbb{S}_i = \{\mathbf{x} \mid S(\mathbf{x}) = i\}$. The DSC for label i and image j with ground-truth set of segmentation pixels $\mathbb{S}_i^{*(j)}$ and the prediction set of segmentation pixels $\hat{\mathbb{S}}_i^{(j)}$ is defined as twice the cardinality of the intersection of ground-truth and prediction divided by the sum of the cardinalities of both ground-truth and prediction, i.e.,

$$\text{DSC}_i^{(j)} = \frac{2\left|\mathbb{S}_i^{*(j)} \cap \hat{\mathbb{S}}_i^{(j)}\right|}{\left|\mathbb{S}_i^{*(j)}\right| + \left|\hat{\mathbb{S}}_i^{(j)}\right|}. \tag{6.7}$$

The Generalized Dice Similarity Coefficient (gDSC) for multi-label segmentations of image j takes the different sizes of labels into account and is defined as

$$\text{gDSC}^{(j)} = \frac{2\sum_{i=1}^{N}\left|\mathbb{S}_i^{*(j)} \cap \hat{\mathbb{S}}_i^{(j)}\right|}{\sum_{i=1}^{N}\left(\left|\mathbb{S}_i^{*(j)}\right| + \left|\hat{\mathbb{S}}_i^{(j)}\right|\right)}. \tag{6.8}$$

Another measure for the segmentation performance is the Hausdorff distance $\mathcal{H}_i^{(j)}$. For label i and image j with ground-truth set of segmentation pixels $\mathbb{S}_i^{*(j)}$ and the prediction set of segmentation pixels $\hat{\mathbb{S}}_i^{(j)}$, the Hausdorff distance is defined as the largest of all Euclidean distances from a point in one set to the closest point in the other set, i.e.,

$$\mathcal{H}_i^{(j)} = \max\left(\max_{\mathbf{a}\in\mathbb{S}_i^{*(j)}}\min_{\mathbf{b}\in\hat{\mathbb{S}}_i^{(j)}}\|\mathbf{a}-\mathbf{b}\|_2, \max_{\mathbf{a}\in\hat{\mathbb{S}}_i^{(j)}}\min_{\mathbf{b}\in\mathbb{S}_i^{*(j)}}\|\mathbf{a}-\mathbf{b}\|_2\right). \tag{6.9}$$

Similar to the measures for localization, the individual segmentation measures can be calculated for specific labels or over all images to generate sets on which various statistical measures (e.g., mean, standard deviation, median) can be calculated.

7 Anatomical Landmark Localization Results

Contents

7.1 Comparison to the State-Of-The-Art 85
7.2 Learning the Size σ of the Gaussian Heatmaps 98
7.3 Interacting Local Appearance $\Leftrightarrow$ Spatial Configuration 103
7.4 Reducing the Number of Training Images 109
7.5 Summary 111

> I am serious. And don't call me Shirley.
>
> *Dr. Rumack*

In this chapter, we show and discuss several results of our methods for anatomical landmark localization. We perform comparisons of our proposed SpatialConfiguration-Net (SCN) to the state-of-the-art on various datasets for anatomical landmark localization in Section 7.1. We show an in-depth analysis of different aspects of the proposed loss function (see Chapter 4) and SCN (see Chapter 5) in Section 7.2 and Section 7.3, respectively. As a final experiment, we evaluate how well our proposed SCN performs with a drastically reduced number of training images in Section 7.4, before summarizing in Section 7.5.

7.1 Comparison to the State-Of-The-Art

Most of the results as discussed within this chapter are from our initially published work at MICCAI 2016 [106], as well as our extended MIA paper from 2019 [108]. As our preliminary results of [106] have shown that CoordReg-Net, ConvOnly-Net, and Contracting-Net have several drawbacks, most of the experiments show comparisons of the U-Net and the SCN only (as published in [108]).

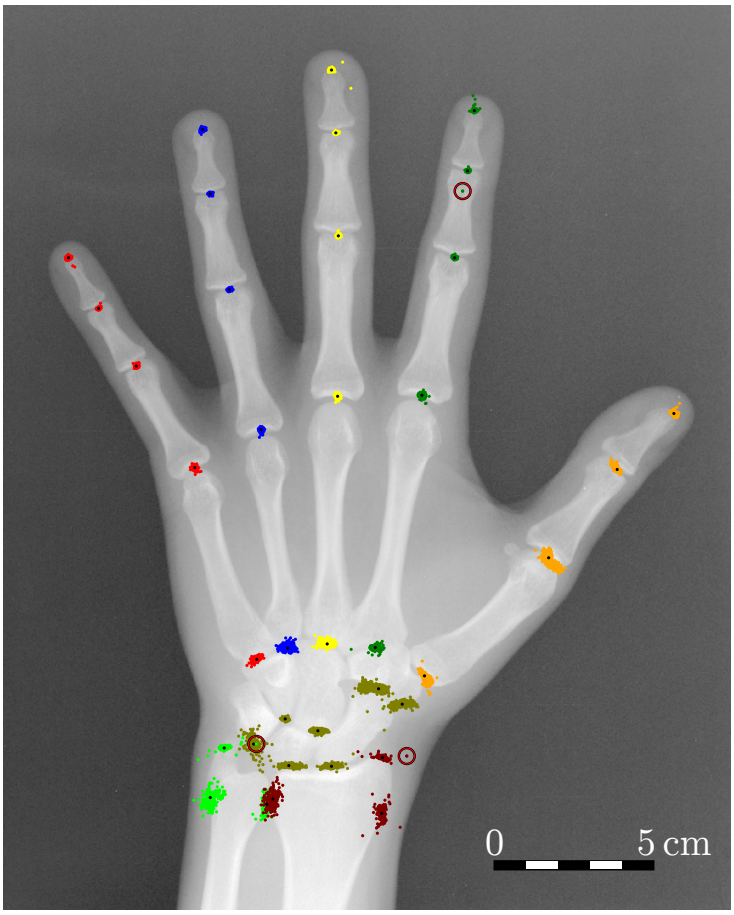


Figure 7.1: Example image with all from the SCN predicted landmarks of all images from the 2DHand dataset. Using an image and its ground-truth landmarks (black dots) as the base, all predicted landmarks of all images are shifted by the offsets to the corresponding ground-truth. The three predicted landmarks that are marked with a red circle are the only three outliers having more than 10 mm distance to the ground-truth.

In our main experiments, we compare our proposed SCN architecture to various state-of-the-art localization methods on four datasets from the medical imaging domain (see Section 6.3), i.e., radiographs of left hands (2DHand in Section 7.1.1), volumetric Magnetic Resonance (MR) scans of left hands (3DHand in Section 7.1.2), lateral cephalograms (2DSkull in Section 7.1.3), and volumetric Computed Tomography (CT) scans of the spine (3DSpine in Section 7.1.4).

7.1.1 Hand Radiographs (2DHand)

On this dataset of 895 hand radiographs of the Digital Hand Atlas Database System [38, 168], we compare our proposed network architectures to the state-of-the-art. Furthermore, we use this dataset for evaluating several hyperparameters of our SCN.

Method	Input Size (in pix)	PE_{all} (in mm) Median	Mean ± SD	$\#O_r$ (in %) $r = 2$ mm	$r = 4$ mm	$r = 10$ mm
SCN from [108]:	512×512	**0.43**	**0.66 ± 0.74**	**1,659 (5.01%)**	**241 (0.73%)**	**3 (0.01%)**
U-Net from [108]:	512×512	0.44	0.70 ± 2.18	1,703 (5.14%)	270 (0.82%)	22 (0.07%)
SCN from [106]:	256×256	0.91	1.13 ± 0.98	4,109 (12.40%)	444 (1.34%)	12 (0.04%)
U-Net from [106]:	256×256	0.68	0.87 ± 1.05	2,064 (6.23%)	235 (0.71%)	15 (0.05%)
CoordReg-Net:	512×512	1.83	2.12 ± 2.12	14,612 (44.13%)	2,598 (7.85%)	77 (0.23%)
ConvOnly-Net:	128×128	1.13	1.29 ± 1.13	4,919 (14.85%)	343 (1.04%)	9 (0.03%)
Contracting-Net:	256×256	1.85	1.96 ± 1.14	14,554 (43.95%)	1,082 (3.27%)	12 (0.04%)
Urschler et al. [147]:	1250×1250	0.51	0.80 ± 0.93	2,586 (7.81%)	510 (1.54%)	18 (0.05%)
Štern et al. [130]:	1250×1250	0.51	0.80 ± 0.91	2,582 (7.80%)	512 (1.55%)	15 (0.05%)
Ebner et al. [31]:	1250×1250	0.51	0.97 ± 2.45	2,781 (8.40%)	716 (2.16%)	228 (0.69%)
Lindner et al. [85]:	1250×1250	0.64	0.85 ± 1.01	2,094 (6.32%)	347 (1.05%)	20 (0.06%)

Table 7.1: Localization results from a three-fold cross-validation on 895 images from the 2DHand dataset with 37 annotated landmarks.

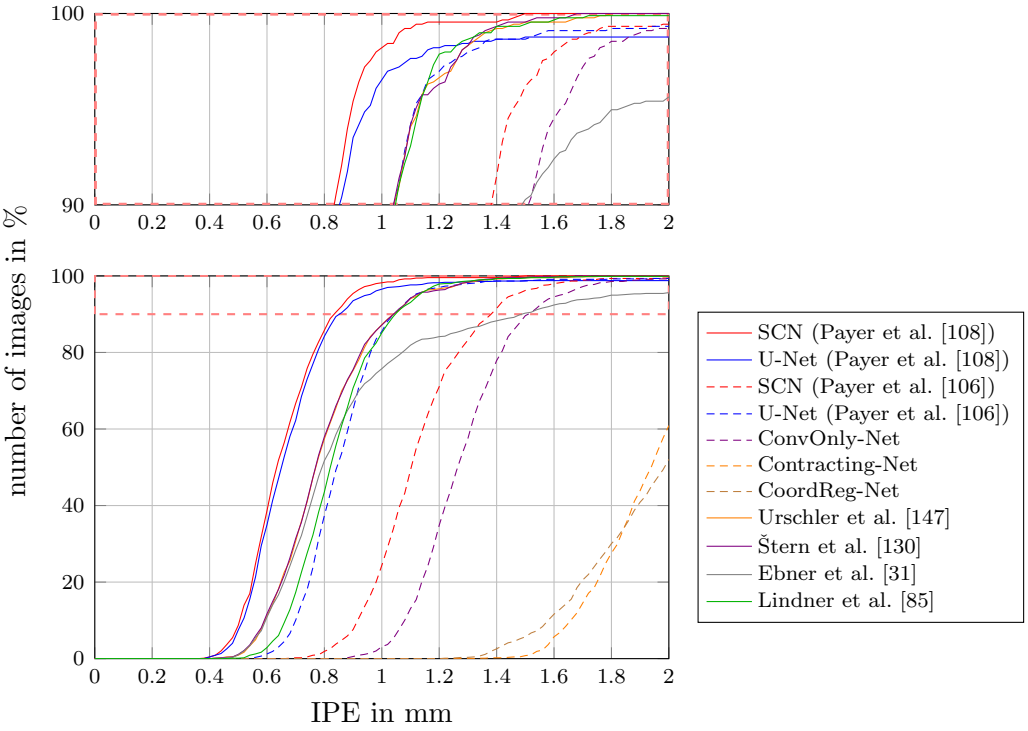


Figure 7.2: Cumulative distribution of the image-specific point-to-point error on 2DHand dataset. The zoomed-in region of the dashed red box is shown on top.

Figure 7.1 visualizes the offsets of the landmarks predicted by the SCN to all ground-truth landmarks for all images of the 2DHand dataset. This figure shows the high accuracy of the landmarks predicted by the SCN, especially for the landmarks on the fingers. Furthermore, all except three landmarks are within 10 mm of the annotated ground-truth, confirming the high robustness towards landmark misidentification of our proposed network architecture.

7.1.1.1 Comparison of Network Architectures

We compare all network architectures as described in the experimental setup in Section 6.4, as well as other state-of-the-art methods. In Table 7.1, we state the point-to-point error (PE_{all}) to assess the overall accuracy, as well as the number of outliers given different error radii ($\#O_r$) to assess robustness towards landmark misidentification. Local accuracy of the identified landmarks can be best seen from the cumulative distribution of the image specific point-to-point error (IPE), which is illustrated graphically in Fig. 7.2.

From Table 7.1 and Fig. 7.2 we see that the CoordReg-Net performs worst among all compared methods, which endorses the heatmap regression framework over the coordinate regression framework. We also observed that the CoordReg-Net has problems with convergence, often getting stuck in local minima by predicting a mean shape of all images of the dataset. Additionally, stacking multiple networks to a cascade can only slightly refine previous predictions to improve the accuracy, but in overall, the heatmap regression framework is to be preferred. While the Contracting-Net architecture has more robust results, leading to fewer outliers, it also suffers from a low accuracy in terms of PE and IPE due to the downsampled heatmap predictions. The ConvOnly-Net leads to more accurate results due to the increased resolution. However, it is still not as accurate as the U-Net and the SCN. While both our preliminary U-Net and SCN from [106] already result in predictions that are both accurate and robust towards landmark misidentification, our refined versions from [108] lead to the most accurate results, not only compared to our own networks, but also to other state-of-the-art methods using Random Forest (RF).

Due to the worse performance of the CoordReg-Net, the ConvOnly-Net, the Contracting-Net, as well as the U-Net and SCN from [106], we do not report results for these networks for subsequent experiments on other datasets. Instead, to remove unnecessary clutter we only report results for the U-Net and SCN from [108], since they outperform the other networks as well as state-of-the-art methods for anatomical landmark localization.

7.1.1.2 Improvements of the SpatialConfiguration-Net

To enable a fair comparison of the SCN from [108] to the SCN of [106] where we used only 256×256 pixels as input image size and to investigate the influence of this input size hyperparameter, we perform an experiment on the 2DHand dataset varying the input/heatmap target size between 128 and 512 pixels. The result of this experiment is shown as a cumulative IPE distribution in Fig. 7.3, demonstrating that we outperform our previous work even if the input/heatmap target size is reduced to 128×128 pixels. Thus, our observed improvement compared to [106] does not come from the increased input/heatmap target size, but instead from the following two extensions:

First, our modified *local appearance* component uses a wider receptive field due to the additional depth levels enabled by the residual architecture [52]. We additionally analyzed the effect of different numbers of convolution outputs for this component. However, we

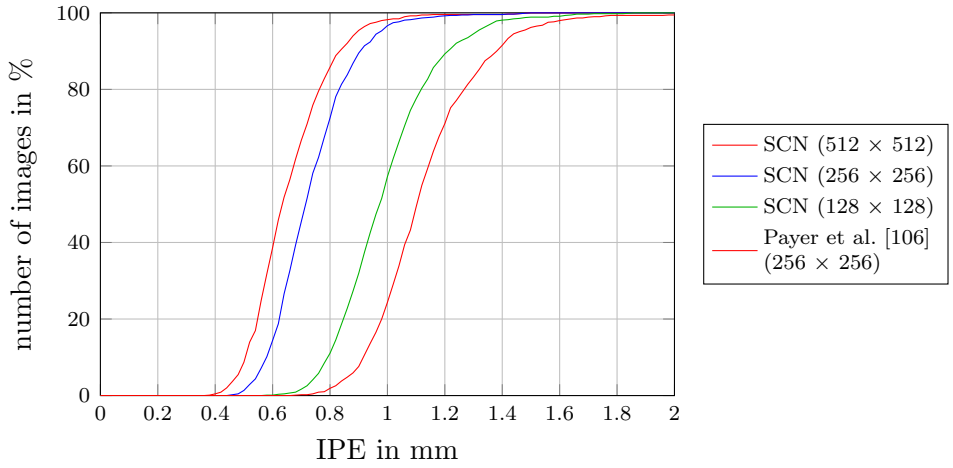


Figure 7.3: Cumulative distribution of IPE on 2DHand dataset for varying input/heatmap target sizes of our SCN architecture.

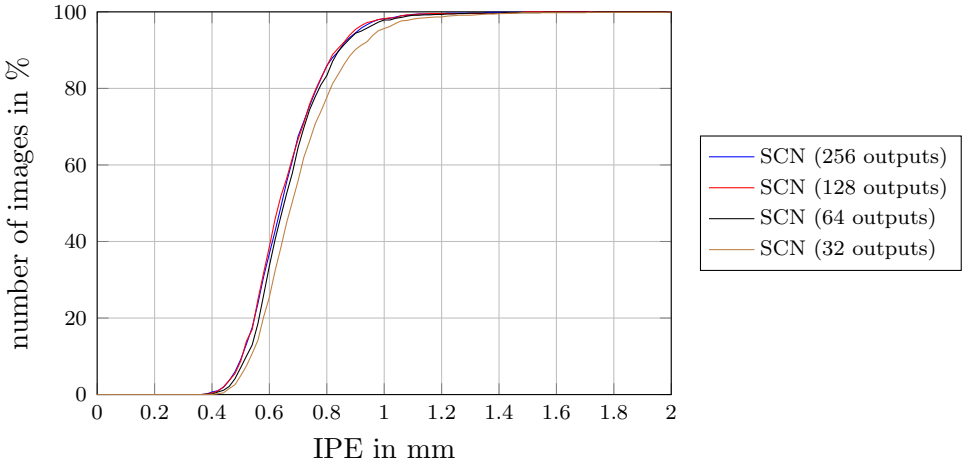


Figure 7.4: Cumulative distribution of IPE on 2DHand dataset for a varying number of intermediate filter outputs of our SCN architecture.

obtained only small differences when varying the number of outputs between 32 and 256, with the difference of 128 and 256 being almost indistinguishable, as shown as a cumulative IPE distribution in Fig. 7.4. Therefore, we use 128 convolution outputs for all our further experiments, since this choice allows faster training. Also, the modified *spatial configuration* component of the SCN does not only allow learning higher-order relationships among the positions of landmarks but also uses smaller convolution kernel sizes to obtain the same receptive field.

Second, improvements in localization performance also come from our objective function involving learnable heatmap peak widths $\boldsymbol{\sigma}$ for each landmark (see Section 4.3.1). To investigate the behavior of this extension, we perform experiments comparing different

fixed values for all σ_i, as proposed in [106], with the variant where $\boldsymbol{\sigma}$ is learned independently per landmark, and weighting factor α is varied. As learning the heatmap peak widths also enables deeper insights into the network predictions, we show results and in-depth analysis of this experiment in Section 7.2.

7.1.1.3 Comparison to Other Methods

When comparing our proposed SCN to other state-of-the-art algorithms, the results in Fig. 7.2 and Table 7.1 indicate that it greatly outperforms all our previous methods, i.e., methods that either solely rely on Random Regression Forests (RRFs) [31, 147], or that additionally incorporate a Markov Random Field (MRF) as an explicit graphical model [130], as well as our preliminary Convolutional Neural Networks (CNNs) of [106]. Moreover, our SCN also outperforms the state-of-the-art landmark localization method of [86], who applied their algorithm to our preprocessed 2DHand dataset with the same cross-validation split. Overall, our SCN achieves an unprecedented low number of three outliers at 10 mm for this dataset. Interestingly, in terms of $\#\mathrm{O}_r$ our optimized U-Net implementation performs nearly as good as the SCN on the 2DHand dataset (see Table 7.1). This presumably is a consequence of the comparatively large amount of training data that were available for such a deep architecture to be trained. However, from the cumulative IPE distribution in Fig. 7.2, we see that already at an error radius of 1.25 mm, U-Net shows a drop in performance compared to most of the other state-of-the-art methods. Thus, from results of $\#\mathrm{O}_r$ and IPE it can be concluded that using the U-Net leads to a few outliers in many images, while methods incorporating graphical models like [85] or [147] lead to fewer images with outliers, but more of them per image.

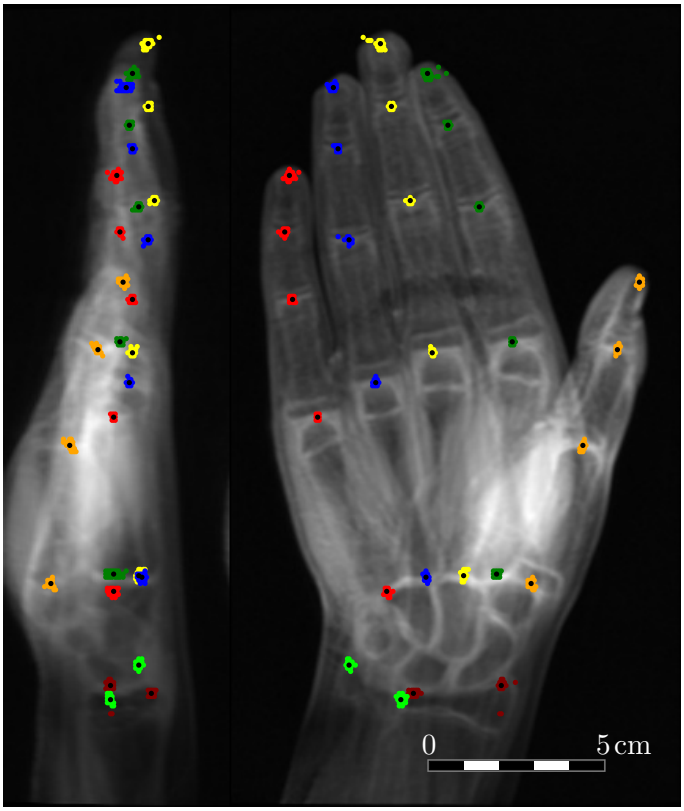


Figure 7.5: Example projection images with all from the SCN predicted landmarks of all images from the 3DHand dataset. Using an image and its ground-truth landmarks (black dots) as the base, all predicted landmarks of all images are shifted by the offsets to the corresponding ground-truth.

7.1.2 Hand MRIs (3DHand)

In our in-house dataset of 60 T1-weighted 3D gradient-echo hand Magnetic Resonance Imaging (MRI) scans, we show applicability of our network architectures for 3D.

The offsets of ground-truth and predicted landmarks from the SCN are visualized in Fig. 7.5. From this figure, we can see that the SCN's predictions are accurate, being close to the ground-truth, without a single landmark being more than 10 mm away from the ground-truth. This can be observed in more detail in Table 7.2, where we compare the CNN predictions to previous works of our group [31, 106, 147]. The results demonstrate that our SCN gives the best localization performance of all compared methods also for 3D MR images. We outperform also the preliminary SCN from [106] by a large margin in terms of landmark localization error (PE_{all}) using the same voxel resolution in both approaches. We are also considerably better in terms of localization error compared to the approaches based on RRF [31, 147], although these methods use a higher voxel resolution being twice as high in each dimension. Interestingly, besides our proposed method, the only other method that achieves a perfect outlier rate at $r = 10$ mm on this dataset also makes

Method	Input Size (in vox)	PE_{all} (in mm) Median	PE_{all} (in mm) Mean ± SD	$\#O_r$ (in %) $r = 2$ mm	$\#O_r$ (in %) $r = 4$ mm	$\#O_r$ (in %) $r = 10$ mm
SCN:	96×128×32	**0.90**	**0.84 ± 0.62**	**96 (4.03%)**	**5 (0.21%)**	**0 (0.00%)**
U-Net:	96×128×32	**0.90**	0.90 ± 1.16	123 (5.17%)	10 (0.42%)	2 (0.08%)
SCN from [106]:	96×128×32	1.01	1.20 ± 1.48	215 (9.03%)	15 (0.63%)	3 (0.13%)
Urschler et al. [147]:	172×300×72	1.10	1.31 ± 0.72	293 (12.15%)	23 (0.97%)	**0 (0.00%)**
Ebner et al. [31]:	172×300×72	1.27	1.44 ± 1.51	416 (17.48%)	29 (1.22%)	6 (0.25%)

Table 7.2: Cross-validation localization results on 60 images from 3DHand with 28 annotated landmarks.

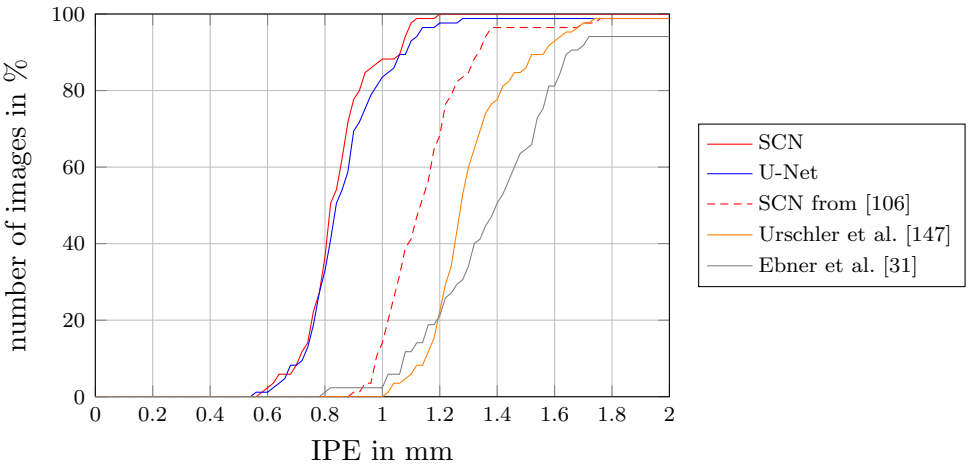
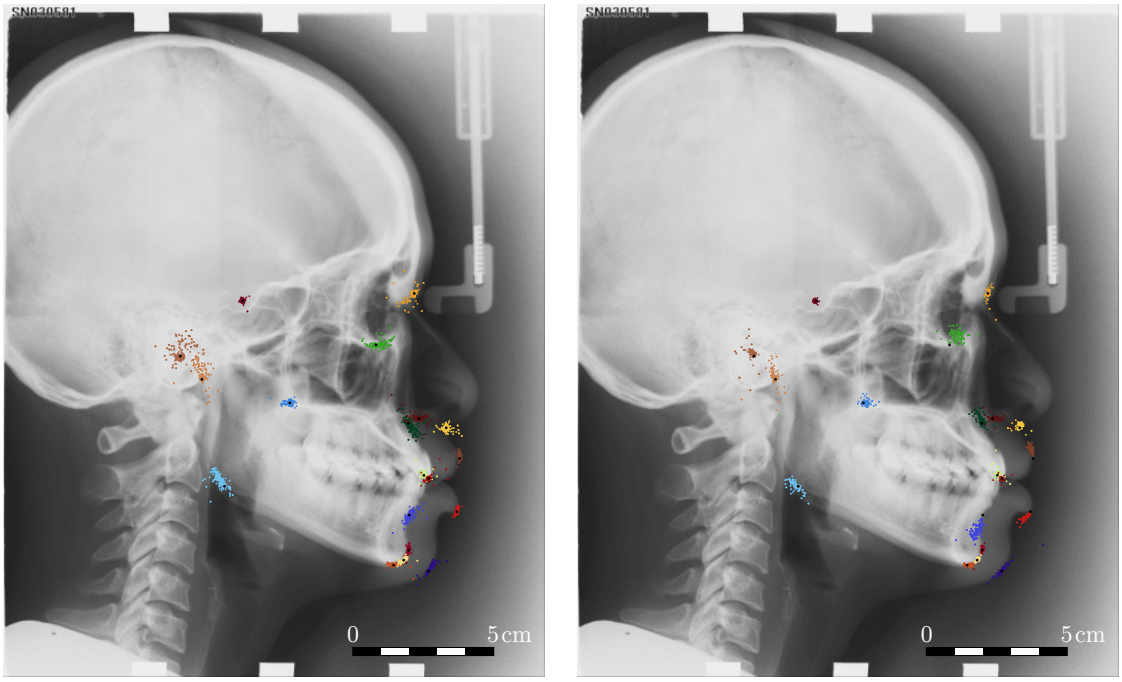


Figure 7.6: Cumulative distribution of IPE on the 3DHand dataset.

extensive use of information on the *spatial configuration* of landmarks [147]. Additionally to our 3D SCN that shows the best localization performance, also the U-Net outperforms all our previous methods on this small dataset comprised of only 60 volumes. We presume that this good performance comes from our extensive 3D training data augmentation scheme involving both intensity and spatial transformations, which is highly beneficial for training such deep CNNs.



(a) *Set1* **(b)** *Set2*

Figure 7.7: Example images with all from the SCN predicted landmarks of all images from the 2DSkull dataset. Using an image and its ground-truth landmarks (black dots) as the base, all predicted landmarks of all images are shifted by the offsets to the corresponding ground-truth.

7.1.3 Lateral Cephalograms (2DSkull)

We compare our proposed SCN in the context of a landmark localization challenge on this dataset of 400 images of the ISBI 2015 Cephalometric X-ray Image Analysis Challenge [152]. As the results presented in [152] lack cumulative distribution graphs, we only compare the reported values of the best-performing methods from [62] and [85] on both test sets *Set1* and *Set2* with 150 and 100 images, respectively.

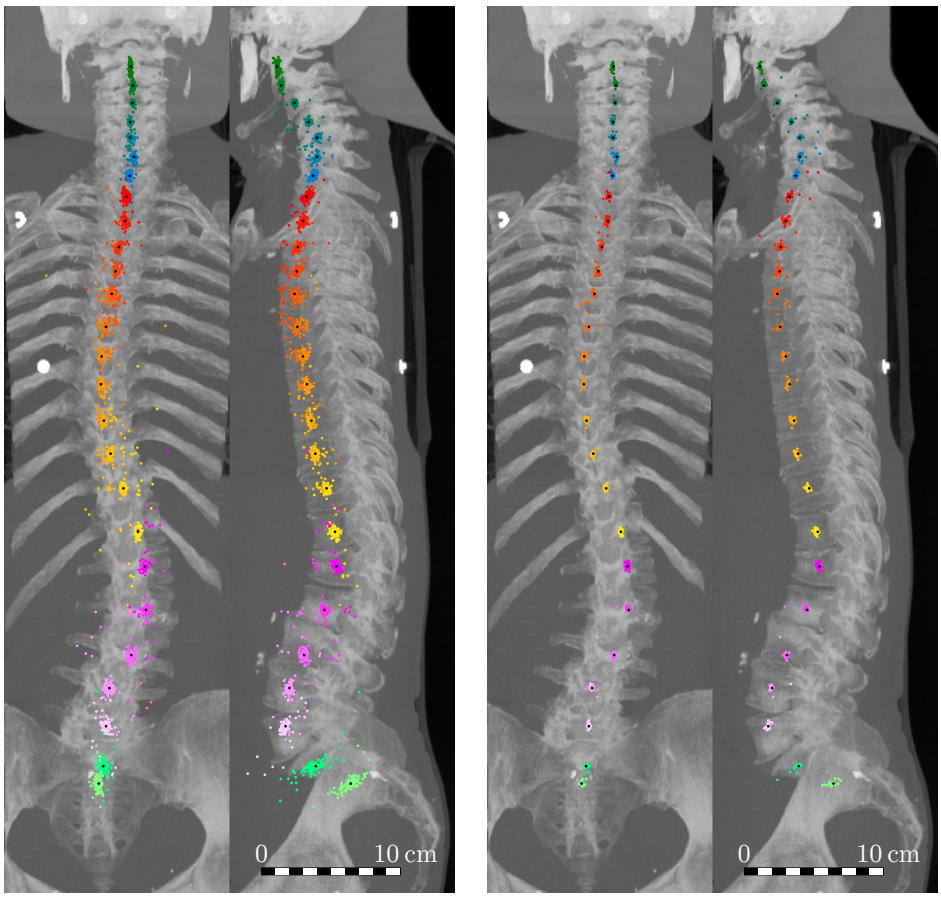
The prediction offsets of all landmarks of all images for both test sets are shown in Fig. 7.7. We found that many problems of our method on this particularly challenging dataset are at anatomically ill-defined landmarks, e.g., the landmarks on the lip ($\mathcal{L}_{13}$ and $\mathcal{L}_{14}$), and landmarks on the jaw bone ($\mathcal{L}_{10}$ and $\mathcal{L}_{19}$). As shown in Table 7.3, our proposed SCN outperforms previous winners of this challenge, especially for test set *Set1*. With [85] resembling the current state-of-the-art on this dataset, our results demonstrate that our method is more accurate locally ($\mathrm{PE}_{\mathrm{all}}$ and $\#\mathrm{O}_r$ with $r = 2$ mm), while robustness towards landmark misidentification is the same ($\#\mathrm{O}_r$ with $r = 4$ mm). Although the results for *Set2* are in line with the state-of-the-art, the presented metrics are much worse as compared to *Set1*, which we attribute the following observation: In *Set2* there is a systematic shift for some landmarks (e.g., $\mathcal{L}_3$, $\mathcal{L}_6$, $\mathcal{L}_{13}$, $\mathcal{L}_{14}$), as can be seen in the offset

	Method	PE_{all} (in mm)		$\#O_r$ (in %)			
		Median	Mean ± SD	$r =$ 2 mm	$r =$ 2.5 mm	$r =$ 3 mm	$r =$ 4 mm
Set1	SCN:	**0.89**	**1.49 ± 1.67**	**22.49%**	**17.16%**	**13.51%**	**8.46%**
	U-Net:	0.95	1.79 ± 3.86	24.46%	19.02%	15.30%	10.00%
	Urschler et al. [147]:	1.08	1.74 ± 1.82	26.46%	20.32%	15.61%	10.07%
	Lindner et al. [85]:	n/a	1.67 ± n/a	26.32%	19.79%	14.81%	8.53%
	Ibragimov et al. [62]:	n/a	1.84 ± n/a	28.28%	22.60%	18.07%	11.96%
Set2	SCN:	**1.19**	**1.91 ± 1.94**	**32.95%**	27.37%	21.63%	12.95%
	U-Net:	1.22	2.04 ± 3.33	**32.95%**	**26.89%**	21.95%	13.00%
	Urschler et al. [147]:	1.33	1.99 ± 1.87	34.79%	27.16%	**21.37%**	**12.37%**
	Lindner et al. [85]:	n/a	1.92 ± n/a	33.89%	28.00%	22.37%	12.58%
	Ibragimov et al. [62]:	n/a	2.14 ± n/a	37.26%	29.53%	23.47%	14.89%

Table 7.3: Localization results on the test images from 2DSkull dataset with 19 landmarks. We report the results of the first (*Set1* with 150 images) and second (*Set2* with 100 images) test dataset of the ISBI 2015 Grand Challenge according to the evaluation protocol in [152].

image Fig. 7.7b. By visually inspecting the ground-truth, we confirmed this systematic shift in the annotation between the training set and the testing set *Set2*, indicating that the train and test annotations were created either by different annotators or by the same annotator but during different periods. Looking at the predictions of the SCN in more detail, for the landmarks $\mathcal{L}_3$, $\mathcal{L}_6$, and $\mathcal{L}_{13}$ almost all predicted landmarks have a larger distance than 2 mm to the ground-truth ($\mathcal{L}_3$: 96%, $\mathcal{L}_6$: 96%, and $\mathcal{L}_{13}$: 91%). Due to this systematic shift in *Set2*, the predictions of more precise methods have a smaller probability of being closer to the misannotated landmarks, distorting the presented overall $\#O_r$ metrics.

This observation shows that the 2DSkull dataset is much more tedious to annotate and further motivates the use of fully automatic methods for anatomical landmark localization, as they are more objective and not influenced by experience and state of the mind of the annotator.



(a) *Set1* **(b)** *Set2*-corrected

Figure 7.8: Example projection images with all from the SCN predicted landmarks of all images from the 3DSpine dataset. Using an image and its ground-truth landmarks (black dots) as the base, all predicted landmarks of all images are shifted by the offsets to the corresponding ground-truth. The image on the left shows the predictions for *Set1*, the image on the right for *Set2*-corrected.

7.1.4 Spine CTs (3DSpine)

To compare our proposed SCN with other recent methods, we also evaluate on a publicly available volumetric dataset of pathological spine CT scans [44] used for the MICCAI CSI 2014 Vertebrae Localization and Identification Challenge. This dataset includes various challenging cases such as scoliotic spines, vertebrae fractures, metal insertions causing severe image artifacts, and highly restrictive field of views. Thus, due to a significant variation in local appearance, as well as repetitive structures in the spine CT scans, the spatial configuration of landmarks has to be learned to distinguish different vertebrae. Input and heatmap target size of the volumes are $96 \times 96 \times 192$ voxels; we resample the input images to have an isotropic spacing of 2 mm per dimension. With these input sizes, the network can process volumes with a physical extent up to $192 \times 192 \times 384$ mm^3. As

Method	*Set1*		*Set2*-mislabeled	
	PE_{all} (mm)	ID_{rate}	PE_{all} (mm)	ID_{rate}
SCN:	**6.2 ± 9.9**	**86.1%**	6.0 ± 16.1	90.9%
U-Net:	7.3 ± 12.9	82.7%	9.1 ± 22.1	83.0%
Liao et al. [83]:	–	–	6.5 ± 8.6	88.3%
Sekuboyina et al. [123]	–	–	7.4 ± 9.3	86.1%
Yang et al. [160]:	9.1 ± 7.2	80.0%	8.6 ± 7.8	85.0%
Chen et al. [12]:	–	–	8.8 ± 13.0	84.2%
Glocker et al. [44]:	12.4±11.2	70.0%	13.2±17.8	74.0%

Method	*Set2*-corrected	
	PE_{all} (mm)	ID_{rate}
SCN:	**2.9 ± 4.4**	**96.0%**
U-Net:	4.9 ± 7.7	87.5%

Table 7.4: Localization results on the 3DSpine dataset with at most 26 landmarks per volume. We report results for the two-fold cross-validation with 224 images (*Set1*) and the MICCAI CSI 2014 Challenge test set with 60 images (*Set2*). For *Set2*, we report results on the unmodified test set with mislabeled vertebrae in the ground-truth (*Set2*-mislabeled), and on the test set, where we corrected these mislabeled vertebrae (*Set2*-corrected).[5]

some volumes have a larger extent in the z-axis (i.e., the axis perpendicular to the axial plane) that would not fit into the network, we process such volumes as follows: During training, we crop a subvolume at a random position at the z-axis. During testing, we split the volumes at the z-axis into multiple subvolumes that overlap for 96 pixels, and process them one after another. Then, we merge the network predictions of the overlapping subvolumes by taking the maximum response over all predictions.

The prediction offsets of all landmarks of all images for both sets are visualized in Fig. 7.8, which shows that the predictions of the SCN are precise, having only a small number of misidentified landmarks. Due to the large variation in the *local appearance* of this challenging dataset, the network profits a lot from more diverse training data, which can be observed in the better results for *Set2*, being trained with twice the number of training images. These results are also shown in Table 7.4, where we compare our network predictions with the latest reported results on this dataset. Our approach outperforms all state-of-the-art methods evaluated on this dataset, on both cross-validation set (*Set1*) and challenge test set (*Set2*[5]). Although the U-Net shows overall good results on this dataset, neither landmark localization error (PE_{all}) in general, nor robustness towards landmark misidentification (ID_{rate}) specifically are as good as for our SCN. Due to the lack of explicitly modeling the spatial configuration of landmarks, the U-Net shows a significant decrease of 8.5% in ID_{rate} for *Set2*. As compared to random forests [44] and other CNNs [12, 83, 160], our SCN performing heatmap regression shows the best localization

[5]When evaluating our proposed algorithm on the testing dataset, we identified three volumes where all ground-truth vertebrae labels were shifted by up to five vertebrae, resulting in wrong landmark positions with more than 100 mm distance to the actual correct ground-truth position. For comparison with previously reported results, we show results on the original dataset (*Set2*-mislabeled). In agreement with the challenge organizers, we also report results on the corrected ground-truth (*Set2*-corrected), while the organizers have updated the challenge dataset accordingly.

performance. In the recent CNN-based work of [123], the authors propose to project the 3D information of spine anatomy into 2D sagittal and coronal views, and solely use these views as input for their CNN. Although their method is tailored towards spine datasets, where the landmarks are situated along the axis orthogonal to the axial plane, our generic SCN that directly processes 3D information outperforms their method. For this experiment in overall, we show that our SCN outperforms all state-of-the-art methods, without the need for sophisticated training procedures (e.g., [83]) or pre-/postprocessing (e.g., [123, 160]).

SCN with	PE_{all} (in mm)		$\#O_r$ (in %)		
	Median	Mean ± SD	$r = 2$ mm	$r = 4$ mm	$r = 10$ mm
$\alpha = 5$:	0.45	0.67 ± 0.71	**1594 (4.81%)**	**220 (0.66%)**	6 (0.02%)
$\alpha = 10$:	0.44	**0.66 ± 0.73**	1638 (4.95%)	224 (0.68%)	**3 (0.01%)**
$\alpha = 20$:	**0.43**	**0.66** ± 0.74	1659 (5.01%)	241 (0.73%)	**3 (0.01%)**
$\alpha = 40$:	**0.43**	**0.66** ± 0.75	1670 (5.04%)	267 (0.81%)	7 (0.02%)
$\sigma = 1.0$:	0.44	0.69 ± 0.81	2033 (6.14%)	348 (1.05%)	6 (0.02%)
$\sigma = 1.5$:	0.45	0.69 ± 0.79	1924 (5.81%)	308 (0.93%)	7 (0.02%)
$\sigma = 2.0$:	0.44	0.66 ± 0.75	1634 (4.93%)	268 (0.81%)	6 (0.02%)

Table 7.5: Localization results on the 2DHand dataset of the SCN for different values of fixed σ and weighting factors α to learn individual σ values.

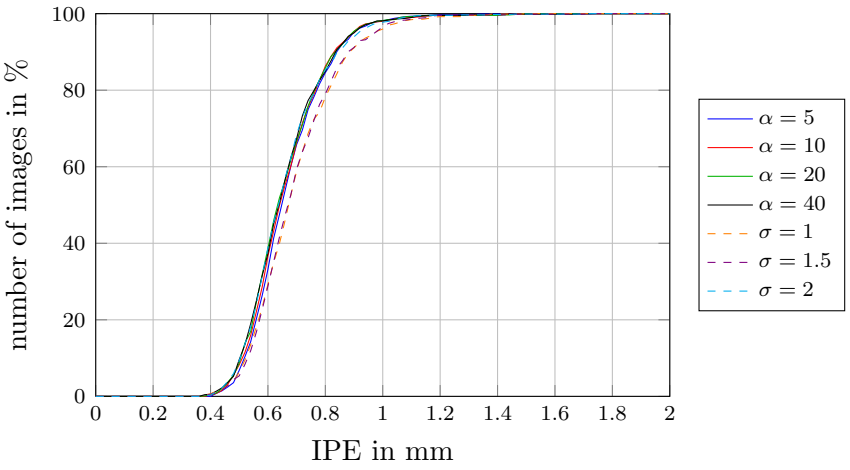


Figure 7.9: Cumulative distribution of IPE on the 2DHand dataset for the SCN with different fixed σ values as well as different weighting factors α to learn individual $\boldsymbol{\sigma}$ values.

7.2 Learning the Size $\boldsymbol{\sigma}$ of the Gaussian Heatmaps

In this section, we analyze the influence of letting the network learn the Gaussian heatmap sigma values of each landmark individually, as we proposed with our novel loss function in Eq. (4.8). We mainly use the 2DHand dataset for this analysis, as it contains both accurately defined landmarks on repetitive structures, e.g., on the phalanges, as well as less accurately defined landmarks on bone borders, e.g., on the carpals (see Fig. 6.1a). In Table 7.5 and Fig. 7.9 we present results on the 2DHand dataset for the SCN with various values of fixed σ and various α weighting factors of Eq. (4.8). These results show that varying $\boldsymbol{\sigma}$ independently outperforms the variants with the same fixed σ_i for all landmarks $\mathcal{L}_i$, while the difference in localization performance for varied values of α is small, indicating that this is an uncritical parameter. With independently learned $\boldsymbol{\sigma}$ values, the heatmap peak widths adapt to the data by encoding the uncertainty of network predictions, see Fig. 4.4. Landmarks that are easier to predict unambiguously have a smaller σ_i than those that show more variation or ambiguity in the training dataset.

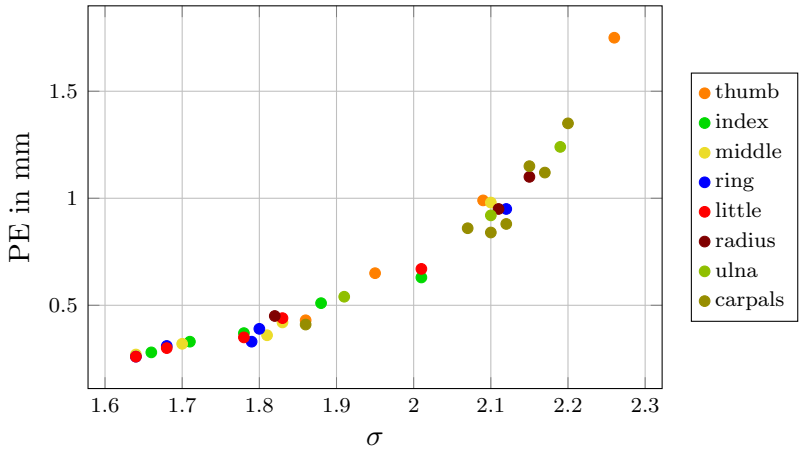


Figure 7.10: Relation of learned σ_i values to the PE_i for each landmark $\mathcal{L}_i$.

Thus, in contrast to fixed σ_i values, we do not need to make a predetermined tradeoff between large σ_i generating oversmoothed, potentially inaccurate predictions, and small σ_i leading to potentially highly accurate responses but with multiple peaks nearby.

To evaluate whether the landmarks' $\boldsymbol{\sigma}$ encodes the uncertainty of network predictions, we analyze the relationship between the predicted landmark's σ_i and its PE_i. The individual predictions in Fig. 7.1, as well as the results in Fig. 7.10 and Table 7.6, show that the landmarks on the fingers are the most precise ones having the smallest σ and PE values, confirming that the network learns small σ values when it is certain to make a small error. As corresponding landmarks on different fingers locally look the same, corresponding landmarks on different fingers (with exception of the thumb) lead to almost the same σ and PE. For the landmarks of the thumb, the network is much less certain, producing higher σ and PE. Specifically, we observed for landmark $\mathcal{L}_2$ and $\mathcal{L}_3$ inconsistent annotations in the ground-truth. Instead of the base of the distal phalanx ($\mathcal{L}_2$) and the base of the proximal phalanx ($\mathcal{L}_3$), some annotations are wrongly located on the head of the proximal phalanx or the head of the metacarpal, respectively. In contrast to other fingers, where the heads and bases of connected bones are close, in the thumb, a misannotation leads to a larger localization error. Thus, as landmarks with inconsistent annotations typically lead to larger σ_i, our proposed loss function could also potentially be used to detect these kinds of misannotations in a dataset. Finally, the remaining landmarks on the radius, ulna, and the metacarpals are more difficult to annotate as they are less precisely defined, usually being located on smooth edges and not on sharp corners (see Fig. 6.1a). Hence, with the exception of landmark $\mathcal{L}_{25}$, $\mathcal{L}_{28}$, and $\mathcal{L}_{34}$, which lie on sharp corners, the σ and PE values are high.

We evaluate whether the final σ_i values are dependent on their initialization at the example of the 2DHand dataset. We visualize the progression of the individual σ_i throughout the network training optimization for the phalanges and metacarpals (Fig. 7.11), and for radius, ulna, and the carpals (Fig. 7.12). These graphs show that already in the beginning

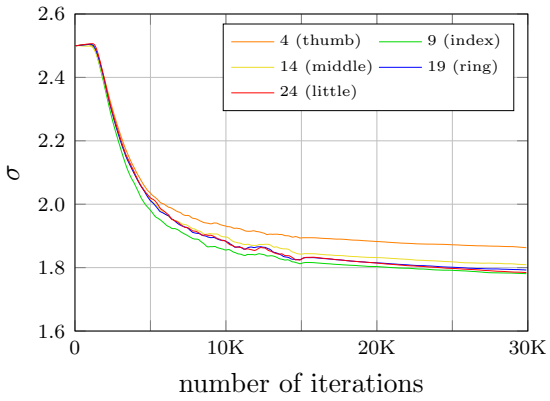
σ mean PE	thumb $\mathcal{L}_1$–$\mathcal{L}_4$	index finger $\mathcal{L}_5$–$\mathcal{L}_9$	middle finger $\mathcal{L}_{10}$–$\mathcal{L}_{14}$	ring finger $\mathcal{L}_{15}$–$\mathcal{L}_{19}$	little finger $\mathcal{L}_{20}$–$\mathcal{L}_{24}$
bases of metacarpals	2.09 0.99 ± 0.74	2.01 0.63 ± 0.50	2.10 0.98 ± 0.67	2.12 0.95 ± 0.63	2.01 0.67 ± 0.47
bases of proximal phalanges	2.26 1.75 ± 1.19	1.88 0.51 ± 0.32	1.83 0.42 ± 0.25	1.80 0.39 ± 0.23	1.83 0.44 ± 0.27
bases of middle phalanges	-	1.71 0.33 ± 0.18	1.70 0.32 ± 0.19	1.68 0.31 ± 0.18	1.68 0.30 ± 0.19
bases of distal phalanges	1.95 0.65 ± 0.48	1.66 0.28 ± 0.17	1.64 0.27 ± 0.16	1.64 0.26 ± 0.16	1.64 0.26 ± 0.16
heads of distal phalanges	1.86 0.43 ± 0.30	1.78 0.37 ± 0.89	1.81 0.36 ± 0.36	1.79 0.33 ± 0.20	1.78 0.35 ± 0.24

	radius			ulna		
	$\mathcal{L}_{25}$	$\mathcal{L}_{26}$	$\mathcal{L}_{27}$	$\mathcal{L}_{28}$	$\mathcal{L}_{29}$	$\mathcal{L}_{30}$
σ	1.82	2.15	2.11	1.91	2.10	2.19
mean PE	0.45 ± 0.71	1.10 ± 1.04	0.95 ± 0.76	0.54 ± 0.54	0.92 ± 1.06	1.24 ± 1.02

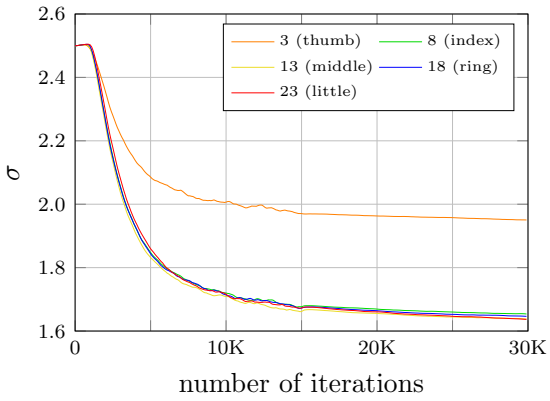
	carpals						
	$\mathcal{L}_{31}$	$\mathcal{L}_{32}$	$\mathcal{L}_{33}$	$\mathcal{L}_{34}$	$\mathcal{L}_{35}$	$\mathcal{L}_{36}$	$\mathcal{L}_{37}$
σ	2.07	2.17	2.10	1.86	2.12	2.20	2.15
mean PE	0.86 ± 1.04	1.12 ± 0.85	0.84 ± 0.72	0.41 ± 0.25	0.88 ± 0.61	1.35 ± 1.28	1.15 ± 1.03

Table 7.6: Individual final σ and mean PE values for the individual landmarks of the 2DHand dataset. The first table shows the values for the phalanges and metacarpals, the second for radius and ulna, and the third for the carpals.

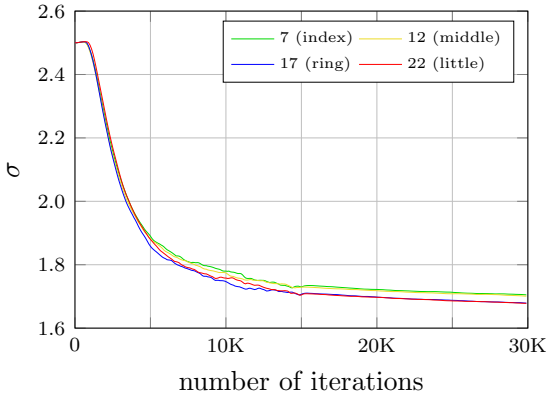
of training, the σ_i of certain and uncertain landmarks start to differ, while individually converging rapidly towards their final value, with only small changes after the first third of training. We also examined whether different initialization values of σ_i lead to different final sigma values. For initial $\sigma_i = \{1.0, 2.5, 5.0\}$, the maximum difference of final σ_i in fully trained networks is 0.01, showing that the initial σ value is not important. Finally, when comparing the resulting σ_i values for the three cross-validation sets, the σ_i values of the same landmarks do not differ by more than 0.02, which confirms that the network initialization is not important for the final predicted σ_i.



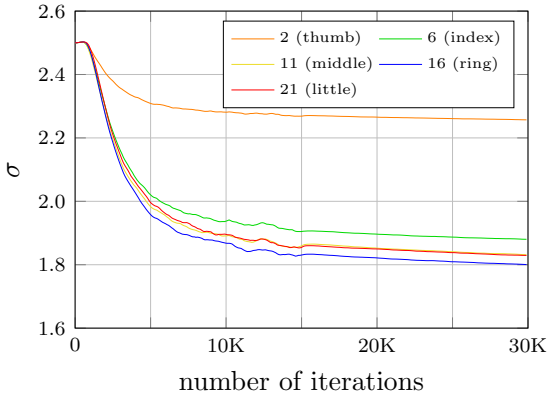
(a) Heads of distal phalanges.



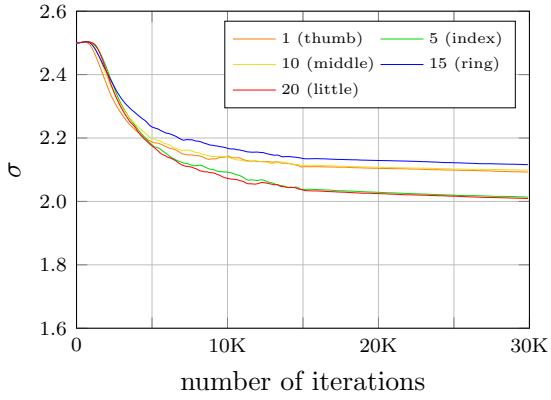
(b) Bases of distal phalanges.



(c) Bases of middle phalanges.

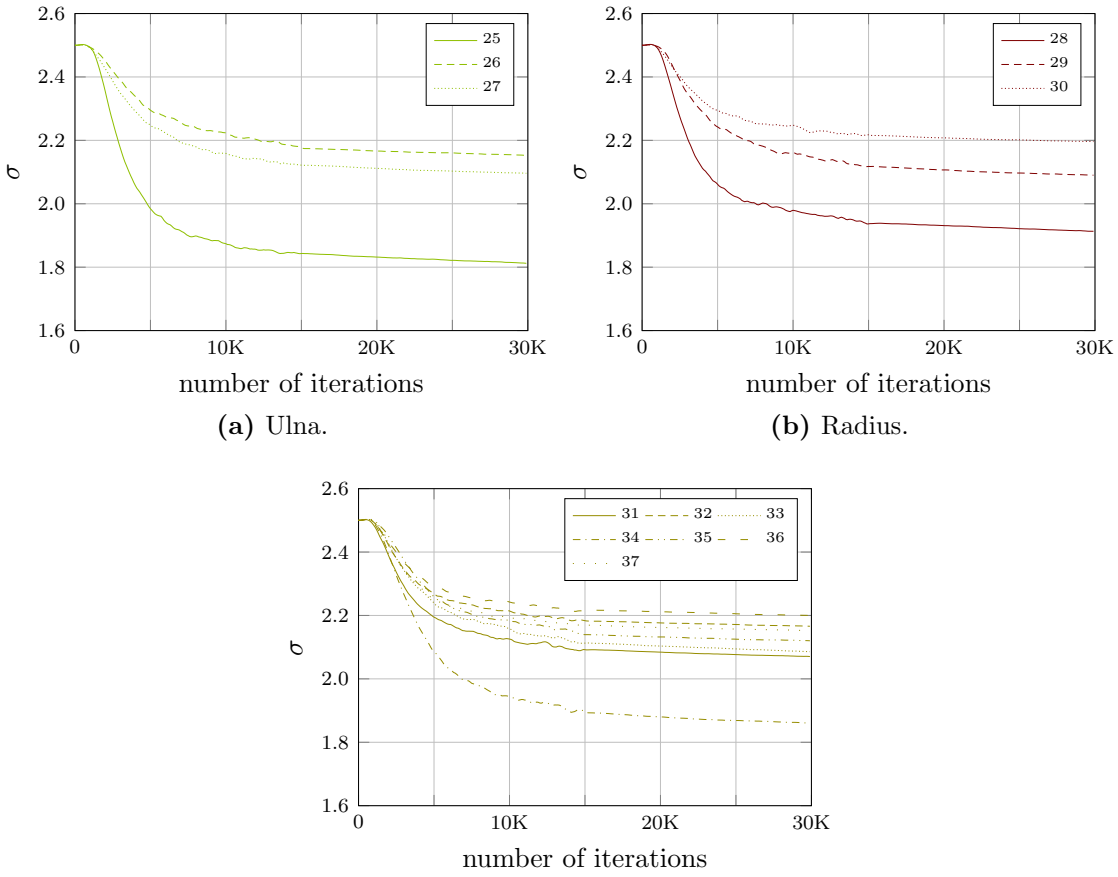


(d) Bases of proximal phalanges.



(e) Bases of metacarpals.

Figure 7.11: Progression of learned σ_i values during training for landmarks on the phalanges and metacarpals.



(a) Ulna.

(b) Radius.

(c) Carpals.

Figure 7.12: Progression of learned σ_i values during training for landmarks on ulna, radius, and carpals.

Network	PE_{all} (in mm)		$\#O_r$ (in %)		
	Median	Mean ± SD	$r = 2$ mm	$r = 4$ mm	$r = 10$ mm
SCN ($\mathbb{H}$):	**0.43**	0.66 ± 0.74	**1659 (5.01%)**	241 (0.73%)	3 (0.01%)
SCN ($\mathbb{H}^{LA}$):	0.84	21.76 ± 39.96	11918 (35.99%)	11033 (33.32%)	10925 (32.99%)
SCN ($\mathbb{H}^{SC}$):	1.91	2.14 ± 1.30	15533 (46.91%)	2856 (8.62%)	7 (0.02%)
LA-Net:	0.54	1.22 ± 6.00	2371 (7.16%)	552 (1.67%)	291 (0.88%)
LAaroundGT:	**0.43**	**0.62 ± 0.57**	1735 (5.24%)	**3 (0.01%)**	**0 (0.00%)**

Table 7.7: Localization results on the 2DHand dataset of the individual heatmap outputs.

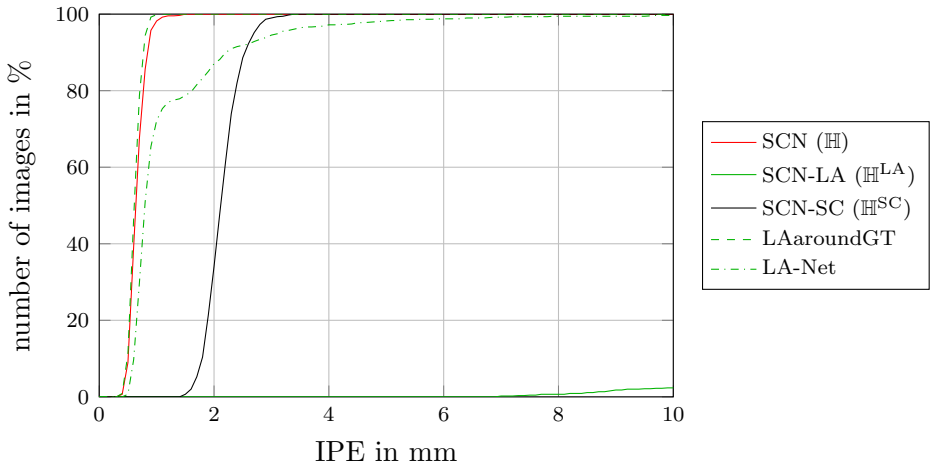


Figure 7.13: Cumulative distribution of IPE on 2DHand dataset for local appearance (LA) and spatial configuration (SC) components of our SCN (see Fig. 5.4). LAaroundGT shows LA predictions restricted to a radius within 10 mm of the artificially provided ground-truth landmark locations during testing.

7.3 Interacting Local Appearance ⇔ Spatial Configuration

In this section, we show how our SCN splits the localization problem into two simpler tasks by multiplying the predictions from the *local appearance* and *spatial configuration* components, which leads to a smaller required amount of annotated training data. We use the 2DHand dataset to show whether the SCN is able to perform this simplification in an experiment, where we separately extract the maxima of the heatmaps of both components $\mathbb{H}^{LA}$ and $\mathbb{H}^{SC}$ of the normally trained SCN. The results of this experiment are given in Table 7.7 and Fig. 7.13 as cumulative distributions of IPE for the *local appearance* heatmaps ($\mathbb{H}^{LA}$) and the *spatial configuration* heatmaps ($\mathbb{H}^{SC}$), respectively.

7.3.1 Individual Components of the SCN

To show that the increased performance of the SCN as compared to the U-Net and our preliminary SCN from [106] is not solely caused by the improved *local appearance* component, we performed an experiment, where a network consisting only of the *local appearance*

component (LA-Net) was trained. Due to the reduced receptive field of the LA-Net and the missing spatial configuration component, it performs worse not only compared to the SCN (see Fig. 7.13), but also to the U-Net (see Fig. 7.2).

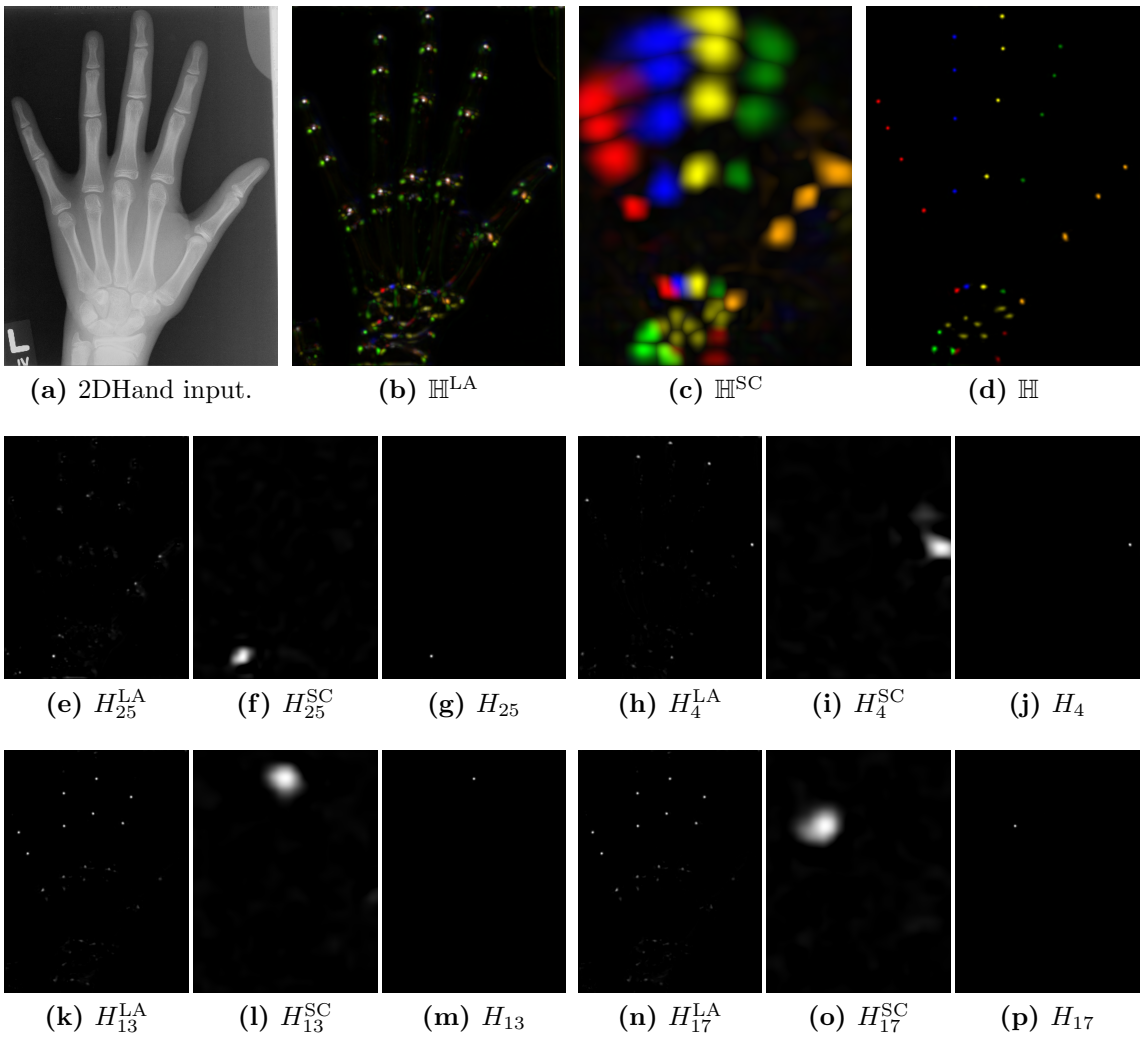
Furthermore, we evaluate the accuracy of the *local appearance* component of the SCN, as well as whether the *spatial configuration* component is able to remove infeasible locations. When taking the maximum responses from the *local appearance* heatmaps $\mathbb{H}^{\text{LA}}$ only (SCN-LA), the results are unsatisfactory. However, when removing responses on $\mathbb{H}^{\text{LA}}$ that are more than 10 mm away from the ground-truth locations, as has been done with LAaroundGT, the predictions are accurate. Therefore, due to ambiguous structures, the responses on LAaroundGT are neither the only ones nor the strongest ones of the *local appearance* heatmaps $\mathbb{H}^{\text{LA}}$, while a strong graphical model can be used to remove these ambiguities. When taking the maximum responses from the *spatial configuration* heatmaps $\mathbb{H}^{\text{SC}}$ only (SCN-SC), although the results are locally not accurate, they are in the surroundings of the target locations. Despite never seeing image intensity information directly, the *spatial configuration* component of the SCN is able to estimate the positions of all landmarks, effectively representing a graphical model. When combining the locally accurate but ambiguous candidate predictions from the *local appearance* heatmap $\mathbb{H}^{\text{LA}}$ with the robust but inaccurate prediction of the *spatial configuration* component $\mathbb{H}^{\text{SC}}$, the ambiguities from the *local appearance* component are eliminated, while the local accuracy of our proposed SCN remains the same as for LAaroundGT.

7.3.2 Representative Heatmap Outputs

To further clarify the interaction of the *local appearance* and *spatial configuration* stage of the SCN, we show heatmap outputs of representative examples for the evaluated datasets in Figs. 7.14 to 7.17. These figures show the input image, all predicted heatmaps ($\mathbb{H}^{\text{LA}}$, $\mathbb{H}^{\text{SC}}$, and $\mathbb{H}$) visualized as projected RGB images, as well as individual heatmaps (H_i^{LA}, H_i^{SC}, and H_i) for representative landmarks.

2DHand: The image in Fig. 7.14a shows a typical input image of this dataset. The images Figs. 7.14b to 7.14d show the predicted heatmap images colored in the respective landmark color as used in Fig. 6.1a and combined by taking the maximum RGB value over all 37 heatmaps. As many landmarks locally look very similar, e.g., the landmarks on the phalanges, this maximum RGB projection over the colored heatmaps for such landmarks leads to a white color in the $\mathbb{H}^{\text{LA}}$ heatmaps (see Fig. 7.14b). However in Fig. 7.14c for the *spatial configuration* heatmaps $\mathbb{H}^{\text{SC}}$, this maximum RGB projection leads to heatmaps, where each landmark can be distinguished easily. While the *spatial configuration* heatmaps $\mathbb{H}^{\text{SC}}$ only show an approximate location of the landmarks, the RGB projection of the final combined heatmaps $\mathbb{H}^{\text{LA}}$ in Fig. 7.14d leads to precise and distinct heatmaps for all landmarks.

When looking at the *local appearance* heatmap H_{25}^{LA} for $\mathcal{L}_{25}$ on the radius, although having the maximum response on the correct location, there are many other local maxi-



(a) 2DHand input. **(b)** $\mathbb{H}^{\mathrm{LA}}$ **(c)** $\mathbb{H}^{\mathrm{SC}}$ **(d)** $\mathbb{H}$

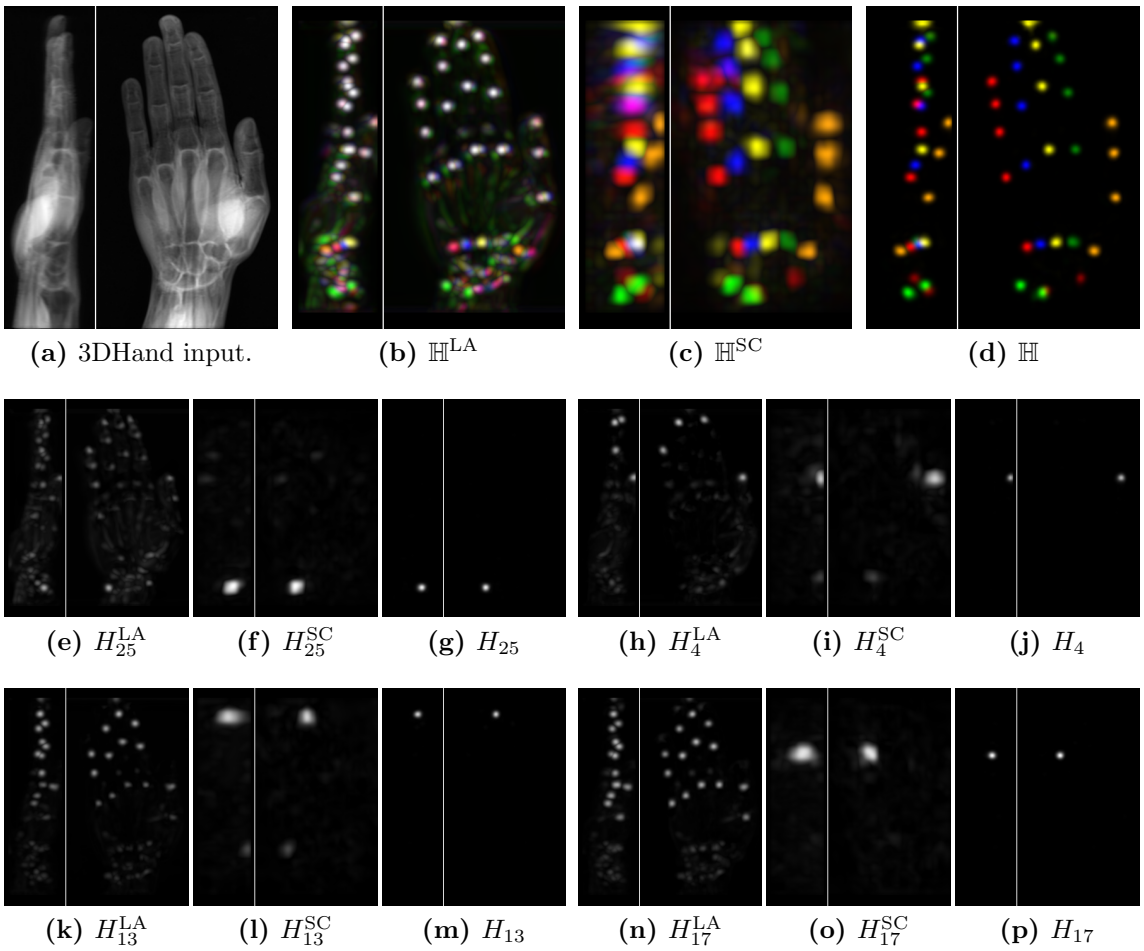
(e) H_{25}^{LA} **(f)** H_{25}^{SC} **(g)** H_{25} **(h)** H_{4}^{LA} **(i)** H_{4}^{SC} **(j)** H_{4}

(k) H_{13}^{LA} **(l)** H_{13}^{SC} **(m)** H_{13} **(n)** H_{17}^{LA} **(o)** H_{17}^{SC} **(p)** H_{17}

Figure 7.14: Example input and result images from the 2DHand dataset. The first row shows the input image, the combined local appearance ($\mathbb{H}^{\mathrm{LA}}$), spatial configuration ($\mathbb{H}^{\mathrm{SC}}$), and final ($\mathbb{H}$) heatmap outputs for all landmarks. The second and the third row show local appearance (H^{LA}), spatial configuration (H^{SC}), and final (H) heatmap outputs for landmarks 25, 4, 13, and 17.

mum responses. These responses are on structures that look locally similar, which indicates that the *local appearance* stage of the SCN is concentrating only on local regions. In contrast to the *local appearance* heatmap, the *spatial configuration* heatmap H_{25}^{SC} shows only a single response at the location of the landmark, although not being accurate. Hence, when combining H_{25}^{LA} and H_{25}^{SC} via multiplication, the final heatmap H_{25} shows only a single local maximum at the accurate position of the landmark.

For landmarks that are locally not distinguishable (e.g., landmarks on the metacarpals and phalanges for different fingers), the *local appearance* heatmaps have strong local maximum responses on all similarly looking landmarks (see Figs. 7.14h, 7.14k and 7.14n). For



(a) 3DHand input. **(b)** $\mathbb{H}^{\mathrm{LA}}$ **(c)** $\mathbb{H}^{\mathrm{SC}}$ **(d)** $\mathbb{H}$

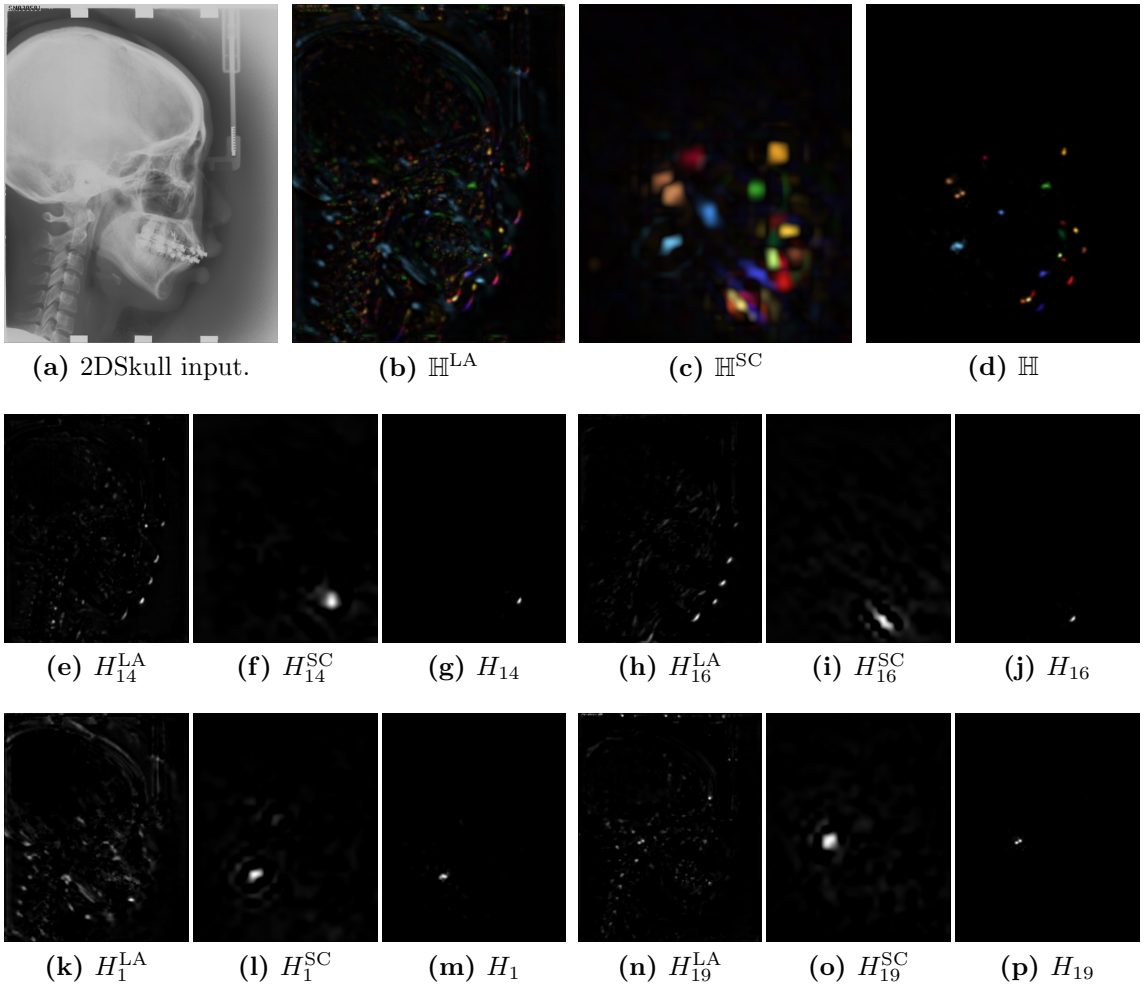
(e) H_{25}^{LA} **(f)** H_{25}^{SC} **(g)** H_{25} **(h)** H_{4}^{LA} **(i)** H_{4}^{SC} **(j)** H_{4}

(k) H_{13}^{LA} **(l)** H_{13}^{SC} **(m)** H_{13} **(n)** H_{17}^{LA} **(o)** H_{17}^{SC} **(p)** H_{17}

Figure 7.15: Example input and result images from the 3DHand dataset visualized via maximum projection. The first row shows the input image, the combined local appearance ($\mathbb{H}^{\mathrm{LA}}$), spatial configuration ($\mathbb{H}^{\mathrm{SC}}$), and final ($\mathbb{H}$) heatmap outputs for all landmarks. The second and the third row show local appearance (H^{LA}), spatial configuration (H^{SC}), and final (H) heatmap outputs for landmarks 25, 4, 13, and 17.

landmarks $\mathcal{L}_{13}$ (on the base of the distal phalanx on the middle finger) and $\mathcal{L}_{17}$ (the base of the distal phalanx on the ring finger) the *local appearance* heatmaps H_{13}^{LA} and H_{17}^{LA} are almost identical. Thus, in contrast to landmark $\mathcal{L}_{25}$ the maximum responses of H_{4}^{LA}, H_{13}^{LA}, and H_{17}^{LA} have a high chance of being on the wrong landmark (see the poor results of SCN-LA in Table 7.7 and Fig. 7.13). However, the *spatial configuration* heatmaps H_{4}^{SC}, H_{13}^{SC}, and H_{17}^{SC}, although only predicting a rough location, allow an easy distinction of the landmarks. Combining the *local appearance* and *spatial configuration* heatmaps via multiplication then leads to the final accurate and distinct heatmaps H_4, H_{13}, and H_{17}.

Interestingly, for similar-looking *local appearance* heatmaps also the learned σ_i values are almost the same, as can be seen in Table 7.6 and Fig. 7.11. This further confirms that



(a) 2DSkull input. **(b)** $\mathbb{H}^{\mathrm{LA}}$ **(c)** $\mathbb{H}^{\mathrm{SC}}$ **(d)** $\mathbb{H}$

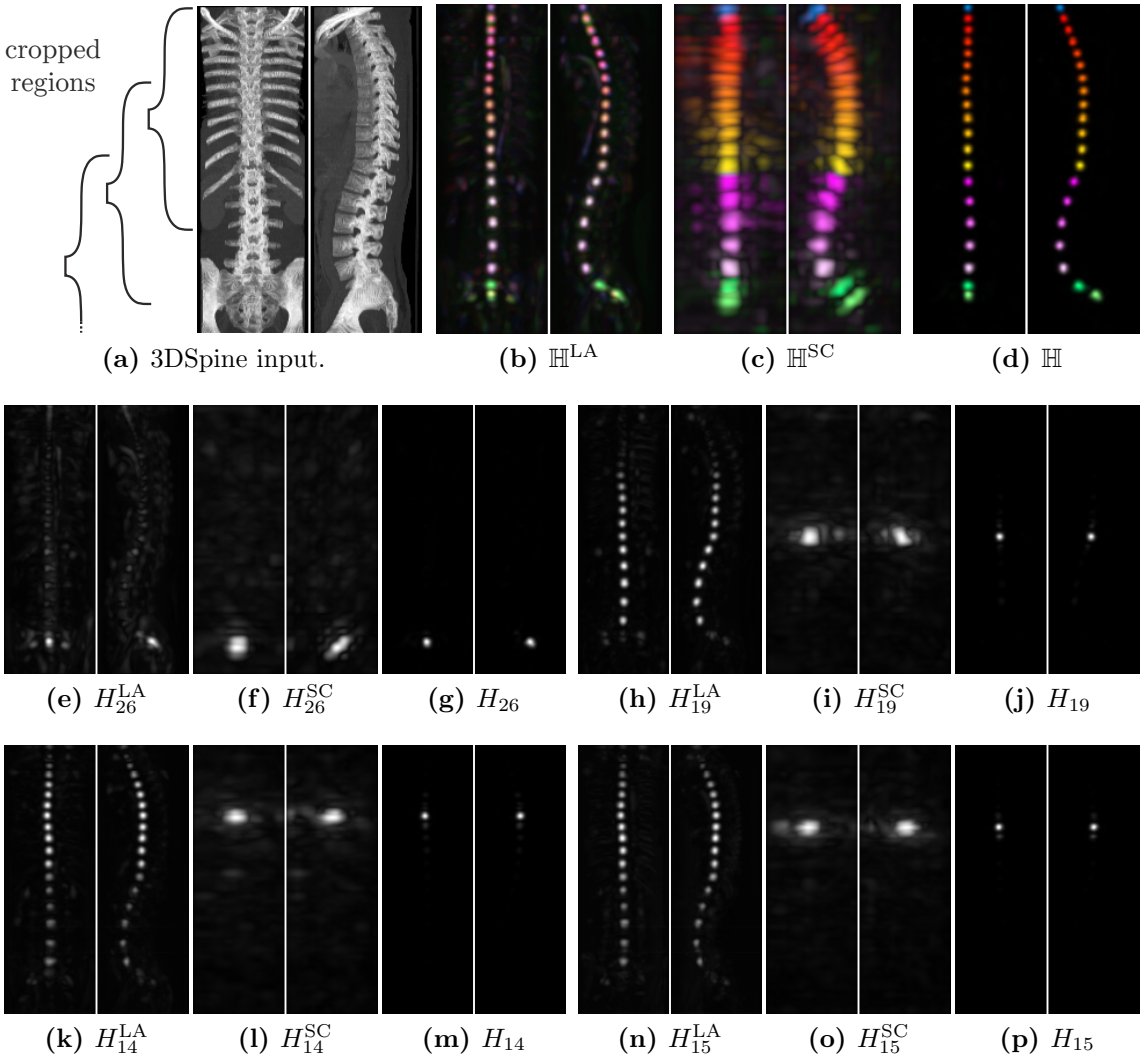
(e) H_{14}^{LA} **(f)** H_{14}^{SC} **(g)** H_{14} **(h)** H_{16}^{LA} **(i)** H_{16}^{SC} **(j)** H_{16}

(k) H_{1}^{LA} **(l)** H_{1}^{SC} **(m)** H_{1} **(n)** H_{19}^{LA} **(o)** H_{19}^{SC} **(p)** H_{19}

Figure 7.16: Example input and result images from the 2DSkull dataset. The first row shows the input image, the combined local appearance ($\mathbb{H}^{\mathrm{LA}}$), spatial configuration ($\mathbb{H}^{\mathrm{SC}}$), and final ($\mathbb{H}$) heatmap outputs for all landmarks. The second and the third row show local appearance (H^{LA}), spatial configuration (H^{SC}), and final (H) heatmap outputs for landmarks 14, 16, 10, and 19.

the learned σ values represent a prediction uncertainty of the heatmap regression CNN.

As a consequence of the similar-looking *local appearance* heatmaps for some landmarks (see Figs. 7.14h, 7.14k and 7.14n), for generating these heatmaps a lot of network parameters can be shared. Thus, the *local appearance* part of the SCN may achieve the same overall prediction performance with a reduced number of network parameters as compared to, e.g., the U-Net. Additionally, the similar-looking heatmaps do not use information from only one, but from multiple landmarks, which serves as additional training data augmentation. While in the U-Net every heatmap is learned from its corresponding landmark annotations only, in the SCN the similar-looking *local appearance* heatmaps *share* the training data annotations from multiple different landmarks. For example, for generating



(a) 3DSpine input. **(b)** $\mathbb{H}^{\mathrm{LA}}$ **(c)** $\mathbb{H}^{\mathrm{SC}}$ **(d)** $\mathbb{H}$

(e) H_{26}^{LA} **(f)** H_{26}^{SC} **(g)** H_{26} **(h)** H_{19}^{LA} **(i)** H_{19}^{SC} **(j)** H_{19}

(k) H_{14}^{LA} **(l)** H_{14}^{SC} **(m)** H_{14} **(n)** H_{15}^{LA} **(o)** H_{15}^{SC} **(p)** H_{15}

Figure 7.17: Example input and result images from the 3DSpine dataset visualized via maximum projection. The first row shows the input image with the cropped and overlapping region size, the combined local appearance ($\mathbb{H}^{\mathrm{LA}}$), spatial configuration ($\mathbb{H}^{\mathrm{SC}}$), and final ($\mathbb{H}$) heatmap outputs for all landmarks. The second and the third row show local appearance (H^{LA}), spatial configuration (H^{SC}), and final (H) heatmap outputs for landmarks $\mathcal{L}_{26}$, $\mathcal{L}_{19}$, $\mathcal{L}_{14}$, and $\mathcal{L}_{15}$.

the single heatmap that represents all *local appearance* heatmaps on the five fingertips, the annotations of all five fingertips are used, thus, effectively increasing the number of annotations per heatmap by a factor of five. These properties further indicate that the SCN requires a much lower number of training images as other CNN architectures.

3DHand: A representative example for the dataset of MR images of the hand is given in Fig. 7.15. This figure shows example heatmap outputs for the same landmarks as used for the 2DHand dataset in Fig. 7.14. Despite being a volumetric dataset, the same properties and observations regarding *local appearance* and *spatial configuration* hold.

Thus, by looking at the *local appearance* heatmaps, locally similar landmarks would not be distinguishable, but when combined with the *spatial configuration* heatmaps, the final heatmap has a single and pronounced maximum response on the position of the landmark.

2DSkull: Figure 7.16 shows an example of the dataset of cephalograms. Although this dataset also contains landmarks on locally similar structures (e.g., $\mathcal{L}_{14}$ and $\mathcal{L}_{16}$), the individual landmarks are much better locally distinguishable, in contrast to e.g., the landmarks on the phalanges for the 2DHand and 3DHand datasets. Similar to these datasets, the individual *local appearance* heatmaps show responses on locally similar structures, while the *spatial configuration* heatmaps show single but coarse responses. The combined final heatmaps then remove the false positive responses on infeasible landmark locations.

As compared to the 2DHand and 3DHand datasets, many landmarks are anatomically ill-defined, e.g., $\mathcal{L}_{10}$ on the jaw bone. As this landmark lies on a smooth edge and not on a sharp corner, it is difficult for experts to annotate this landmark unambiguously and consistently. Due to such inconsistent annotations, the network has troubles in creating heatmaps with a single pronounced maximum at the correct location of the landmark, as can be seen in Figs. 7.16k to 7.16m. Moreover, as lateral cephalograms are 2D projections, the volumetric information is lost. This can be observed for the left and right temple (see landmark $\mathcal{L}_{19}$), which are spatially separated in 3D, but close together and difficult to distinguish in these 2D projections. Due to this ambiguity, the SCN creates a heatmap with two local maxima, and only by chance the maximum at the annotated position is taken, as can be seen in Figs. 7.16n to 7.16p.

3DSpine: Figure 7.17 shows an example input image for the dataset of spine CT images. In this dataset, all landmarks lie in the center of the individual vertebrae, sharing similar local appearances. Only the landmarks on the sacrum, which is shaped differently as compared to other vertebrae, have *local appearance* heatmaps with a single response (see Fig. 7.17e). For the other landmarks, the *local appearance* heatmaps have responses on neighboring, similar-looking vertebrae, as can be seen in Figs. 7.17h, 7.17k and 7.17n. However, these slight differences among the *local appearance* heatmaps of neighboring vertebrae are sufficient for the SCN to identify the correct location and to create a graphical model as *spatial configuration* heatmaps with single responses, leading to final results that are both accurate and robust towards landmark misidentification.

7.4 Reducing the Number of Training Images

In this section, we present results of an experiment on the 2DHand dataset with a drastically reduced number of training images. Due to the simplification of the localization task into *local appearance* and *spatial configuration* in the SCN, we hypothesize that the network needs less annotated training data than other fully convolutional network architectures for heatmap regression. Therefore, we designed an experiment for the 2DHand dataset with its total of 895 images to systematically evaluate how our proposed SCN performs when decreasing the number of training images, while still using the same validation images

# Training Images	Method	PE_{all} (in mm)		$\#O_r$ (in %)		
		Median	Mean ± SD	$r = 2$ mm	$r = 4$ mm	$r = 10$ mm
10	SCN:	**0.55**	**0.91 ± 1.13**	**3564 (10.76%)**	**707 (2.14%)**	**30 (0.09%)**
	U-Net:	0.71	2.00 ± 9.38	4855 (14.66%)	1627 (4.92%)	633 (1.91%)
50	SCN:	**0.45**	**0.72 ± 0.84**	**2204 (6.66%)**	**379 (1.15%)**	**8 (0.02%)**
	U-Net:	0.59	1.19 ± 5.44	2977 (8.99%)	656 (1.98%)	199 (0.60%)
100	SCN:	**0.44**	**0.69 ± 0.81**	**1895 (5.72%)**	**293 (0.89%)**	**8 (0.02%)**
	U-Net:	0.59	1.09 ± 4.68	2766 (8.35%)	539 (1.63%)	151 (0.46%)
≈ 600	SCN:	**0.43**	**0.66 ± 0.74**	**1659 (5.01%)**	**241 (0.73%)**	**3 (0.01%)**
	U-Net:	0.44	0.70 ± 2.18	1703 (5.14%)	270 (0.82%)	22 (0.07%)

Table 7.8: Cross-validation results on 895 images from 2DHandReduced dataset, averaged from three random selections of training images.

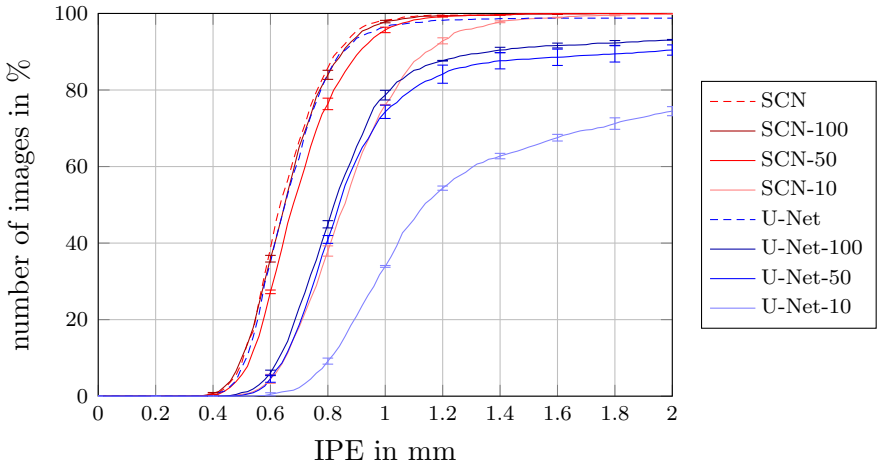


Figure 7.18: Cumulative distribution of IPE on 2DHandReduced dataset for different numbers of training images.

and three-fold cross-validation setup as before. Thus, to generate the 2DHandReduced dataset, instead of using all 600 training images per fold, we solely train on a random subset of 10, 50, or 100 images, while we perform inference on the same 300 images per fold, such that still every image of the full dataset is tested exactly once. Other training hyperparameters remain the same as in Section 7.1.1. For each cross-validation round of 10, 50, and 100 training images, we report our evaluation metrics as averages from three experiments with different random training images to compensate biased choices of the selected subsets of training images.

Results of this experiment in Fig. 7.18 and Table 7.8 show that for smaller training datasets, our SCN performs much better than the U-Net, since our SCN-10 has 30 outliers at $r = 10$ mm when trained from solely 10 images, while U-Net-10 trained from 10 images has 633 outliers for the same error radius. Even when trained on 100 images, U-Net-100 still has 151 outliers. Interestingly, already when training with 50 images, the landmark localization performance of our SCN-50 is almost the same as the original SCN trained on

the full 2DHand dataset. This experiment shows that increasing the number of training images leads to better localization performance of CNN based methods in general, while it also confirms our hypothesis that by incorporating *spatial configuration* inside our SCN, a smaller amount of data is sufficient for training.

7.5 Summary

In this chapter, we have evaluated the performance of our proposed method for anatomical landmark localization, as well as examined our proposed loss function and SCN in more detail. We have shown that our method outperforms state-of-the-art methods on four datasets for anatomical landmark localization, i.e., radiographs of left hands, volumetric MR scans of left hands, lateral cephalograms, and volumetric CT scans of the spine. Furthermore, we have compared several network architectures and evaluated several hyperparameters of our SCN on the dataset of hand radiographs to determine the optimal network architecture. We have also evaluated our proposed loss function and have shown that the learned sizes of the Gaussian heatmaps correlate with the expected point error. By analyzing the individual components of the SCN, we have shown that both components work as expected, i.e., modeling both the *local appearance* and *spatial configuration* of landmarks. Finally, we have shown that our proposed SCN greatly outperforms the U-Net on the dataset of hand radiographs with a drastically reduced number of training images, thus, effectively using information of the *spatial configuration* of anatomical landmarks.

8

Applications and Evaluations on Other Domains

Contents

8.1 Landmark Localization in Computer Vision Datasets 114

8.2 Fully Automatic Age Estimation 119

8.3 Multi-Label Whole Heart Segmentation 128

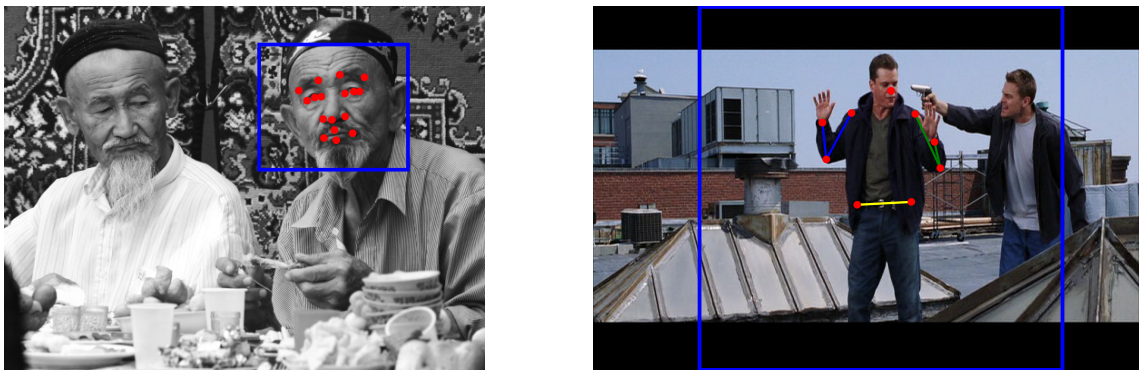
8.4 Vertebrae Segmentation 136

8.5 Summary 145

I'm doing everything I can... And stop calling me Shirley!

Dr. Rumack

In this chapter, we show several applications of our proposed landmark localization method. We show generic applicability of the SpatialConfiguration-Net (SCN) on computer vision dataset in Section 8.1, where we evaluate on the tasks face point detection and human pose estimation. We present a forensic application of anatomical landmark localization for age estimation of living individuals from structures in Magnetic Resonance (MR) images from hands, clavicles, and teeth in Section 8.2. In Section 8.3, we show how to extend the SCN for multi-label segmentation at challenging Computed Tomography (CT) and MR datasets for multi-label whole-heart segmentation. Finally, in Section 8.4, we demonstrate how to apply our method for anatomical landmark localization to segment individual vertebrae in spine CT volumes. We summarize findings of this chapter in Section 8.5.



(a) AFLW **(b)** FLIC

Figure 8.1: Example images of the AFLW dataset for face point detection and the FLIC dataset for human pose estimation. The red dots indicate landmark locations; the blue boxes represent the cropped regions that are used as inputs for the networks.

8.1 Landmark Localization in Computer Vision Datasets

To show the applicability of our method to tasks of the computer vision domain, we evaluate our method for landmark localization on datasets for face point detection, as well as human pose estimation. As many related works using Random Forests (RFs) and Convolutional Neural Networks (CNNs) were evaluated on the datasets used in this section (see Chapter 3), the results presented in this section further demonstrate how our proposed method compares to the state-of-the-art. The method and results presented in this section have been published in [110].

8.1.1 Method

Same as for the anatomical landmark localization tasks of the previous chapter, we use the heatmap regression framework with our proposed loss function to learn the heatmap peak widths (see Chapter 4) and apply our proposed SCN architecture that explicitly learns the *local appearance* and the *spatial configuration* of landmarks (see Chapter 5). As there may be more than one human face or body visible in the evaluated datasets (see Fig. 8.1), the network needs to know, for which person in the image the landmarks should be extracted. Therefore, following the literature on the evaluated datasets, we crop the input for the networks such that only the face of the target person is visible (for face point detection) and the target person is centered in the image (for human pose estimation), respectively.

8.1.2 Evaluation

We evaluate our SCN architecture as well as the U-Net and compare them to state-of-the-art results on a dataset for face point detection as well as a dataset for human pose estimation. Due to the lack of other reported measures in the respective papers of the

compared methods, we only show plots of the cumulative distributions of the image specific point-to-point error (IPE) as defined in Eq. (6.4).

8.1.2.1 Datasets

The face point detection and human pose estimation datasets are as follows:

Face Point Detection Dataset (AFLW): The AFLW dataset consists of images from Flickr and shows a large variation in poses and illumination [77]. Additionally, some of the images contain more than one face. To assure that the network only detects the points from the currently evaluated face, we need to crop the face beforehand. We use the same cropped face images as [85], who provided us with their setup of 326 training and 4755 testing images of AFLW, as well as their additional point annotations. As the top and bottom points on the lips are missing in the original AFLW dataset, Lindner et al. [85] manually annotated these additional two points. While we use the same setup of 326 training and 4755 testing images with 17 annotated landmarks as [21, 85], in [161] 1000 different testing images and 19 point annotations are used. Due to the lack of RGB information in the annotations from [21, 85], we use grayscale images as provided by the authors as input to predict 17 landmarks.

Human Pose Estimation Dataset (FLIC): Compared to face point detection, the variety of human poses is much larger, making it a very challenging problem that has recently seen a lot of interest in computer vision. We evaluate our algorithm on the FLIC dataset introduced in [121], consisting of 3987 training and 1016 testing images from various Hollywood movies with actors in mainly front-facing standing up poses. Each image is annotated with 11 landmarks, i.e., face, eyes, shoulders, elbows, wrists, and hips. As many images of the FLIC dataset contain more than one person, we follow the same approach as [100] and move the center x coordinate of the person's bounding box to the center of the input image. Thus, the network learns to solely estimate the pose of the person in the center. We use the full 3-channel RGB input and localize seven landmarks, i.e., head, shoulders, elbows, and wrists.

8.1.2.2 Implementation Details

Training and testing of the network were done with our framework as described in Chapter 6. We use the same preprocessing and augmentation hyperparameters as for the 2D datasets of anatomical landmark localization that we defined in Section 6.2.

Regarding the network architecture of our SCN, we again use the same implementation and hyperparameters as defined in the previous chapter. Due to the smaller image sizes of the computer vision datasets compared to the datasets for anatomical landmark localization, we use networks with 256×256 pixels input and heatmap target size, while the *spatial configuration* component operates at $1/8^{\text{th}}$ of the input resolution. We train the networks with a mini-batch size of one for 100,000 iterations for AFLW, and 1,100,000 iterations for FLIC, respectively.

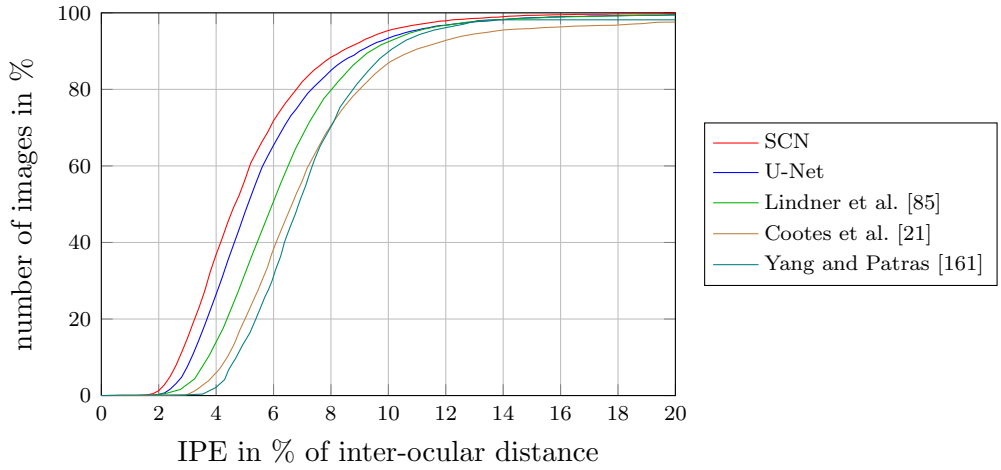


Figure 8.2: Cumulative distribution of IPE on the AFLW dataset.

8.1.2.3 Results

We present the results of our approach and compare it to state-of-the-art methods of the respective datasets.

Face Point Detection Dataset (AFLW): We compare our SCN method to [21, 85], who use constrained local models, as well as to [161], who use sieved random forests. Additionally, we also compare it to our implementation of the U-Net. Note that although there exist many more methods that are evaluated on the AFLW dataset, we mainly use this dataset to compare to methods based on RF that were initially proposed for anatomical landmark localization. As the results of the compared RF based methods are evaluated on different training/validation/testing images [85], a fair comparison to other presented results would not be possible.

To alleviate different scales throughout the dataset, same as the compared methods, we normalize the point-to-point errors with the inter-ocular distance, i.e., $s^{(j)} = 1/\|\mathbf{x}^{*(j)}_{\text{l\_eye}} - \mathbf{x}^{*(j)}_{\text{r\_eye}}\|_2$ as used in Eq. (6.3).

The results are shown quantitatively in Fig. 8.2 in a plot of the cumulative image-specific point-to-point error distributions, which present the proportion of tested images that achieve a certain IPE. Similar to Lindner et al. [85], we use the face point detection application to confirm that, despite being developed for medical image analysis tasks, our proposed SCN also gives a state-of-the-art performance on a computer vision dataset. From our results, we see that even without modifications, our proposed SCN outperforms both U-Net and the previous state-of-the-art facial landmark localization approach [85].

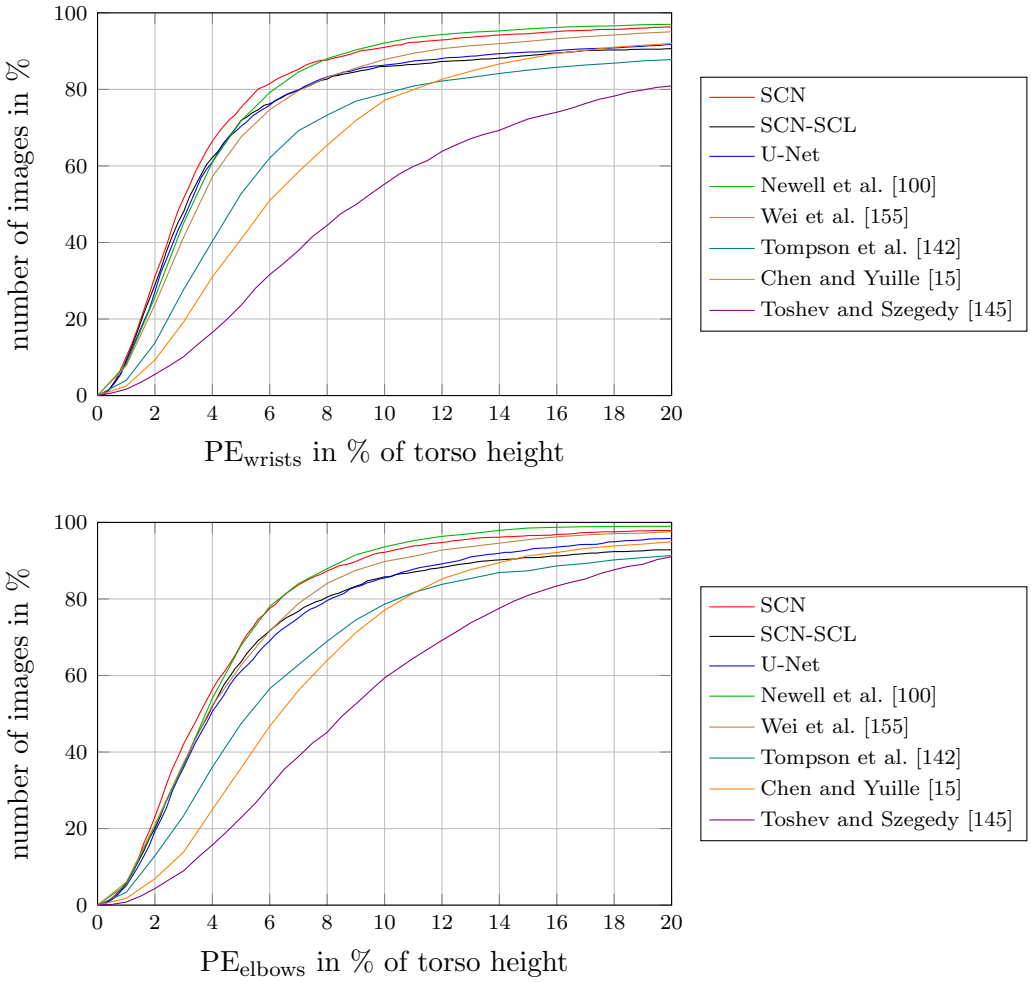


Figure 8.3: Cumulative distribution of PE$_{\text{wrist}}$ and PE$_{\text{elbows}}$ on the FLIC dataset.

Human Pose Estimation Dataset (FLIC): We compare our results to various methods from the literature that represent the latest developments in human pose estimation like coordinate regression on its own [145] or refined with a handcrafted graphical model [15], as well as heatmap regression [142] and cascades of fully convolutional CNNs [100, 155]. Again, we also compare to our U-Net heatmap regression baseline.

We follow a common evaluation protocol [100] by solely reporting the overall point-to-point error rate of elbows and wrists, respectively, and by evaluating on observer-centric annotations, meaning that left and right landmarks are defined from the observer's perspective. To alleviate different scale throughout the dataset, we use the approach proposed by Sapp and Taskar [121] and normalize the point-to-point errors with the torso height, defined by the distance of left hip and right shoulder, i.e., $s^{(j)} = 1/\|\mathbf{x}^{*(j)}_{\text{l\_hip}} - \mathbf{x}^{*(j)}_{\text{r\_shoulder}}\|_2$ as used in Eq. (6.3).

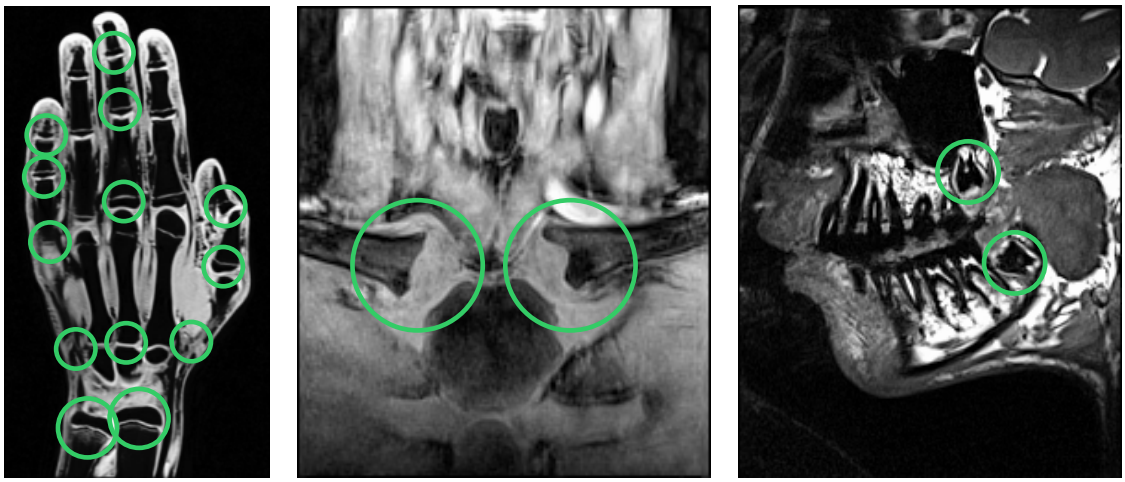
The results in Fig. 8.3 show that our proposed SCN outperforms all state-of-the-art

methods in human pose estimation with the exception of the cascade approach from [100], which performs slightly worse in accuracy but better in robustness towards landmark misidentification. However, due to their cascaded refinement CNN architecture, their model complexity is larger than in our SCN, and their approach is solely dedicated and therefore finetuned to the task of human pose estimation. Differently to the medical datasets in Chapter 7, here the results of the single-stage U-Net approach are no longer competitive due to the large variation in the human pose estimation dataset. To improve performance, the U-Net based architecture could benefit from the larger training dataset by extending it to a cascade of U-Net stages, which would lead to a method similar to [100]. On the other hand, without any modifications or extensions, our method that uses constraints on the *spatial configuration* shows its benefits not only in the presence of small training datasets as demonstrated in Section 7.4, but it is also capable of capturing large variation in bigger datasets. Compared to [15] and [142], who use Markov Random Field (MRF) based constraints on the *spatial configuration*, we improve by a large margin due to our *spatial configuration* component being a deep CNN that can be trained together with the *local appearance* component in an end-to-end manner.

While works like [142] and our preliminary SCN of [106] have shown that even a single convolution layer can be used to model an MRF in a CNN framework, we hypothesize that our *spatial configuration* component as described in Chapter 5 is able to be more robust to the presence of complex variations in landmark configuration, as present in body pose estimation. Thus, to more deeply investigate the influence of the number of convolution layers in the *spatial configuration* component, we have also performed an ablation study by evaluating a version of our SCN, where only a single convolution layer is used in the *spatial configuration* component (SCN-SCL). Results of this comparison in Fig. 8.3 show that in both accuracy and robustness towards landmark misidentification, SCN-SCL already outperforms [142]. However, our proposed SCN provides better results than both [142] and SCN-SCL, since it is capable of modeling more complex spatial relationships due to its additional convolution layers that address the larger variety of human poses. Thus, our end-to-end trained fully convolutional SCN with its deep *spatial configuration* component is able to achieve results that reproduce the best performing state-of-the-art method from Newell et al. [100] on a large training dataset for human pose estimation.

8.1.3 Conclusion

Our experiments show the general applicability of our proposed SCN for heatmap regression also for landmark localization tasks in the computer vision domain. The SCN is able to achieve results that reproduce the best performing but task-specific state-of-the-art method from [100] on pose estimation, while outperforming all other evaluated methods on both studied computer vision datasets. Thus, we have demonstrated that our SCN can not only be used in medical imaging applications, but also for face point detection and human body pose estimation.



(a) Hand. **(b)** Clavicles. **(c)** Wisdom Teeth.

Figure 8.4: Images showing the structures containing relevant information for age estimation, marked with green circles. All images show slices of volumetric MR images. In the hand images, the relevant information is in the epiphyseal gaps of the bone ends of the phalanges and metacarpals, as well as radius and ulna. In the clavicles, the information is present in the epiphyseal gaps as well. In the teeth, the age-relevant information is in the wisdom teeth.

8.2 Fully Automatic Age Estimation

Age estimation of living individuals is an active research field in clinical medicine, e.g., to assess endocrinological diseases [92] or to plan orthopedic interventions [82, 153], as well as legal medicine to robustly distinguish minors from majors. To estimate the *chronological age* in children and adolescents, the anatomical changes during physical maturation can be investigated by non-invasive, imaging-based radiological methods. Predominantly the ossification of bones [48, 139] and the mineralization of wisdom teeth [26] allows experts in forensic radiology and forensic dentistry the examination of the biological development of a subject. As different anatomical structures only show the physical maturation in specific age ranges (hand bones: < 18 years, clavicles: $16 - 25$ years, wisdom teeth: $13 - 25$ years), the information of multiple complementary sites needs to be combined to make a robust age estimate at a broad age range. Structures containing relevant information for age estimation in hand bones, clavicles, and wisdom teeth are shown in Fig. 8.4.

As the established radiological methods for assessing biological development suffer from intra- and inter-rater variability [68], in our working group we experimented with machine-learning-based fully automatic age estimation from MR images of the left hand, upper thorax, and the jaw. The biological development in these images is mainly visible in the epiphyseal gaps of the finger and wrist bones, the epiphyseal gaps of the clavicles, and the mineralization of the wisdom teeth, respectively. Hence, the relevant information for estimating the age is restricted to only small regions of the entire MR volumes. To

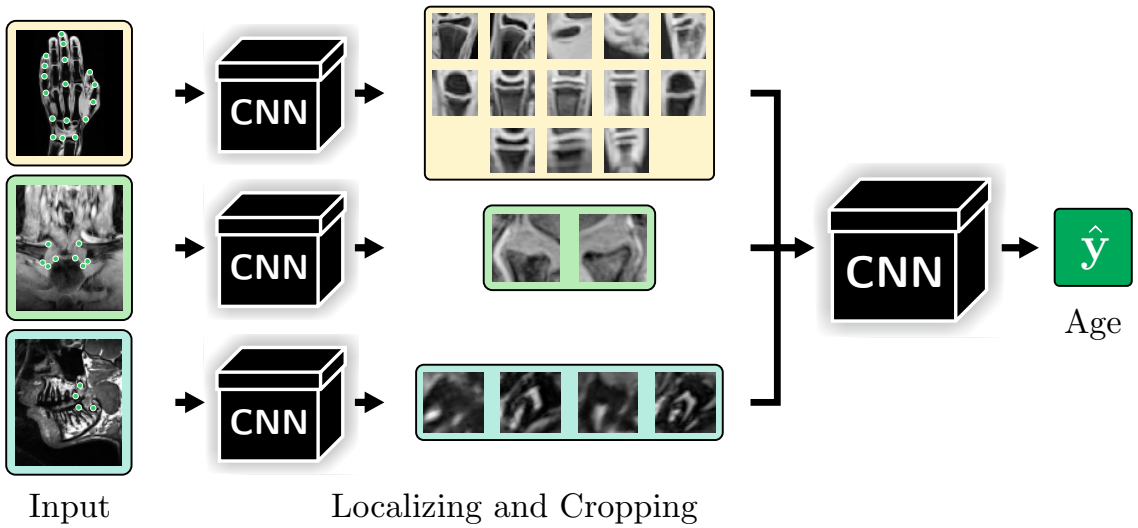


Figure 8.5: Overview of our method for fully automatic age estimation. A first CNN localizes the structures relevant for age estimation. Cropped images around these regions are then used as input for a second CNN that performs the age estimation.

increase the performance of machine learning methods, they should be trained to focus on these regions containing the structures of interest.

The method and results presented in this section were published in [131, 133–135].

8.2.1 Method

For our proposed methods for fully automatic age estimation from images of the hand ([134, 135]), as well as images from hands, clavicles, and wisdom teeth combined ([131, 133]), we train CNNs to focus on the relevant anatomical structures by cropping around the regions containing information of physical maturation, i.e., regions around the specific bone ends for hands and clavicles, and the regions containing wisdom teeth (see Fig. 8.5). Thus, we localize the relevant structures on the MR images at first with a CNN for anatomical landmark localization, i.e., our proposed SCN for heatmap regression. By locating two anatomical landmarks per bone for the hand MR images, we crop the same 13 bones that are used in the Tanner-Whitehouse RUS method for age estimation (TW2) [139]. In MR data of the upper thorax, we crop the two clavicle bones separately based on two identified landmarks for each clavicle. The regions encapsulating wisdom teeth are extracted from the dental MRI data using the locations of the centers of the second and third molars. After localizing the age-relevant structures, we use the cropped regions around the structures to perform age estimation with a second CNN (see Fig. 8.5).

Localizing and cropping the regions containing the age-relevant information has multiple advantages. First, the input image domain is simplified by eliminating pose variations in the images since the relevant structures are aligned to the same canonical position. This allows the CNN to focus only on the developmental stages of the bones/teeth and removes

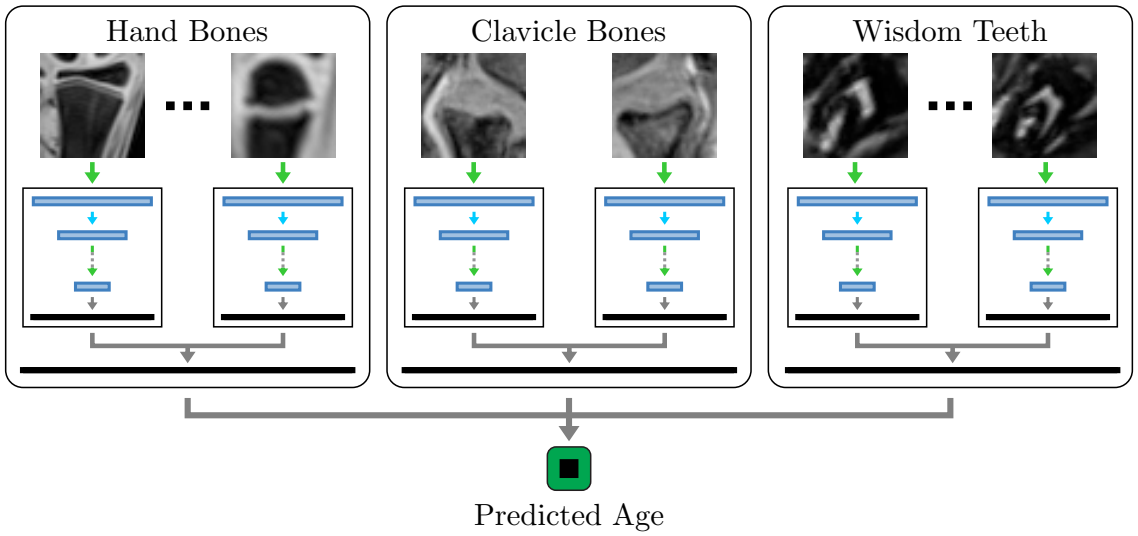


Figure 8.6: Schematic representation of our CNN for age estimation using cropped regions around age-relevant structures as input. Each cropped bone/tooth containing age-relevant structures has its own feature extraction and encoding block that consists of consecutive convolution and pooling layers and finishes with a fully connected layer. The outputs of the three individual sites (hand bones, clavicle bones, wisdom teeth) are generated with another fully connected layer, which are then combined with a final fully connected layer leading to the final age estimate. Blue boxes represent (intermediate) images; black boxes represent the feature vectors; arrows represent connections, i.e., → convolution, → downsampling, → fully connected.

the chance of overfitting to irrelevant information, e.g., pose, size, and image background. Moreover, due to memory restrictions, when using the whole acquired MR volumes as input for the CNN, the input volumes would need to be downsampled, possibly removing information at the relevant anatomical structures. Therefore, cropping enables the age estimation CNN to observe these structures in higher resolution.

Our proposed CNN architecture for assessing bone ages from such cropped regions is shown in Fig. 8.6. It consists of per-bone/tooth feature extraction and encoding blocks that are combined with fully connected layers. Each block consists of multiple repeated levels of two consecutive convolution layers with $3 \times 3 \times 3$ kernels and one max-pooling layer, where Rectified Linear Unit (ReLU) is used as the nonlinear activation function. The block finishes with a fully connected layer, leading to an encoded output feature representation for each cropped bone/tooth. To fuse feature representations into a single feature vector per anatomical site (hand bones, clavicle bones, wisdom teeth), we use a fully connected layer followed by a ReLU activation unit. A final fully connected layer with a single output combines the feature outputs for the three sites and predicts the final age estimate. We train the networks with the L_2 loss function comparing network prediction and ground-truth chronological age (see Eq. (2.9)).

8.2.2 Evaluation

We evaluate several methods for fully automatic age estimation on our in-house dataset consisting of MR volumes of the left hand, upper thorax, and the jaw. First, we use only the hand images to compare baseline experiments for age estimation with RFs and CNNs on the cropped input images containing. There we also compare to a CNN using the whole volume of the hand as input, and visualize from which regions on the input image the CNN takes the information to generate its age prediction. Second, we use cropped images of the three sites as input for our age estimation CNN and evaluate their individual contribution to the final age prediction.

8.2.2.1 Dataset

We evaluate the methods for age estimation on an in-house dataset of MR volumes of the left hand, upper thorax, and the jaw. This dataset was collected at the Ludwig Boltzmann Institute for Clinical Forensic Imaging in Graz as part of a study investigating the role of Magnetic Resonance Imaging (MRI) in forensic age estimation. It contains $N = 322$ subjects with known chronological age ranging between 13.0 and 25.0 years (mean $\pm$ std: 19.1 ± 3.3 years). T1-weighted 3D gradient echo sequences with fat saturation were used for acquiring the hand and clavicle data (physical voxel resolutions of $0.45 \times 0.45 \times 0.9$ mm^3 and $0.9 \times 0.9 \times 0.9$ mm^3, respectively), while teeth were scanned using a proton density-weighted turbo-spin-echo sequence ($0.59 \times 0.59 \times 1.0$ mm^3). Voxel sizes of the whole input volumes were $288 \times 512 \times 72$ for hand, $168 \times 192 \times 44$ for clavicles, and $208 \times 256 \times 56$ for wisdom teeth, respectively. Acquisition times of hand, clavicles, and wisdom teeth MR sequences were around 4, 6, and 10 min, respectively, but show potential for further acceleration through undersampling [99].

8.2.2.2 Implementation Details

For data preprocessing and augmentation as well as training the CNNs we use the framework as described in Chapter 6. For our experiments only incorporating information from the hand images, due to the age-relevant information being present in the epiphyseal gaps, we additionally compare to images with enhanced epiphyseal gaps to evaluate the automatic feature extraction capabilities of the individual methods. As in the cropped images, the long bones are aligned to a fixed axis, the images with enhanced epiphyseal gaps are created by using a Laplacian of Gaussian filter with different values for σ along this axis. See [135] for more details on this preprocessing step.

For all experiments, the intensity values of the input images for the RFs and CNNs are rescaled such that they lie approximately within [-1, 1]. For training data augmentation, we use values randomly sampled from a uniform distribution within the following intervals. The intensity values were shifted by $[-0.1, 0.1]$ and scaled by $[0.8, 1.2]$. Additionally, the cropped MR bone images were geometrically transformed using a translation

by $[-2, 2]$ mm, scaling by $[0.85, 1.15]$, and rotation by $[-5°, 5°]$ in each dimension.

Due to an imbalanced dataset with regards to the age distribution, during training, we randomly sample images such that subjects within each full year are represented uniformly across the whole age range.

RF: Compared to CNNs, which automatically learn features relevant for the task, features used by the RF have to be constructed manually. Since after cropping, the epiphyseal plate is orthogonal to the axis along the bone image, we generate features for the RF as an average image value along a randomly generated line parallel to this axis. In each tree node of the RF, a feature is generated by randomly selecting a hand bone out of the set of 13 possible hand bones, followed by randomly generating a bone image coordinate as well as a specific length along the bone axis.

For training our Random Regression Forest (RRF) we identify the best discriminative features and thresholds by maximizing the information gain [24, 135]. During testing, the feature response is computed for a test subject based on the stored feature parameters and thresholds. To generate the final age prediction, the individual age predictions from the reached leaf nodes of all trees in the RF are combined as a truncated mean after discarding 5% of ages with the highest and 5% with the lowest values, respectively.

CNN: The per-bone feature extraction and encoding blocks in Fig. 8.6 are set up to consist of two consecutive convolution layers followed by a max pooling at every level. At each of the three levels, the convolution layers are set up to generate 24, 48, and 96 intermediate outputs, respectively, followed by a fully connected layer with 96 outputs that represent the extracted bone/tooth feature vectors. Further increasing the number of intermediate outputs was not feasible, due to its demands on GPU memory consumption. The fused feature outputs for each one of the three anatomical sites were generated by a fully connected layer with 96 outputs, followed by generating the final output with a last fully connected layer. For the experiments on the hand images only in Section 8.2.2.3, we do not use a single CNN using hyperparameters as described in this section but ensembles of CNNs using different numbers of feature outputs. Refer to [135] for the individual numbers for the individual CNNs of the ensemble. Optimization was done with the Adam optimizer [71] with a maximum of $20,000$ iterations, a mini-batch size of 8, and a learning rate of 10^{-5}. For the CNN trained on the whole hand, we use $40,000$ iterations. We perform L_2 weight decay with a factor of 0.0005, as well as dropout [129] with a ratio of 0.5 before the fully connected layers to reduce overfitting.

8.2.2.3 Age Estimation from Hand MR Images

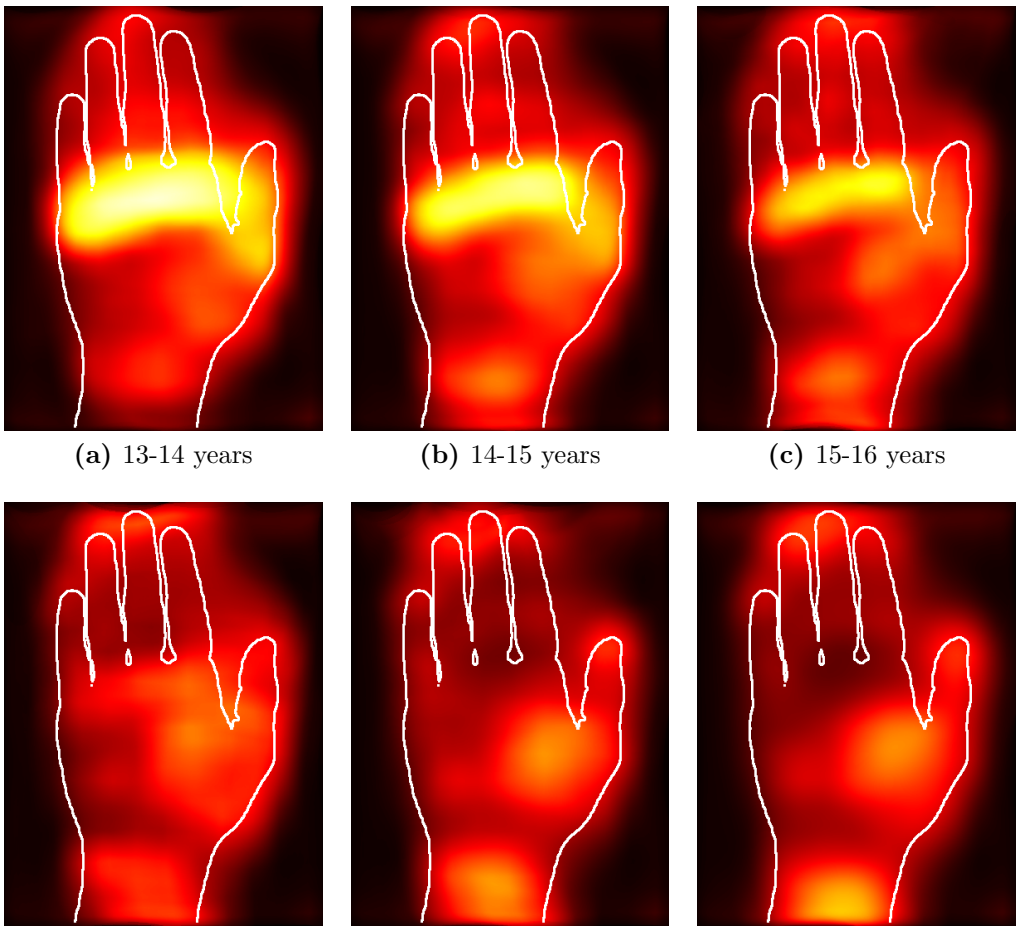
As a baseline, we evaluate various methods for predicting the chronological age of subjects from MR images of left hands. As the ossification process of the epiphyseal gaps of the hands is finished at around 18 years, it is not possible to distinguish between an 18-year-old and a 25-year-old from images of the hand only. Thus, we restrict training on 141 subjects below 18 years for estimating the chronological age and perform four-fold cross-validation.

Input	Method	MAE chronological age Mean ± SD
cropped	CNN	**0.82 ± 0.65**
	RF	1.48 ± 1.00
cropped with enhanced gaps	CNN	0.83 ± 0.62
	RF	0.96 ± 0.74
whole image	CNN	0.96 ± 0.77

Table 8.1: Results of fully automatic age estimation methods for a four-fold cross-validation on hand MR images of 141 subjects younger than 18 years. Values are shown for RFs and CNNs using the unmodified raw cropped input images, as well as cropped images with enhanced epiphyseal gaps. For the CNNs, we also show results when training on whole downsampled input images. The MAE shows the mean and standard deviation of the absolute error of each individual age prediction to the ground-truth age of the 141 subjects.

We compare RFs and CNNs trained on cropped images of the bone ends, both without and with enhancing the epiphyseal gaps as preprocessing, as well as CNNs trained on downsampled images of whole hands. The results of four-fold cross-validation, comparing the MAE of the ground-truth and predicted age of the different methods are presented in Table 8.1. When comparing the RFs methods with the CNNs on the cropped images, we can see that the CNNs perform much better than the RF-based methods, reaching an MAE of 0.82 ± 0.65 as compared to 0.96 ± 0.74. Also, for the RF to be competitive, the epiphyseal gaps must be enhanced, as otherwise, the results are much worse with an MAE of 1.48 ± 1.00. For the CNNs, both raw input images, as well as images with enhanced gaps, lead to almost the same performance (0.82 ± 0.65 with, 0.83 ± 0.62 without enhancement). This further confirms that the age-relevant information is in the epiphyseal gaps and that the feature extraction capabilities of CNNs are superior as compared to RFs. When using the whole image as an input for training the CNN, the results are worse leading to an MAE of 0.96 ± 0.77, presumably due to the reduced resolution and higher probability of overfitting to structures irrelevant to age estimation. Hence, to get the best performance, the CNN should be trained on the cropped regions containing the relevant age information.

Additionally, we evaluate on which image regions the CNN trained on the whole hand image is concentrating most. We visualize the network's focus by summing up the activations of the second convolution filter output after the third max pooling. To show that the network is focusing on different regions depending on the biological age of a subject, we group the visualization results by summing up values for all subjects within the same age group. Figure 8.7 shows the resulting visualization. For all age ranges, these results show that the network focuses on the same regions that we used when cropping the input volumes. Although this indicates that the CNN is able to identify the relevant structures on its own, the network using the whole volume as an input requires more training iterations for converging, more training time per iteration, and more memory. Moreover, the results are worse as compared to using the cropped regions as network input (see Table 8.1).



(a) 13-14 years **(b)** 14-15 years **(c)** 15-16 years

(d) 16-17 years **(e)** 17-18 years **(f)** $\geq$ 18 years

Figure 8.7: Visualizations of the hand image regions delivering the most responses for generating the age predictions of a CNN grouped by age ranges. Individual responses are deformed with a non-linear transformation that was calculated by transforming the landmarks of the corresponding input image to the same reference hand (as outlined in white). The individual responses of subjects within the same age range are merged by summing up and normalized for visualization. Images are taken from [135].

8.2.2.4 Multi-Factorial Age Estimation from MR Images

As MR images of hands show the physical maturation of a subject only up until the age of around 19 years, complementary anatomical information is needed to extend the predictable age range. To be able to predict the chronological age up until 25 years, we also incorporate the maturation information from wisdom teeth and clavicles. Thus, we set up our CNN pipeline to use also MR images of the jaw and the upper thorax. As the maturation information is concentrated on the wisdom teeth and the epiphyseal gaps of the clavicle bones, respectively, similar to the MR images of the hand, we crop the images to only contain these regions containing the maturation information.

Hand	Clavicles	Teeth	MAE chronological age Mean ± SD
✓	✓	✓	**1.01 ± 0.74**
✓			1.25 ± 0.96
	✓		1.22 ± 0.95
		✓	1.42 ± 1.14
✓	✓		1.04 ± 0.78
✓		✓	1.11 ± 0.95
	✓	✓	1.10 ± 0.87

Table 8.2: Results for CNNs trained on all combinations of the anatomical sites for four-fold cross-validation on images of 322 subjects in the age range of 13 to 25 years. The checkmarks on of hands, clavicles, and teeth indicate on which anatomical sites the CNN was trained. The MAE shows the mean and standard deviation of the absolute error of each individual age prediction to the ground-truth age of all subjects.

We perform four-fold cross-validation to predict the chronological age of all 322 patients on all combinations of the three anatomical sites hands, clavicles, and teeth. Due to the larger memory requirements using MR images of all three anatomical sites, we were not able to compare to CNNs using the whole MR images as input. Moreover, we do not compare to RFs, since for being competitive, they require image enhancement as preprocessing. Implementing an image enhancement for clavicles and wisdom teeth that is similar to the one used for hands would need a lot of work, while the RFs would probably still be beaten by the CNNs that do not need the image enhancement of the epiphyseal gaps (see Table 8.1). However, we do not consider this as a drawback, as the purpose of this experiment is to show the importance of combining maturation information from multiple sites for improving age estimation.

The results of the four-fold cross-validation, comparing the MAE of the ground-truth and predicted age on all combinations of hands (H), clavicles (C), and teeth (T) are shown in Table 8.2. From these results we can see that CNNs trained on hands and clavicles alone perform similarly well, resulting in an MAE of 1.25 ± 0.96 and 1.22 ± 0.95 years for the whole age range from 13 to 25 years. Note that as compared to using subjects with an age up to 18 only (see Section 8.2.2.3), the results of the CNN trained on the hand only for the whole age range are worse, indicating that the subjects from 18 to 25 years are more difficult to distinguish. The results for the CNN trained on teeth alone are worse with an MAE of 1.42 ± 1.14, due to less reliable information of the mineralization of the wisdom teeth as well as possibly completely missing wisdom teeth. When using CNNs trained on two sites, combining teeth with hands or clavicles improves the MAE already, while combining hands and clavicles gives the best MAE 1.04 ± 0.78 years. Finally, the CNN trained on all three anatomical sites gives the best results with an MAE of 1.01 ± 0.74 years, showing that the CNN incorporates information from all three sites to generate the age prediction.

Additionally, we investigate from which anatomical site the network uses most infor-

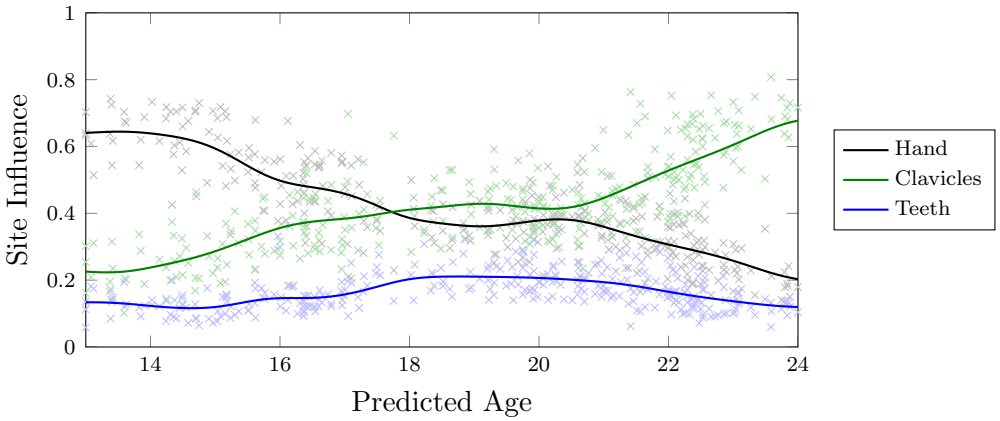


Figure 8.8: The influences of the individual anatomical sites for all subjects over the whole predicted age range. The individual influences per subject are normalized to one and shown as small × in the respective colors, while smoothed running averages of all subjects for the individual site influences are plotted as lines.

mation to predict a certain age. For this, we sum up the activations of the last fully connected layers of the feature extraction and encoding blocks of each anatomical site and calculate the ratio to the total sum of all three sites combined. The results of all subjects are plotted in Fig. 8.8. We can see that up until ≈ 18 years, most activations are coming from the hands, while after ≈ 21 years, most activations are coming from the clavicles. This confirms that with higher age the hand information becomes less reliable as less information is used, while the influence of the clavicles increases. The teeth provide the least activations over the whole predicted age range, indicating that they are the least reliable source of information. In conclusion, this experiment confirms that information from all three sites combined is required for a reliable age prediction of subjects between 13 and 25 years.

8.2.3 Conclusion

In this section, we have shown that CNNs may be used for fully automatic age estimation of living individuals from MR images of the hand, clavicles, and teeth. Furthermore, we have shown in several experiments, that the age-relevant information is concentrated only in specific regions of the images. Thus, localizing and cropping the relevant structures improve the results, showing that fully automatic age estimation is a clinically and forensically relevant application for anatomical landmark localization.

8.3 Multi-Label Whole Heart Segmentation

Cardiovascular diseases are the leading cause of death worldwide [94] with an estimated 17.9 million people who died in 2016, representing 31% of all global deaths [158]. Early diagnosis from CT or MR volumes of the heart plays a vital role in reducing the mortality and morbidity of cardiovascular diseases [67]. For quantifying morphological and pathological changes of the heart, it is highly relevant to segment and analyze the individual substructures of the heart, e.g., left and right ventricle, left and right atrium, myocardium, pulmonary artery, and the aorta. As manually segmenting the heart substructures is a laborious and time-consuming process that suffers from large intra- and inter-rater variability, fully automatic segmentation of the heart substructures is preferred [174].

Challenges for automatic heart substructure segmentation are the large anatomical variability in shape among subjects, the potential indistinctive boundaries between substructures, and, especially for MRI data, artifacts and intensity inhomogeneities resulting from the acquisition process. While earlier methods for heart substructure segmentation were often based on (multi-)atlases or deformable models (see [67, 112, 172] for extensive overviews of existing methods published before 2015), recently there has been a shift towards deep learning. However, most deep-learning-based methods focus only on segmenting one heart substructure at a time (e.g., [4, 69, 101, 138, 162]), and not on segmenting all individual heart substructures, i.e., Whole Heart Segmentation (WHS). Due to the lack of a publicly available dataset for WHS, the MICCAI 2017 Multi-Modality Whole Heart Segmentation (MM-WHS) challenge was organized to objectively compare the state-of-the-art in WHS.

Similar to anatomical landmark localization, in WHS the spatial configuration of the substructures is predetermined by physiology. As we have already shown that our SCN incorporates the *spatial configuration* of landmarks (see Chapters 5 and 7), we adapt the SCN for multi-label segmentation to also incorporate the *spatial configuration* of anatomical substructures for WHS.

To validate our SCN for WHS, we participated in the MM-WHS 2017 challenge. Evaluation on both CT and MR datasets of the MM-WHS 2017 challenge has shown that our proposed method overall outperforms all other participating methods, ranking first and winning the challenge.

The method and results of our method presented in this section are published in detail in [107], while [173] summarizes the MM-WHS 2017 challenge, comparing all participating methods and results.

8.3.1 Method

We perform fully automatic multi-label whole heart segmentation from CT and MR data with the following two-step approach (see Fig. 8.9). At first, due to the large variation of the field-of-view of the input volumes, a volumetric CNN with low input resolution roughly

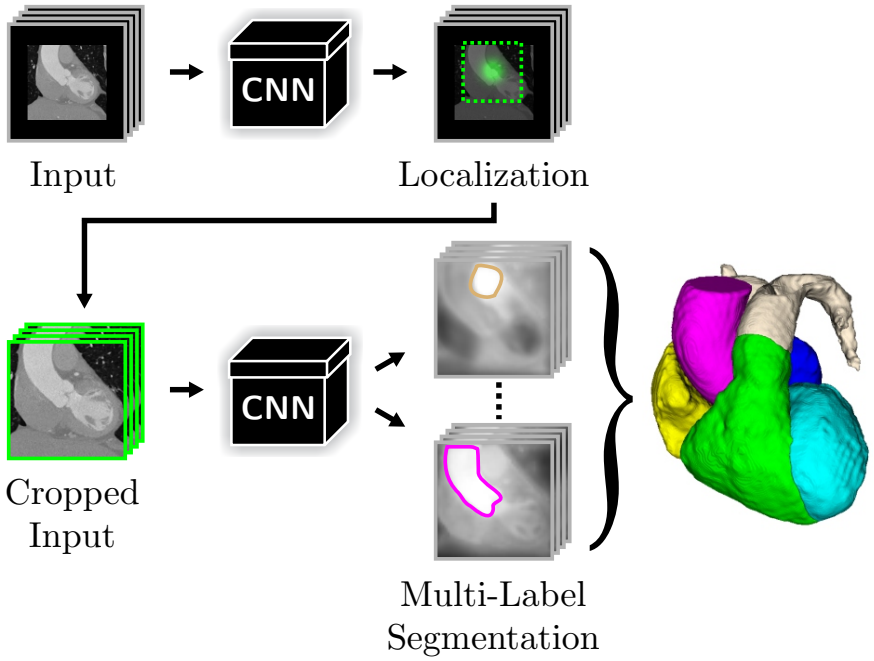


Figure 8.9: Overview of our fully automatic two-step multi-label whole heart segmentation pipeline. The first CNN uses a low-resolution volume as input to localize the center of the bounding box around all heart substructures. The second CNN uses a region cropped around this center and performs the multi-label segmentation.

localizes the center of the heart. Then, after cropping a region around the localized heart, another volumetric CNN with high input resolution performs multi-label segmentation of the individual heart substructures.

8.3.1.1 Heart Localization

Due to the varying field-of-view, the input volumes contain lots of background that is not needed for segmenting the individual heart substructures, while the heart is not necessarily in the center of the input volume. Therefore, to ensure that the heart is in the center as well as to remove the uninformative background from the input volume for the subsequent segmentation network, we localize the approximate center of the heart $\hat{\mathbf{x}}_{\text{heart}} \in \mathbb{R}^3$ and crop a region of fixed size around the center of the heart.

Before processing an input volume, it is resampled to a uniform voxel spacing of 10 mm for CT and 12 mm for MRI and centered at the network input. The network input resolution is $[32 \times 32 \times 32]$, which allows volumes with an extent of up to $[320 \times 320 \times 320]$ mm for CT and $[384 \times 384 \times 384]$ mm for MRI to fit into the network input. Although some volumes of the evaluated dataset have a larger extent, this extent of the network input volumes was sufficient to predict the center of the heart $\mathbf{x}_{\text{heart}}$ for all volumes of the evaluated datasets.

For localizing $\hat{\mathbf{x}}_{\text{heart}}$ we use a U-Net to perform heatmap regression (see Chapter 4).

For a 3-dimensional input image $I : \Omega_I \to \mathbb{R}$ with support $\Omega_I \subset \mathbb{R}^3$, the target heatmap $\overset{*}{H}_{\text{heart}} : \Omega_I \to \mathbb{R}$ (see Eq. (4.6)) is generated from the center $\hat{\mathbf{x}}_{\text{heart}}$ of the bounding box of the foreground label $\overset{*}{S}_{\text{heart}}$ of all heart substructures. The networks are trained with the L_2 loss of Eq. (4.7), where the size of the target Gaussian heatmap is not learned, but fixed with $\sigma = 1.5$.

Our variant of the U-Net is adapted such that it performs average instead of max pooling and linear upsampling instead of transposed convolutions. It uses four levels where each convolution layer has a kernel size of $[3 \times 3 \times 3]$ and a ReLU activation function. Starting from 32 outputs at the first level with the highest resolution, the number of outputs of each convolution layer at the same level is identical, while it is doubled at the next deeper level. We employ dropout of 0.5 after the convolutions of the contracting path in the deepest two levels. Furthermore, the convolution layers use zero padding such that the network input and output sizes stay the same. Taking the output of the U-Net as input, the predicted heatmap $\hat{H}_{\text{heart}}$ is generated with a final convolution layer with a kernel size of $[1 \times 1 \times 1]$ and a single output channel. The final predicted coordinate $\hat{\mathbf{x}}_{\text{heart}}$ is the coordinate, where $\hat{H}_{\text{heart}}$ has its maximum value.

8.3.1.2 Whole Heart Segmentation

As the original input volumes contain unnecessary background information, while the heart is not centered at the input, we resample the input volume, center it at the heart coordinate $\mathbf{x}_{\text{heart}}$, and crop a region of fixed size around this center. We resample the input volume to a uniform voxel spacing of 3 mm for CT and 4 mm for MRI. The network input volume size is $[64 \times 64 \times 64]$ voxels, which allows volumes of an extent of up to $[192 \times 192 \times 192]$ mm for CT and $[256 \times 256 \times 256]$ mm for MRI.

For segmenting the N individual heart substructures, we incorporate their *spatial configuration* into a CNN by adapting the SCN to perform multi-label segmentation. From the target heart label volume $\overset{*}{S}_{\text{heart}} : \Omega_I \to \{0 \dots N\}$, we generate the one-hot-encoded target substructures label volumes $\overset{*}{S}_i : \Omega_I \to \{0, 1\}$ with $i = 0 \dots N$. The SCN is set up to predict $N + 1$ label volumes $\hat{S}_i : \Omega_I \to (0, 1)$, i.e., each substructure plus the background.

We adapt the SCN as follows: In the *local appearance* part of the network uses four levels with two convolution layers before downsampling to the lower level, and two convolution layers after concatenating with the upsampled lower level. Starting from 64 filter outputs at the level with the highest resolution, the filter outputs are doubled at each level having a lower resolution. Each convolution layer has a kernel size of $[3 \times 3 \times 3]$ and uses a ReLU activation function. The final convolution layer of the *local appearance* part generates N outputs and has a kernel size of $[1 \times 1 \times 1]$. The N output volumes are activated with a sigmoid function to limit the values between 0 and 1, to represent the *local appearance* predictions $\hat{S}_i^{\text{LA}} : \Omega_I \to (0, 1)$.

The *spatial configuration* part of the network is processed in lower resolution as compared to the *local appearance* part. To do so, average pooling with a factor 4 per dimension

downsamples $\hat{S}_i^{\text{LA}}$, which serves as an input for the subsequent layers. Then, three consecutive convolution layers with 64 filter outputs and one final convolution layer with N outputs transform the *local appearance* predictions to the *spatial configuration* predictions. Each convolution layer has a kernel size of $[5 \times 5 \times 5]$ and uses a ReLU activation function. The N output volumes have a linear activation function and represent the *spatial configuration* predictions $\hat{S}_i^{\text{SC}} : \Omega_I \to \mathbb{R}$.

The final predicted output volumes $\hat{S}_i : \Omega_I \to \mathbb{R}$ are obtained by element-wise multiplication $\odot$ of the corresponding *local appearance* and *spatial configuration* outputs, i.e.,

$$\hat{S}_i = \hat{S}_i^{\text{LA}} \odot \hat{S}_i^{\text{SC}}. \tag{8.1}$$

We apply the softmax cross-entropy loss only on the final predicted output volume $\hat{S}_{\text{heart}} : \Omega_I \to \{0 \dots N\}$ to minimize the difference between target volume $\overset{*}{S}_i$ and predicted volume $\hat{S}_i$.

To generate the final multi-label segmentation, the predicted output volumes $\hat{S}_i(\mathbf{x})$ are resampled back to the original input resolution with tricubic interpolation. Then, for each voxel, the output label is set to the label i with the largest response in $\hat{S}_i$ with $i = 0 \dots N$, resulting in the final multi-label segmentation $\hat{S}_{\text{heart}} : \Omega_I \to \{0 \dots N\}$.

8.3.2 Evaluation

We evaluate our method for multi-label whole heart segmentation on the CT and MRI datasets of the MICCAI 2017 Multi-Modality Whole Heart Segmentation (MM-WHS) challenge. We perform three-fold cross-validation and compare the SCN with a U-Net on both training datasets, as well as show results of our SCN compared to other challenge participants on the test set of the challenge.

8.3.2.1 Dataset

We evaluated the networks on the datasets of the MM-WHS 2017 challenge. The organizers provided 20 CT and 20 MR volumes with corresponding manual segmentations of $N = 7$ heart substructures, i.e., left ventricle, left ventricle myocardium, right ventricle, left atrium, right atrium, ascending aorta, and pulmonary artery. The volumes were acquired in clinics with different scanners, resulting in varying image quality, resolution, and voxel spacing. The maximum physical size of the input volumes for CT is $300 \times 300 \times 188$ mm^3 while for MRI it is $400 \times 360 \times 400$ mm^3. The maximum size of the bounding box around the segmentation labels for CT is $155 \times 151 \times 160$ mm^3 (MRI: $180 \times 153 \times 209$ mm^3). For more details on scanners and acquisition protocols, we refer to [173].

To take part in the MM-WHS 2017 challenge, the participants had to submit predictions on 40 CT and 40 MR volumes provided by the organizers of the challenge. The organizers then evaluated the submissions by calculating the metrics on the hidden manual segmentations.

8.3.2.2 Implementation Details

We train and test the networks with our framework described in Chapter 6. The CT and MRI datasets are treated independently, i.e., every network is trained for either CT or MRI, but not both. We keep the network architectures for CT and MRI the same, while we use different input volume preprocessing due to the different intensity value ranges. We evaluate the localization and segmentation networks on three-fold cross-validation experiments, where we split the 20 annotated training volumes into three equally sized sets (13/13/14 training and 7/7/6 validation volumes), such that every training example is validated exactly once. To generate the final segmentation results for the 40 CT and 40 MRI volumes of the MM-WHS 2017 challenge test sets, we use networks trained on the 20 training volumes of CT and MRI, respectively. Network and data augmentation hyperparameter evaluation is performed on initial cross-validation experiments on the training set.

The U-Net for localization is set up as described in Section 8.3.1.1; the SCN for multi-label segmentation as described in Section 8.3.1.2. We additionally compare to a U-Net like architecture for multi-label segmentation, which is set up to be the same as the *local appearance* stage of the SCN. The biases of the networks are initialized with 0; the weights with the method as described in [51]. The weights of the layers generating $\hat{H}_{\text{heart}}$, $\hat{S}_i^{\text{LA}}$, and $\hat{S}_i^{\text{SC}}$ are sampled from a Gaussian distribution with a standard deviation of 0.001 to lead to initial values close to 0, which showed faster convergence.

We optimize the networks using Adam with learning rate 10^{-4} and the recommended default parameters from [71]. All networks are trained with a mini-batch size of 1 for 30,000 iterations (localization networks), and 50,000 iterations (segmentation networks). To reduce overfitting, training employs an L_2 weight regularization factor of $5 \cdot 10^{-4}$.

Due to the different orientations of the input volumes, we transform each volume into a common orientation. We preprocess the input volumes for the network as follows: The intensity values of the CT volumes are divided by $2,048$ and clamped between -1 and 1. Due to the different intensity value range for each MRI, we rescale each MR volume such that its output intensity values are between -1 and 1. To be more robust against intensity outliers, we use a robust maximum value, which we calculate as the 90^{th} percentile of all intensity values. During training, we additionally employ random data augmentations. The specific values for each transformation are sampled from a uniform distribution within the specified intervals. For intensity data augmentations, we shift the voxel values by $[-0.1, 0.1]$ and scale them by $[0.9, 1.1]$. For spatial data augmentations, we translate the volume by $[-10, 10]$ mm, scale them by $[0.8, 1.2]$, and rotate them by $[-10°, 10°]$ in each dimension. We additionally employ elastic deformations by moving points on a regular $8 \times 8 \times 8$ voxel grid randomly by up to 10 voxels and interpolating with 3^{rd} order B-splines.

All experiments were performed on an Intel Core i7-4820K based workstation with a 12 GB NVIDIA Geforce TitanX. Training took $\approx$ 1 h for the localization networks and $\approx$ 12 h

for the segmentation networks. To generate the multi-label segmentation results on the test dataset, the pipeline of preprocessing, inference of both networks, and postprocessing took in total ≈ 20 s for MRI and ≈ 1:45 m for CT.

8.3.3 Results

In the three-fold cross-validation, the localization network achieved a mean PE_{heart} of 13.2 mm with 5.4 mm standard deviation for CT, and 20.0 mm ± 30.5 mm for MRI, respectively. Despite the larger standard deviation for MRI, we observed that the accuracy is sufficient for detecting the approximate center of the heart used in the subsequent cropping of the input volume for the multi-label segmentation networks. For all evaluated images, the cropped region encloses the segmentation labels of all heart substructures.

	CT		MRI	
	U-Net	SCN	U-Net	SCN
LV	0.910 ± 0.043	**0.924** ± 0.033	0.811 ± 0.238	**0.877** ± 0.077
Myo	0.861 ± 0.042	**0.872** ± 0.039	0.681 ± 0.253	**0.752** ± 0.121
RV	**0.888** ± 0.039	0.879 ± 0.065	0.762 ± 0.249	**0.777** ± 0.195
LA	0.910 ± 0.052	**0.924** ± 0.036	0.740 ± 0.247	**0.811** ± 0.138
RA	0.865 ± 0.060	**0.878** ± 0.065	0.770 ± 0.221	**0.827** ± 0.158
aorta	**0.940** ± 0.062	0.911 ± 0.184	0.706 ± 0.202	**0.766** ± 0.138
PA	**0.837** ± 0.077	0.833 ± 0.091	0.687 ± 0.165	**0.720** ± 0.161
gDSC	0.891 ± 0.030	**0.895** ± 0.034	0.754 ± 0.225	**0.806** ± 0.114

Table 8.3: Segmentation results of three-fold cross-validation on the MM-WHS 2017 challenge training set, showing the Dice Similarity Coefficient (DSC) for U-Net and our SCN. The values show the mean ± SD of all images from the CT and MRI cross-validation setup for each segmentation label. Label abbreviations: LV - left ventricle blood cavity, Myo - myocardium of the left ventricle, RV - right ventricle blood cavity, LA - left atrium blood cavity, RA - right atrium blood cavity, aorta - ascending aorta, PA - pulmonary artery, μ - average of the seven whole heart substructures.

The segmentation metrics of the cross-validation for SCN and U-Net are shown in Table 8.3. On the CT dataset both U-Net and SCN perform similarly with a mean Generalized Dice Similarity Coefficient (gDSC) of 0.891 and 0.895, respectively. Most substructures are identified with an average DSC of more than 0.85, which shows the good performance of the segmentation CNNs for WHS. The only structure in CT with a DSC of less than 0.85 is the pulmonary artery. We suppose that the lower DSC value results from inconsistencies in the ground-truth segmentation in terms of how deep the pulmonary artery vessel tree is annotated.

On the MRI dataset the improvements from U-Net to SCN are more prominent with a mean gDSC increasing from 0.754 to 0.806. Despite the better results of the SCN, the predictions of the individual substructures have a lower mean DSC as compared to CT, due to inconsistencies in the ground-truth segmentation (e.g., aorta, pulmonary artery), as well as more variation in the anatomical field-of-view, intensity ranges and acquisition artifacts of the MRI data. We assume that the larger variability of MRI data would require more

annotated training data for the CNNs to achieve similar results as for CT. However, while the U-Net fails to recognize the substructures reliably on such a low amount of training data, the SCN with its *spatial configuration* stage compensates the lack of training data by focusing on anatomically feasible configurations.

Rank	Team	gDSC	$\mathcal{H}_{\text{all}}$ mean (in mm)	DL/MAS
1st	SCN	**0.908** ± 0.086	**25.242** ± 10.813	DL
2nd	Wang and Smedby [151]	0.894 ± 0.030	31.146 ± 13.203	DL
3rd	Yang et al. [163]	0.890 ± 0.049	29.006 ± 15.804	DL
4th	Yang et al. [164]	0.886 ± 0.047	41.974 ± 16.287	DL
5th	Mortazi et al. [96]	0.879 ± 0.079	28.481 ± 11.434	DL
6th	Tong et al. [143]	0.879 ± 0.023	34.129 ± 12.528	MAS
7th	Tong et al. [143]	0.849 ± 0.061	44.880 ± 16.084	DL
8th	Galisot et al. [37]	0.838 ± 0.152	34.634 ± 12.351	MAS

Table 8.4: Ranked results of the gDSC of all labels on the CT test sets of all MM-WHS 2017 challenge participants. The results of our approach are highlighted.

Rank	Team	gDSC	$\mathcal{H}_{\text{all}}$ mean (in mm)	DL/MAS
1st	Heinrich and Oster [54]	**0.870** ± 0.035	**28.535** ± 13.220	MAS
2nd	SCN	0.863 ± 0.043	30.227 ± 14.046	DL
3rd	Wang and Smedby [151]	0.855 ± 0.069	30.201 ± 13.216	DL
4th	Mortazi et al. [96]	0.818 ± 0.096	40.092 ± 21.119	DL
5th	Galisot et al. [37]	0.817 ± 0.059	30.938 ± 12.190	MAS
6th	Yang et al. [164]	0.810 ± 0.071	33.101 ± 13.804	DL
7th	Yang et al. [163]	0.783 ± 0.097	44.837 ± 15.658	DL
8th	Tong et al. [143]	0.674 ± 0.182	92.889 ± 18.001	DL

Table 8.5: Ranked results of the gDSC of all labels on the MRI test sets of all MM-WHS 2017 challenge participants. The results of our approach are highlighted.

For generating the segmentations on the test set, we trained the networks on all training images with the same hyperparameters as used for the cross-validation. We submitted the predicted segmentations to the challenge organizers, who evaluated the results on the hidden annotations of the test set. The ranked results of all participating teams of the MM-WHS 2017 challenge are taken from [173] and shown in Tables 8.4 and 8.5 for CT and MRI, respectively. By ranking first on the CT dataset and second on the MRI dataset, our method achieved the overall first place and won the MM-WHS 2017 challenge. When comparing the results on the test set with the cross-validation (Table 8.3), we can see that for both CT and MRI the mean gDSC improved. While for CT the mean gDSC improved from the already high baseline of 0.895 in cross-validation to 0.908 in the test set, in MRI the improvement is much more prominent from 0.806 to 0.863. The reason for the larger improvement in the MRI dataset is the larger variability of the input data, which is better covered with the larger amount of training data of the networks used to generate results for the test set. Although the SCN is already much better able to cope with a smaller amount of training data as compared to the U-Net (see Table 8.3), as generally anticipated

in deep learning, increasing the amount of training data improves the results also for the SCN (as already observed previously, e.g., Section 7.4).

8.3.4 Conclusion

In this section, we have shown how to adapt our SCN for multi-label segmentation. By at first localizing the position of the heart with heatmap regression, and then segmenting the individual substructures of the heart with the SCN incorporating the *spatial configuration* of anatomical structures, we were not only able to outperform the U-Net, but also all other participants of the MICCAI 2017 Multi-Modality Whole Heart Segmentation (MM-WHS) challenge. While the results and the adapted SCN still need more investigation, they already indicate a great potential of the SCN also for multi-label segmentation tasks.

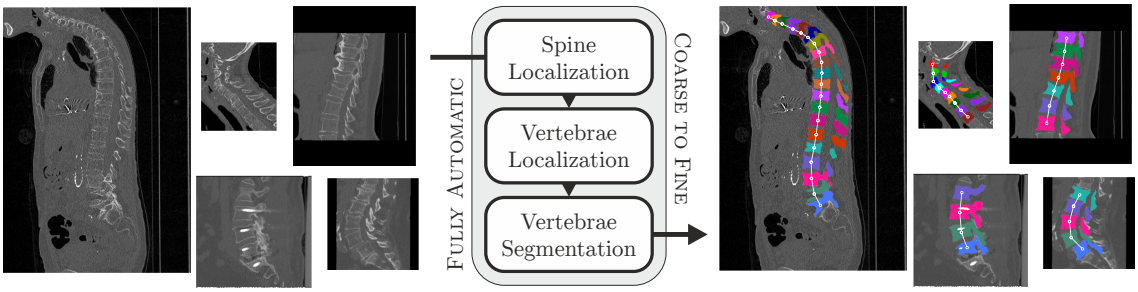


Figure 8.10: Overview of our proposed coarse-to-fine fully automatic vertebrae localization, identification, and segmentation. The three-step approach works for spine CT volumes having a large range of different field-of-views, as well as pathologies.

8.4 Vertebrae Segmentation

Localization and segmentation of vertebral bodies from spinal CT volumes is a crucial step for many clinical applications involving the spine, e.g., pathological diagnosis [36], surgical planning [75], and post-operative assessment [2]. Due to the highly repetitive structure of vertebrae, large variation in the appearance of different pathologies including fractures and implants, as well as different field-of-views, analyzing spinal CT volumes is challenging. To objectively compare methods for vertebrae localization and segmentation, Sekuboyina et al. [123] organized the MICCAI 2019 Large Scale Vertebrae Segmentation Challenge (VerSe 2019), involving real-world conditions with respect to image composition and pathologies. We participated in this challenge to objectively compare the performance of our proposed methods with the state-of-the-art for anatomical landmark localization and multi-label segmentation.

Differently to multi-label whole heart segmentation (see Section 8.3), where each already segmented heart substructure is straightforward to discriminate and identify, this is much more difficult for the individual vertebrae. Although there exist methods that use a single CNN for multi-label segmentation of the five lumbar vertebrae only [64, 124], they do not generalize well to all vertebrae of the whole spine. The reason for the bad generalization is that the multi-label CNN applied for the whole spine would not only need to accurately differentiate vertebral bodies from the background but also identify each vertebra. This could lead to erroneous predictions, where although the vertebral bodies are correctly differentiated from the background, multiple different foreground labels are present within the same vertebral body since the differentiation of individual vertebrae is difficult. However, if the vertebrae are already identified correctly, the segmentation CNN would only need to focus on distinguishing the vertebral bodies from the background, and not to identify the vertebrae label. Hence, similar to other methods in the literature, we separate multi-label vertebrae segmentation into the two tasks vertebrae localization and identification, and binary vertebrae segmentation.

For the task vertebrae localization and identification, [44] have introduced the MICCAI

CSI 2014 Vertebrae Localization and Identification Challenge dataset, which has been used as a benchmark for localizing and identifying vertebrae in spinal CT volumes (see 3DSpine dataset in Chapter 7). While earlier methods use RFs [44], or incorporate CNNs only for identification [12], recent methods perform heatmap regression to simultaneously localize and identify vertebrae [83, 123, 160]. As we have shown in Section 7.1.4 that our SCN outperforms other methods on the 3DSpine dataset, the SCN is our method of choice for vertebrae localization and identification.

For the task vertebrae segmentation, due to the specific shape of vertebrae, many methods incorporate models of their shape, e.g., statistical shape models [74], superquadric models [132], atlas-based models [154], and deformable surface models [76]. However, due to its high segmentation performance and easy setup, we use the U-Net for binary vertebrae segmentation.

We developed a coarse-to-fine fully automatic method for localization, identification, and segmentation of vertebrae in spine CT volumes, as presented in this section. We first roughly localize the spine, then localize and identify individual vertebrae, and finally segment each vertebra individually in high-resolution. In Section 8.4.2, we perform evaluation and comparison to state-of-the-art methods on the VerSe 2019 challenge. Our proposed method achieves top performance, ranking first place, and winning the VerSe 2019 challenge.

The method and results presented in this section were published in [109], while a summary of the VerSe 2019 challenge is published in [122].

8.4.1 Method

We perform vertebrae localization and segmentation in a three-step approach (see Fig. 8.10). Firstly, due to the large variation of the field-of-view of the input CT volumes, a CNN with a coarse input resolution predicts the approximate location of the spine. Secondly, another CNN in higher resolution performs multiple landmark localization and identification of the individual vertebra centroids. Lastly, the segmentation CNN in the highest resolution performs a binary segmentation of each localized vertebra. The results of the individually segmented vertebrae are then merged into the final multi-label segmentation.

8.4.1.1 Spine Localization

Due to the varying field-of-view, spine CT volumes often contain lots of background that does not contain useful information, while the spine may not be in the center of the volume. To ensure that the spine is centered at the input for the subsequent vertebrae localization step, as a first step, we predict the approximate x and y coordinates $\hat{\mathbf{x}}_{\text{spine}} \in \mathbb{R}^2$ of the spine. For localizing $\hat{\mathbf{x}}_{\text{spine}}$, we use a variant of the U-Net [117] to perform heatmap regression [106, 142] of the spinal centerline, i.e., the line passing through all vertebral centroids. For a 3-dimensional input image $I : \Omega_I \rightarrow \mathbb{R}$ with support $\Omega_I \subset \mathbb{R}^3$, the

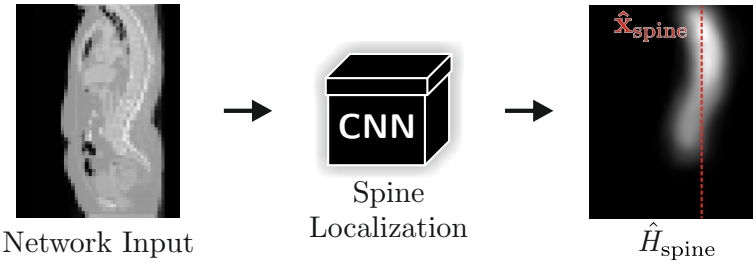


Figure 8.11: Input volume and the predicted spine heatmap $\hat{H}_{\text{spine}}$ of the spine localization network. The predicted coordinate $\hat{\mathbf{x}}_{\text{spine}}$ is the x and y coordinate of the center of mass of $\hat{H}_{\text{spine}}$.

target heatmap volume $\overset{*}{H}_{\text{spine}} : \Omega_I \rightarrow \mathbb{R}$ of the spinal centerline is generated by merging Gaussian heatmaps with size σ_{spine} of all individual vertebrae target coordinates $\overset{*}{\mathbf{x}}_i$ into a single volume (see Fig. 8.11). We use the L_2 loss to minimize the difference between the target heatmap volume $\overset{*}{H}_{\text{spine}}$ and the predicted heatmap volume $\hat{H}_{\text{spine}} : \Omega_I \rightarrow \mathbb{R}$. The final predicted spine coordinate $\hat{\mathbf{x}}_{\text{spine}}$ is the x and y coordinate of the center of mass of $\hat{H}_{\text{spine}}$.

Our variant of the U-Net is adapted such that it performs average instead of max pooling and linear upsampling instead of transposed convolutions. It uses five levels where each convolution layer has a kernel size of $[3 \times 3 \times 3]$ and 64 filter outputs. Furthermore, the convolution layers use zero padding such that the network input and output sizes stay the same.

Before processing a spine CT volume, it is resampled to a uniform voxel spacing of 8 mm and centered at the network input. The network input resolution is $[64 \times 64 \times 128]$, which allows spine CT volumes with an extent of up to $[512 \times 512 \times 1024]$ mm to fit into the network input. This extent was sufficient for the network to predict $\mathbf{x}_{\text{spine}}$ for all spine CT volumes of the evaluated dataset.

8.4.1.2 Vertebrae Localization

To localize centers of the vertebral bodies, we use the SCN (see Chapter 5). The network effectively combines the *local appearance* of landmarks with their *spatial configuration*. The *local appearance* part of the network uses five levels consisting of two convolution layers before downsampling to the lower level, and two convolution layers after concatenating with the upsampled lower level. Each convolution layer uses a leaky ReLU activation function and has a kernel size of $[3 \times 3 \times 3]$ and 64 filter outputs. The *spatial configuration* part consists of four convolutions with $[7 \times 7 \times 7]$ kernels in a row and is processed in $1/4^{\text{th}}$ of the resolution of the *local appearance* part.

The SCN performs heatmap regression of the N target vertebrae $\mathcal{L}_i$ with $i = 1 \ldots N$, i.e., each target coordinate $\overset{*}{\mathbf{x}}_i$ is represented as a Gaussian heatmap volume $\overset{*}{H}_i : \Omega_I \rightarrow \mathbb{R}$ centered at $\overset{*}{\mathbf{x}}_i$. For N target vertebra landmark $\mathcal{L}_i$, the network predicts simultaneously all N output heatmap volumes $\hat{H}_i : \Omega_I \rightarrow \mathbb{R}$. As loss function, we use our proposed

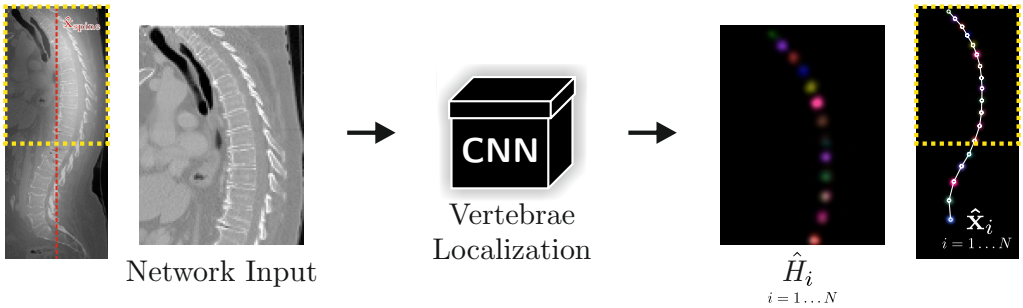


Figure 8.12: Input volume and individual heatmap predictions of the vertebrae localization network. The yellow rectangle indicates that not the whole input volume is processed at once, but overlapping cropped sub-volumes that are centered at $\hat{\mathbf{x}}_{\text{spine}}$. The network predicts simultaneously N heatmaps, i.e., a single heatmap $\hat{H}_i$ for each individual vertebrae $\mathcal{L}_i$. For visualization, the predicted heatmaps are colored individually and combined into a single image. The final landmark coordinates $\hat{\mathbf{x}}_i$ are identified as the longest sequence of local maxima of $\hat{\mathbf{x}}_i$ that does not violate anatomical constraints.

modified L_2 loss function of Eq. (4.8), which also allows learning of the individual σ_i values for the target Gaussian heatmap volumes $\overset{*}{H}_i$.

A schematic representation of how the input volumes are processed to predict all heatmaps $\hat{H}_i$ is shown in Fig. 8.12. Each network input volume is resampled to have a uniform voxel spacing of 2 mm, while the network is set up for inputs of size $[96 \times 96 \times 128]$, which allows volumes with an extent of $[192 \times 192 \times 256]$ mm to fit into the network. With this extent, many images of the dataset do not fit into the network and cannot be processed at once. To narrow the processed volume to the approximate location of the spine, we center the network input at the predicted spine coordinate $\hat{\mathbf{x}}_{\text{spine}}$ (see Section 8.4.1.1). Furthermore, as some spine CT volumes have a larger extent in the z-axis (i.e., the axis perpendicular to the axial plane) that would not fit into the network, we process such volumes the same way as we did with the 3DSpine dataset in Section 7.1.4. During training, we crop a subvolume at a random position at the z-axis. During inference, we split the volumes at the z-axis into multiple subvolumes that overlap for 96 pixels, and process them one after another. Then, we merge the network predictions of the overlapping subvolumes by taking the maximum response over all predictions.

We predict the final landmark coordinates $\hat{\mathbf{x}}$ as follows: For each predicted heatmap volume, we detect multiple local heatmap maxima that are above a certain threshold. Then, we determine the first and last vertebrae that are visible on the volume by taking the heatmap with the largest value that is closest to the volume top or bottom, respectively. We identify the final predicted landmark sequence by taking the sequence that does not violate the following conditions: consecutive vertebrae may not be closer than 12.5 mm and farther away than 50 mm, as well as a subsequent landmark may not be above a previous one.

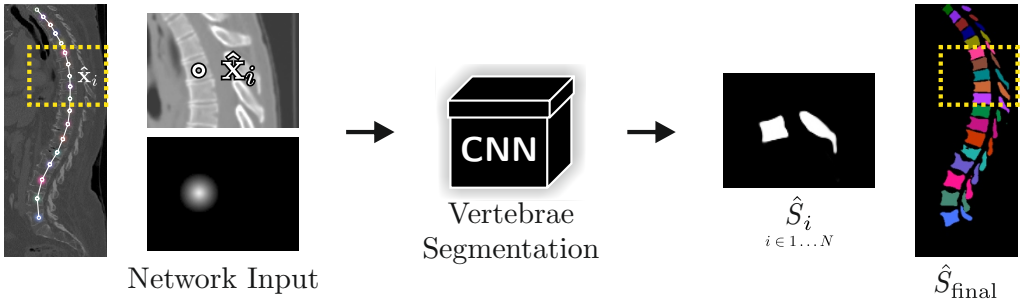


Figure 8.13: Input volume and segmented vertebrae of the spine segmentation network. The yellow rectangle shows the cropped region around a single vertebrae $\mathcal{L}_i$ and indicates that each localized vertebrae $\hat{\mathbf{x}}_i$ is processed individually. Each individual vertebra sigmoid prediction $\hat{S}_i$ is then transformed and resampled back to the original position. The final multi-label segmentation $\hat{S}_{\text{final}}$ is obtained by setting the label at each voxel to the label of $\hat{S}_i$ that has the largest response.

8.4.1.3 Vertebrae Segmentation

For creating the final vertebrae segmentation, we use a U-Net [117] set up for binary segmentation to separate a vertebra from the background (see Fig. 8.13). The final semantic label of a vertebra is identified through the localization label as predicted by the vertebrae localization network (see Section 8.4.1.2). Hence, we use a single network for all vertebrae landmarks $\mathcal{L}_i$, as the network does not need to identify, which vertebra it is segmenting, but it only needs to separate each vertebra individually from the background.

Since each vertebra is segmented independently, the CNN needs to know, which vertebra it should segment in the input volume. Thus, from the whole spine CT image, we crop the region around the localized vertebra, such that the vertebra is in the center of the cropped image. During training, we use the ground-truth vertebra location $\overset{*}{\mathbf{x}}_i$, while during inference, we use the predicted vertebra coordinate $\hat{\mathbf{x}}_i$. Additionally, we create an image of a Gaussian heatmap centered at the vertebra coordinate $\hat{\mathbf{x}}_i$. Both the cropped and the heatmap image are used as an input for the segmentation U-Net. The U-Net is modified as described in Section 8.4.1.1. It is set up to predict a single output volume $\hat{S}_i : \Omega_I \rightarrow (0, 1)$, while the sigmoid cross-entropy loss is minimized to generate predictions close to the target binary label volume $\overset{*}{S}_i : \Omega_I \rightarrow \{0, 1\}$. The input volumes are resampled to have a uniform voxel spacing of 1 mm, while the network is set up for inputs of size $[128 \times 128 \times 96]$, which allows volumes with an extent of $[128 \times 128 \times 96]$ mm.

To create the final multi-label segmentation result, the individual predictions of the cropped vertebra inputs need to be merged. Therefore, the sigmoid output volumes $\hat{S}_i$ of each cropped vertebrae i are transformed and resampled back to their position in the original input volume. Then, for each voxel in the final label image $\hat{S}_{\text{final}} : \Omega_I \rightarrow \{0 \dots N\}$, the predicted label is set to the label i of the vertebra that has the largest sigmoid response. If for a pixel no vertebra prediction $\hat{S}_i$ has a response > 0.5, the pixel is set to be the background.

8.4.2 Evaluation

We evaluate our proposed framework for multi-label spine localization and segmentation on the dataset of the VerSe 2019 challenge. We perform three-fold cross-validation on the training set, where we evaluate the performance of our approach for individual vertebra types. Then, we show the results and comparisons to other methods on the test sets of the challenge.

8.4.2.1 Dataset

The dataset consists of spine CT volumes of subjects with various pathologies, where every fully visible vertebra from C1 to L5 is annotated. As some subjects contain the additional vertebra L6, at maximum $N = 25$ vertebrae are annotated. The training set consists of 80 spinal CT volumes with corresponding ground-truth centroids $\overset{*}{\mathbf{x}}_i$ and segmentations $\overset{*}{S}_i$ for each vertebra $\mathcal{L}_i$.

The VerSe 2019 challenge contains two test sets. The first test set consists of 40 publicly available spine CT volumes with hidden annotations. The participants of the challenge had to submit the predictions on the first test set to the evaluation servers, which did in turn evaluate and rank the submitted results on a public leaderboard. The second test set consists of an additional 40 hidden spine CT volumes. In order to obtain evaluation results on the second test set, the challenge participants had to submit a Docker[1] image of the proposed method that creates the predictions. The organizers of the challenge then performed an internal evaluation on the hidden second test set. The final rank of each participant of the VerSe 2019 challenge is defined by the performance on the 80 CT volumes of both test sets and was announced at the workshop on MICCAI 2019.

8.4.2.2 Implementation Details

Training and testing of the network were done with the framework described in Chapter 6. We evaluate the spine localization, vertebrae localization, and vertebrae segmentation networks individually on three-fold cross-validation experiments, where we split the 80 annotated training volumes into three equally sized sets (53/53/54 training and 27/27/26 validation volumes), such that every training example is validated exactly once. To generate the final vertebrae localization and segmentation results on the test sets, we use networks trained on all 80 training volumes. We performed network and data augmentation hyperparameter evaluation on initial cross-validation experiments on the training dataset. All networks are trained with a mini-batch size of 1, while the spine localization network is trained for 20,000 iterations, the vertebrae localization network for 100,000 iterations, and the vertebrae segmentation network for 50,000 iterations. For the U-Net we use the Adam optimizer [71] with a learning rate of 10^{-4}, for the SCN we use the Nesterov [98] optimizer with a learning rate of 10^{-8}. The spine and vertebrae localization

[1] `https://www.docker.com/`

networks use L_2 weight regularization factor of $5 \cdot 10^{-4}$, the vertebrae segmentation network uses a factor of 10^{-7}. We set $\sigma_{\text{spine}} = 3$ pixel for the spine localization network; We set $\alpha = 100$ in Eq. (4.8) for learning the size σ_i of the target heatmaps $\overset{*}{H}_i$ in the vertebrae localization network.

Due to the different orientations, each CT volume is transformed into a common orientation for later processing. Furthermore, to reduce noise on the input volumes, they are smoothed with a Gaussian kernel with $\sigma = 0.75$ mm. To obtain an appropriate range of intensity values for neural networks, each intensity value of the CT volumes is divided by 2,048 and clamped between -1 and 1. For data augmentation during training, the intensity values are multiplied randomly with $[0.75, 1.25]$ and shifted by $[-0.25, 0.25]$. The images are randomly translated by $[-30, 30]$ voxels, rotated by $[-15°, 15°]$, and scaled by $[-0.85, 1.15]$. We additionally employ elastic deformations by randomly moving points on a regular 6×6 pixel grid by 15 pixels and interpolating with 3rd order B-splines. All augmentation operations sample randomly from a uniform distribution within the specified intervals.

Training took $\approx$ 3:30 h for the spine localization network, $\approx$ 28:00 h for the vertebrae localization network, and $\approx$ 12:00 h for the vertebrae segmentation network, on an Intel Core i7-4820K workstation with an NVIDIA Titan V running Arch Linux. The inference time is dependent on the field-of-view and the number of visible vertebrae on the input CT volume. On the 40 volumes of the test 1 set of the VerSe 2019 challenge, inference per volume takes on average $\approx$ 4:20 m, divided into $\approx$ 5 s for spine localization, $\approx$ 45 s for vertebrae localization, and $\approx$ 3:30 m for vertebrae segmentation.

8.4.2.3 Results

We evaluated our proposed framework on the MICCAI VerSe 2019 Grand Challenge. We performed three-fold cross-validation on the publicly available training set consisting of 80 annotated volumes to evaluate the individual steps of our proposed approach, i.e., spine localization, vertebrae localization, and vertebrae segmentation. As the purpose of this cross-validation is to show the performance of the individual steps, instead of using the predictions of the previous steps as inputs (i.e., $\hat{\mathbf{x}}_{\text{spine}}$ for vertebrae localization, and $\hat{\mathbf{x}}_i$ for vertebrae segmentation), the networks use the ground-truth annotations as inputs (i.e., $\overset{*}{\mathbf{x}}_{\text{spine}}$ for vertebrae localization, and $\overset{*}{\mathbf{x}}_i$ for vertebrae segmentation).

The cross-validation experiments on the 80 annotated training volumes confirm the good performance of the individual steps of our proposed three-step approach.

The first stage, the spine localization, performs well in approximating the center position of the spine, achieving a point error PE_{spine} of 4.13 ± 8.97 mm. Visual inspection showed only one failure case for a CT volume that is completely out of the training data distribution. This volume does not show the spine, but the whole legs. Only in the top region of the volume a small part of the spine is visible, specifically the two vertebrae L4 and L5.

Vertebrae $\mathcal{L}_i$	PE_i (in mm) mean ± SD	ID_i % (#correct of #total)	DSC_i mean ± SD	$\mathcal{H}_i$ (in mm) mean ± SD
Cervical (C1 . . . C7)	7.45 ± 8.70	91.07% (102 of 112)	0.91 ± 0.10	5.88 ± 9.50
Thoracic (T1 . . . T12)	5.56 ± 6.31	88.99% (388 of 436)	0.93 ± 0.14	6.78 ± 16.67
Lumbar (L1 . . . L6)	4.48 ± 2.08	90.45% (284 of 314)	0.96 ± 0.02	6.41 ± 9.05
All (i = C1 . . . L6)	5.71 ± 6.28	89.79% (774 of 862)	0.94 ± 0.11	6.53 ± 13.49

Table 8.6: Results of a three-fold cross-validation on the VerSe 2019 challenge training set consisting of 80 volumes. The table shows results grouped by cervical, thoracic, and lumbar vertebrae, as well as results for all vertebrae combined.

The second stage, the vertebrae localization, achieves a mean point error PE_{all} of 5.71 mm and an identification rate ID_{all} of 89.79% for all vertebrae (see Table 8.6). By analyzing the individual predictions for cervical, thoracic, and lumbar vertebrae, we see differences among the vertebrae types. As the thoracic vertebrae are in the middle, being far away from the visible top or bottom of the spine, it is harder for the network to distinguish between these vertebrae. This can be seen in the smaller $\text{ID}_{\text{thoracic}}$ of 88.99%, as compared to $\text{ID}_{\text{cervical}} = 91.07\%$ and $\text{ID}_{\text{lumbar}} = 90.45\%$. However, having more training data of individual vertebrae helps the networks predicting the vertebral centroids more accurately, which can be seen at the smaller $\text{PE}_{\text{lumbar}}$ of 4.48 mm (on average $\approx$ 62 annotations per vertebrae) as compared to $\text{PE}_{\text{thoracic}} = 5.56$ mm ($\approx$ 36 annotations) and $\text{PE}_{\text{lumbar}} = 7.45$ mm ($\approx$ 16 annotations).

Having more annotations per vertebrae is also beneficial for the final third stage, the vertebrae segmentation (see Table 8.6). Here we can observe that again the lumbar vertebrae have the best performance in terms of Dice similarity coefficient $\text{DSC}_{\text{lumbar}} = 0.96$, while the DSC decreases with less training data per vertebrae type, i.e., $\text{DSC}_{\text{thoracic}} = 0.93$ and $\text{DSC}_{\text{thoracic}} = 0.91$. However, for the Hausdorff metric $\mathcal{H}$ we do not see noteworthy differences among the vertebrae types. Moreover, the standard deviations of $\mathcal{H}$ are large, indicating outliers. We think that this is due to noise in the ground-truth annotation, sometimes containing spuriously annotated voxels far off the actual vertebrae region. Such misannotated isolated pixels are negligible in the DSC, but lead to large errors and standard deviations in the Hausdorff metric.

We participated in the VerSe 2019 challenge at MICCAI 2019 to evaluate our whole fully automatic approach and compare the performance to other methods. For this, we trained the three individual networks for spine localization, vertebrae localization, and vertebrae segmentation on all 80 training images. We performed inference on the test volumes by using the predictions from the previous step as inputs for the next step. We submitted our predictions on the test 1 set, as well as a Docker image for the organizers to generate predictions on the hidden test 2 set. Table 8.7 shows the quantitative results on the test 1 and test 2 sets of methods that submitted valid predictions before the deadlines of the VerSe 2019 challenge as announced at the challenge workshop at MICCAI 2019 [122].

Rank	Team	Score	test 1 set				test 2 set			
			ID_{all}	PE_{all}	DSC_{all}	$\mathcal{H}_{all}$	ID_{all}	PE_{all}	DSC_{all}	$\mathcal{H}_{all}$
1st	christian_payer	**0.691**	95.65	**4.27**	0.909	**6.35**	**94.25**	**4.80**	**0.898**	**7.34**
2nd	iFLYTEK	0.597	**96.94**	4.43	**0.930**	6.39	86.73	7.13	0.837	11.67
3rd	nlessmann	0.496	89.86	14.12	0.851	8.58	90.42	7.04	0.858	9.01
4th	huyujin	0.279	–	–	0.847	12.79	–	–	0.818	29.44
5th	yangd05	0.216	62.56	18.52	0.767	14.09	67.21	15.82	0.671	28.76
6th	ZIB	0.215	71.63	11.09	0.670	17.35	73.32	13.61	0.690	19.25
7th	AlibabaDAMO	0.140	89.82	7.39	0.827	11.22	–	–	–	–
8th	christoph	0.107	55.80	44.92	0.431	44.27	54.85	19.83	0.464	42.85
9th	INIT	0.084	84.02	12.40	0.719	24.59	–	–	–	–
10th	brown	0.022	–	–	0.627	35.90	–	–	–	–
11th	LRDE	0.007	0.01	205.41	0.140	77.48	0.00	1000.00	0.356	64.52

Table 8.7: Results on the overall VerSe 2019 challenge test set, which is comprised of 40 volumes in test 1 set and 40 volumes in test 2 set. The table lists all methods that submitted valid localizations or segmentations, which allowed the organizers to calculate the evaluated metrics. The predictions for vertebrae localization and segmentation of test 1 set were generated and submitted by the participants, while the predictions of test 2 set were generated by the organizers with the submitted Docker images. The methods are ranked as described in the VerSe 2019 challenge report [122]. The metrics show the mean values for all vertebrae of test 1 and test 2 set, respectively. Entries with a "–" indicate failure of metric calculation, because of erroneous or missing predictions, or missing Docker images.

Our fully automatic approach ranked first on the combined localization and segmentation metrics on the overall 80 volumes of both test sets.

When compared with the results of the cross-validation, the localization results improved on both test sets, as can be seen in both $PE_{all} = 4.27$ mm and $PE_{all} = 4.80$ mm, as well as $ID_{all} = 95.65\%$ and $ID_{all} = 94.25\%$. This indicates that the localization network benefits from more training data (80 CT volumes in the test sets as compared to ≈ 54 in the cross-validation), especially due to the large variation and different pathologies in the dataset.

For the segmentation metrics, the results on the test sets are slightly worse as compared to the cross-validation, i.e., $DSC_{all} = 0.909$ and $DSC_{all} = 0.898$, as well as $\mathcal{H}_{all} = 6.35$ mm and $\mathcal{H}_{all} = 7.34$ mm. The reason for this performance drop is that the vertebrae segmentation is dependent on the vertebrae localization. In contrast to the cross-validation, which uses the ground-truth vertebral centroids $\overset{*}{\mathbf{x}}_i$ as input to show the performance of the segmentation network alone, the segmentation network that generated results on the test sets takes the predicted vertebral centroids $\hat{\mathbf{x}}_i$ as input to show the performance of the whole fully automatic approach.

When compared to other methods on both test sets, our method achieves the overall best performance. There exists a large gap between our method and the next best ranking methods in both localization and segmentation performance. However, when looking at the individual test sets, we can see that in test 1 set the second-best method has a better

ID_{all} and DSC_{all} as compared to our method, while our method has a better PE_{all} and $\mathcal{H}_{\text{all}}$. Nevertheless, in test 2 set the second-best method has a performance drop in all evaluation metrics, while the results from our method are stable. The better performance on the hidden test 2 set shows good generalization capabilities of our method, enabling it to surpass all other methods and to win the VerSe 2019 challenge.

8.4.3 Conclusion

In this section, we have presented a three-step fully automatic approach that performs vertebrae localization and segmentation in a coarse-to-fine manner. By combining the SCN for vertebrae localization and identification with the U-Net for vertebrae segmentation, our method has achieved top performance in the dataset of the VerSe 2019 challenge. The good generalization of our method to the hidden test 2 set of the challenge has enabled our method to rank first and to win the challenge overall, showing another powerful application of our proposed SCN.

8.5 Summary

In this chapter, we have presented various applications of our proposed methods for landmark localization. We have shown that our proposed SCN achieves state-of-the-art results on two datasets of the computer vision domain, i.e., facepoint detection and human pose estimation. Furthermore, we have shown that our landmark localization method can be used to improve results for fully automatic age estimation from MR images. By letting the CNNs for age estimation focus on previously located landmarks in regions containing age relevant structures, results improve as compared to CNNs that observe the whole images also containing information irrelevant for age estimation. We also adapted the SCN for multi-label segmentation. On datasets for whole heart segmentation, the adapted SCN outperforms other methods, including the U-Net, while achieving top performance and ranking first on the MM-WHS 2017 challenge. As a last application, we have shown how to use our SCN in a coarse-to-fine pipeline for vertebrae localization, identification, and segmentation. Comparisons to other methods have shown that our proposed pipeline also ranks first on the VerSe 2019 challenge. In conclusion, we have shown several applications and adaptions of our proposed method for landmark localization, where our method achieves state-of-the-art results, often outperforming other methods.

9 Discussion and Conclusion

Contents

9.1 Discussion 147
9.2 Limitations and Future Work 152
9.3 Conclusion 154

That'll do, pig. That'll do.

Arthur H. Hoggett

In this chapter, we will give a concluding discussion of our proposed method in Section 9.1. Furthermore, we show certain limitations that we identified and possible directions for future work in Section 9.2, before finishing with final remarks in Section 9.3.

9.1 Discussion

In this thesis, we have shown that by incorporating *spatial configuration* into a Convolutional Neural Network (CNN) based deep learning architecture, we achieve high performance even with limited amounts of training data. We have evaluated localization performance of our proposed CNN on four size-limited datasets for anatomical landmark localization that contain 2D radiographs and 3D Magnetic Resonance Imaging (MRI) and Computed Tomography (CT) volumes of different anatomical structures (see Chapter 7). Furthermore, we have shown several applications of our CNN architecture, i.e., face point detection and human pose estimation, fully automatic age estimation from living individuals, multi-modal whole heart segmentation, and vertebrae localization and segmentation (see Chapter 8). In these experiments, most of our deep CNNs outperform other methods, demonstrating not only the generic applicability but also the superior performance of our proposed CNNs.

9.1.1 Landmark Localization with CNNs Using Heatmap Regression

Motivated by the recent success of deep network architectures in several tasks and domains (Chapter 2), we applied CNNs for landmark localization (Chapter 3). By using CNNs that do not directly regress the landmark coordinates (Section 4.2), but images of heatmaps, the CNNs do not need to learn a highly non-linear mapping from image to coordinate data, but from image to image data (Section 4.3). Experiments on the 2DHand dataset (Section 7.1.1), as well as experiments on the FLIC dataset for human pose estimation (Section 8.1) show the better performance of heatmap regression CNNs as compared to coordinate regression CNNs.

Additionally to achieving better performance, regressing heatmaps has several other properties and advantages. Since a heatmap represents an image of a pseudo probability of a landmark being located at a certain position, the CNNs need to learn to generate responses on locations near the target landmark, while locations far away from the target coordinate need to be suppressed. Thus, a target heatmap needs to be defined to have large responses on locations near the target coordinate, while rapidly decreasing towards zero when farther away. Similar to other methods, we represent target heatmaps with d-dimensional Gaussian functions [100, 114, 142], while the network is set up to learn to predict heatmaps as close as possible to the target heatmaps.

A critical hyperparameter for heatmap regression is the peak width $\boldsymbol{\sigma}$ of the target heatmaps. If the target heatmap is too large, the predicted heatmap will be inaccurate; if it is too small, the predicted heatmaps may have multiple peaks near the correct location (Fig. 4.4), or no peaks at all. However, due to the often large number of landmarks, evaluating and defining the optimal heatmap peak width for each landmark individually is infeasible. Thus, we have proposed to modify the network loss function to also allow learning of the optimal heatmap peak widths $\boldsymbol{\sigma}$ for each landmark individually. Our experiments in Section 7.2 have shown increased performance when learning the optimal $\boldsymbol{\sigma}$ for each landmark, while also removing the requirement of manually finetuning this hyperparameter for each landmark. When analyzing the results in more detail, we additionally observed that landmarks that are more difficult to annotate unambiguously, e.g., landmarks on edges instead of sharp corners, lead to larger optimal $\boldsymbol{\sigma}$. Furthermore, we saw a correlation of the expected localization error per landmark to its learned $\boldsymbol{\sigma}$. Thus, the individual $\boldsymbol{\sigma}$ encode an uncertainty of the landmark annotation on a dataset level.

Additionally to the aforementioned advantages, heatmap regression also enables better visualization and interpretation of the network's predictions, as the predicted heatmaps directly show where the network expects the location of the landmark (Section 7.3). We exploited this property by transforming intermediate local heatmap responses to positions of other landmarks to enable learning a graphical model of the landmarks within a fully-convolutional [88, 117, 126], end-to-end trainable network architecture, i.e., our proposed SpatialConfiguration-Net (SCN).

9.1.2 SpatialConfiguration-Net for Anatomical Landmark Localization

Inspired by the use of prior knowledge to restrict landmark configurations to anatomically feasible ones, in our proposed SCN, constraints on the relative position of landmarks are automatically learned from training data and integrated inside its *spatial configuration* component (Chapter 5). Previous state-of-the-art approaches in medical image analysis that were not based on CNN [28, 85] introduce appearance features and handcrafted models resembling anatomical constraints in separate components. Instead of using a handcrafted model, in [147] our research group has shown a possibility of learning the automatic integration of appearance information and geometric configuration into a single Random Forests (RFs) framework for localization. Thus, differently to previous methods that require the two separate components to be trained sequentially without any interaction between them, in our end-to-end trained SCN, the *local appearance* and *spatial configuration* components are simultaneously optimized, together providing both high robustness towards landmark misidentification as well as high accuracy locally at each identified landmark.

By simultaneously optimizing the two components, the problem of landmark localization is separated into two simpler sub-problems that can be modeled from a low amount of training data, as shown in our experiments (**??**). Such a simplification is only possible when the regression objective, which is calculated by multiplying the output of the two network components, is optimized in a single process and in an end-to-end manner. This is different from the methods of Tompson et al. [142] and Pfister et al. [114], where each of their network components has to be locally accurate and robust towards misidentification simultaneously. It is important to notice that when the multiplication given in Eq. (5.3) is replaced by additional convolutional layers, as Pfister et al. [114] used for human pose estimation in videos, this does not lead to the simplification of the localization problem, thus the need for large amounts of training data remains. Furthermore, our proposed network architecture learns to dedicate the *local appearance* component to locally accurate candidate predictions, without the need to distinguish locally similar structures, while the *spatial configuration* component solely focuses on eliminating ambiguities to improve robustness towards landmark misidentification, without the need for being locally accurate (see Figs. 7.14 to 7.17). Although there is no theoretical guarantee that the optimization process will lead to such a separation, we have observed this behavior in all our experiments, and we have shown it in more detail in Section 7.3.

We also adapted several other fully-convolutional network architectures inspired by literature for anatomical landmark localization. However, as seen in results for the 2DHand dataset (Section 7.1.1), the other architectures cannot compete with our proposed SCN. The architecture that comes closest to the performance of the SCN is our reimplementation of the fully convolutional U-Net [117]. After tuning it for heatmap regression, our U-Net achieved competitive localization performance in the experiments, often being outperformed only by our proposed SCN. However, when evaluated only regarding robustness

towards landmark misidentification, the U-Net shows limitations. This can be seen especially in the 3DSpine experiment with its large anatomical and pathological variation, where our SCN is 8.5% better in the ID_{rate} evaluation (Section 7.1.4). We think that this drop in performance is due to the multi-scale U-Net architecture not using the prior knowledge that landmarks are not uniformly distributed in image space but are constrained by other anatomical landmarks. Thus, without prior knowledge of the existence of such constraints on the *spatial configuration*, the U-Net requires a large amount of training data to learn these constraints on multiple scales solely from the data. This can best be seen from the 2DHandReduced experiment (Section 7.4), where our proposed SCN benefits from this prior knowledge when learning the same anatomical variation from an extremely reduced amount of training data.

When comparing our SCN to other state-of-the-art approaches, in our two in-house datasets 2DHand and 3DHand we outperform all our previously reported results [31, 130, 147] based on Random Regression Forests (RRFs). Moreover, on the 2DHand dataset, we show better results than another state-of-the-art localization method based on RFs of [85], who applied their method on the same cross-validation split, using our landmark annotation. On the 2DSkull dataset, which was used for two public challenges in previous years, we also outperform both challenge winners, [85] as well as [62]. For 3DSpine, Chen et al. [12] use both RF and CNN predictions to significantly improve results compared to the pure RF-based method of Glocker et al. [44]. Similar results were obtained by the complex method of Yang et al. [160] using only CNNs. Both of them were outperformed by CNN methods that are highly tailored to this dataset [83, 123]. However, our SCN outperforms all these methods in terms of landmark localization error on this challenging dataset, without the need for complex or tailored implementations.

9.1.3 Other Applications of the SpatialConfiguration-Net

Additionally to datasets for anatomical landmark localization, we also show the applicability of the SCN to tasks of the computer vision community, specifically face point detection and human pose estimation (Section 8.1). There, while not tuned for these tasks and datasets, our SCN outperforms other RF-based methods for face point detection, and achieves state-of-the-art results for human pose estimation, outperforming other CNN-based methods. Especially in the dataset of human pose estimation, where the variability of the *spatial configuration* of landmarks is much larger as compared to other tasks, our improved *spatial configuration* block from [108], which allows higher-order terms of the graphical model as compared to models of [106, 142], greatly improves the results.

As a clinically and forensically application for anatomical landmark localization we show results for fully automatic age estimation of living individuals (Section 8.2). Here, as supported by clinical and forensic literature, we demonstrate that age-relevant information is concentrated in specific small regions of the input Magnetic Resonance (MR) images of the hand, clavicles, and teeth. While networks trained on whole MR images of the hand are

also able to identify these regions (Fig. 8.7), due to memory limitations of training CNNs, the images needed to be downsampled to fit into the memory. Thus, as the regions with age-relevant information only compose a small part of the whole input image, important information disappears due to the downsampling. When training networks only on small cropped images of the regions with age-relevant structure, the CNNs is able to observe this information in a much larger resolution, while at the same time other irrelevant image information is discarded, which reduces the chances of overfitting. We showed that this localization, cropping and age estimation framework achieved the best results on MR images of the left hand as compared to another method based on RF, while it enabled fully automatic multi-factorial age estimation of MR images of left hand, clavicles, and teeth in the first place.

We additionally apply our proposed SCN to multi-label segmentation, which also benefits from the information of the *spatial configuration* of anatomical structures (Section 8.3). Similar to landmark localization, in this task, local responses of anatomical structures can be used to estimate the position of other structures as well. Thus, we modify the SCN and adopt the loss function to be applicable to multi-label segmentation, specifically for the task Whole Heart Segmentation (WHS). Evaluation of the MICCAI 2017 Multi-Modality Whole Heart Segmentation Challenge (MM-WHS 2017) showed superior performance of our proposed localization and segmentation method, ranking first in the CT dataset and second in the MR dataset, and winning the challenge in overall. Although both datasets consist of only 20 training images, the results on the test sets with a Generalized Dice Similarity Coefficient (gDSC) of 0.908 in CT and 0.863 in MR, respectively, again showed that our proposed deep learning approach can cope with also only small amounts of training data. Furthermore, our comparison on the three-fold cross-validation on the training datasets confirmed that our SCN outperforms the U-Net baseline on both CT and MR datasets, while the gap is larger for the MR images, presumably due to their larger variance. However, we consider the results only preliminary, and an in-depth analysis of the SCN for multi-label segmentation is still required to confirm that the network effectively uses the *spatial configuration* of the anatomical structures.

Finally, we also proposed a coarse-to-fine framework for fully automatic vertebrae localization and segmentation (Section 8.4). Evaluation of the MICCAI 2019 Large Scale Vertebrae Segmentation Challenge (VerSe 2019) showed that our proposed framework outperforms all other participating methods, winning another competitive challenge in overall. In our three-step approach, first, we adapted the heatmap regression framework to predict the spinal column in low resolution. Then, we employ our SCN to localize and identified individual vertebral bodies. Finally, for each vertebra individually, we perform a binary segmentation to separate the vertebral body from the background and merge them based on their identification to generate the final multi-label vertebrae segmentation. Comparisons with other methods showed that our SCN outperforms other localization techniques, while in combination with the U-Net for binary segmentation, also in terms of segmentation performance our proposed framework surpasses the other participants.

9.2 Limitations and Future Work

Although our method shows remarkable performance and general applicability, several limitations need to be addressed in future work.

While the heatmap regression framework with CNNs is currently one of the best-performing methods for landmark localization, other methods also show great performance, e.g., landmark localization with reinforcement learning [39, 40]. Furthermore, while Gaussian functions have several beneficial properties, it needs to be confirmed whether they are the best representation of target heatmaps. Gaussian functions with different extent in different dimensions could also lead to improved results, while other representations like distance functions should be investigated. Also, when not using a heatmap image as the regression target, but a robust center of mass of the heatmap output, the specific target heatmap function would not need to be defined. A more comprehensive comparison could more strongly confirm the superior performance of the Gaussian heatmap regression scheme.

There are also some limitations of the SCN architecture. As the graphical model is learned and represented as convolution layers of a CNN, there are no hard constraints on the model. Thus, when the landmarks' locations are strongly restricted by anatomical constraints, the learned model could be too permissive. We observed this behavior in the VerSe 2019 challenge dataset (Section 8.4), where we also needed to define a heuristic model to obtain feasible locations of the vertebrae. Nevertheless, a too restrictive model could also be counterproductive since uncommon landmark configurations, e.g., in patients with pathologies, are not allowed and excluded from the possible solutions. As observed in previous work, too restrictive graphical models, e.g., Hidden Markov Models (HMMs) and Markov Random Fields (MRFs), can reduce prediction performance. Thus, while using them in initially [31, 42], some authors refrained from using them in later works [44, 147]. However, a model that is neither too restrictive nor too permissive that works for all possible tasks is impracticable to obtain. The model with a suitable balance, such that it is neither too restrictive nor too permissive, needs to be determined for each task and dataset independently.

While the multiplication of the SCN is important to simplify the localization task, such that both parts of the network can focus on their dedicated tasks, it may also hinder convergence speed. If one part of the network predicts values close to zero at a specific pixel location, due to the multiplication, the gradient obtained via Backpropagation (BP) is also close to zero at this location. Especially at the beginning of training, when the network outputs for all pixels are close to zero, the training is slow. This drawback could be resolved by performing intermediate supervision by applying the loss function on both the *local appearance* and *spatial configuration* part. However, then the simplification of the localization task could be hindered, as both parts are set up to solve the same problem. Nevertheless, this could improve training speed and performance for specific tasks.

Additionally, the multiplication of the two components of the SCN is also counterpro-

ductive in cases when one of them erroneously predicts outputs close to zero on locations near the groundtruth position. In this case, even if the other part of the SCN generates a correct prediction, due to the multiplication with zero, the final prediction will be zero as well. This could happen for occluded landmarks, where the *local appearance* part does not generate predictions. However, one could still use the output of the *spatial configuration* part in such cases to approximate the landmark location, as the estimation of the occluded landmark based on other landmarks is still possible. Since our datasets are lacking occluded or missing landmarks, we did not evaluate this behavior. In future work, we will address this limitation with appropriate datasets.

The multiplication of our SCN could also be seen as some kind of attention mechanism [105], as the individual parts of the network are set up to focus only on specific regions of intermediate feature outputs. However, comparisons to other methods incorporating attention mechanisms are missing and would be beyond the scope of this thesis.

Furthermore, other recent advances in terms of network building blocks and architectures are not evaluated. While our proposed SCN is tuned for the localization tasks and uses ideas from residual networks [52], more recent advances of building blocks of CNNs in general could be applied within the network architecture, e.g., densely connected convolutional networks [58] and self-normalizing networks [73], or improved fully connected architectures like U-Net++ [171]. Nevertheless, incorporating such advances into the SCN would probably improve the results, while the general fundamental concept of the SCN would stay the same (see Section 5.1), still reducing the amount of required training data.

Another influence on the performance of machine learning methods is data preprocessing and augmentation [17, 117] (Section 2.5.3). Especially in the medical imaging domain, where the number of training images is typically low, while excessive deformations still result in realistic images, data augmentation is crucial for good performance. Although we compare several network architectures with the same data preprocessing and augmentation hyperparameters for the individual datasets (Chapters 6 to 8), comparisons of different network architectures proposed within different methods are biased towards the effort put into finding favorable data augmentation hyperparameters. Thus, for an unbiased comparison of different network architectures, it would be required to train them within the same training and augmentation framework. To enable a fair comparison, for most of our experiments we also evaluate baseline U-Net that has been trained within the same framework, with the same data preprocessing and augmentation. Still, the biased comparison towards the results of network architectures taken from other papers remains.

There are further limitations of our proposed method in the task of multi-label segmentation (Sections 8.3 and 8.4). While the SCN provided the best segmentation results in the MM-WHS 2017 challenge among all participants, a detailed analysis of the results is missing. Although the results indicate that using the *spatial configuration* to transform the information from local responses to estimated positions of other structures is beneficial, it needs to be confirmed with further analysis. Moreover, the proposed framework that first localizes and then segments the individual substructures consists of two consecutive

CNNs that do not share any information. Extension to an end-to-end trainable approach would be desirable. Similarly, in the VerSe 2019 challenge, we were not able to adapt our SCN for the multi-label vertebrae segmentation, possibly due to the much more difficult differentiation among the individual labels, i.e., vertebrae. However, we still think that an end-to-end trainable CNN that performs localization, identification, and segmentation would be preferable. As a conclusion, we see a lot of potential and room for improvements in the extension of the SCN to the semantic segmentation tasks.

9.3 Conclusion

In this thesis we have shown how to combine information from the *local appearance* and *spatial configuration* of anatomical structures into a single end-to-end trained CNN, effectively representing a graphical model of their relative positions. Within the heatmap regression framework, our generic architecture does not require any postprocessing step for achieving state-of-the-art results in terms of landmark localization error on different 2D and 3D datasets, even when only limited amounts of training data are available. Furthermore, we showed the generic applicability of our proposed SCN architecture for multiple tasks, including age estimation of living individuals, and multi-label segmentation of heart substructures and individual vertebrae. Extensive comparisons demonstrate the excellent performance of our proposed frameworks, outperforming various state-of-the-art methods in several datasets, and winning two competitive challenges.

A

List of Acronyms

k-NN	k-Nearest Neighbors
ANN	Artificial Neural Network
ASM	Active Shape Model
BP	Backpropagation
CNN	Convolutional Neural Network
CT	Computed Tomography
DSC	Dice Similarity Coefficient
FNN	Feed-forward Neural Network
GD	Gradient Descent
gDSC	Generalized Dice Similarity Coefficient
GPU	Graphics Processing Unit
HMM	Hidden Markov Model
LDA	Linear Discriminant Analysis
LNN	Layered Neural Network
MAE	Mean Absolute Error
MAP	Maximum-A-Posteriori
MLE	Maximum Likelihood Estimation
MR	Magnetic Resonance
MRF	Markov Random Field
MRI	Magnetic Resonance Imaging
PCA	Principal Component Analysis
PS	Pictorial Structures
ReLU	Rectified Linear Unit
RF	Random Forest
RNN	Recurrent Neural Network
RRF	Random Regression Forest
SCN	SpatialConfiguration-Net

SGD	Stochastic Gradient Descent
SVM	Support Vector Machine
SVR	Support Vector Regression
WHS	Whole Heart Segmentation

B

List of Publications

Good news, everyone!

Prof. Hubert J. Farnsworth

B.1 Peer-Reviewed Publications Relevant for the Thesis

Working on my thesis at the Institute of Computer Graphics and Vision, and the Ludwig Boltzmann Institute for Clinical Forensic Imaging led to the following peer-reviewed publications:

2016

Regressing Heatmaps for Multiple Landmark Localization Using CNNs
Christian Payer, Darko Štern, Horst Bischof, and Martin Urschler
In: *Proceedings of Medical Image Computing and Computer-Assisted Intervention*
September 2016, Athens, Greece
Accepted for Oral Presentation, Acceptance Rate 5%

Automated Age Estimation from Hand MRI Volumes Using Deep Learning
Darko Štern, Christian Payer, Vincent Lepetit, and Martin Urschler
In: *Proceedings of Medical Image Computing and Computer-Assisted Intervention*
September 2016, Athens, Greece
Accepted for Poster Presentation

2017

Simultaneous Multi-Person Detection and Single-Person Pose Estimation With a Single Heatmap Regression Network
Christian Payer, Thomax Neff, Horst Bischof, Martin Urschler, and Darko Štern
In: *ICCV 2017 Pose Track Challenge: Human Pose Estimation and Tracking in the Wild*
September 2017, Venice, Italy

Multi-factorial Age Estimation from Skeletal and Dental MRI Volumes
Darko Štern, Philipp Kainz, Christian Payer, and Martin Urschler
In: *International Workshop on Machine Learning in Medical Imaging*
September 2017, Quebec, Canada
Accepted for Poster Presentation

Multi-label Whole Heart Segmentation Using CNNs and Anatomical Label Configurations
Christian Payer, Darko Štern, Horst Bischof, and Martin Urschler
In: *Statistical Atlases and Computational Models of the Heart. ACDC and MMWHS Challenges.*
September 2017, Quebec, Canada
Accepted for Oral Presentation, Best Paper Award, Winner of the MICCAI 2017 Multi-Modality Whole Heart Segmentation Challenge (MM-WHS 2017)

2018

Multi-Label Whole Heart Segmentation Using Anatomical Label Configurations and CNNs
Christian Payer, Darko Štern, Horst Bischof, and Martin Urschler
In: *Proceedings of the OAGM Workshop*
May 2018, Hall, Austria
Accepted for Oral Presentation

Volumetric Reconstruction from a Limited Number of Digitally Reconstructed Radiographs Using CNNs
Franz Thaler, Christian Payer, and Darko Štern
In: *Proceedings of the OAGM Workshop*
May 2018, Hall, Austria
Accepted for Oral Presentation

Automatic Age Estimation and Majority Age Classification from Multi-Factorial MRI Data
Darko Štern, Christian Payer, Nicola Giuliani, and Martin Urschler
In: *IEEE Journal of Biomedical and Health Informatics*

Reducing Acquisition Time for MRI-based Forensic Age Estimation
Bernhard Neumayer, Matthias Schloegl, Christian Payer, Thomas Widek, Sebastian Tschauner, Thomas Ehammer, Rudolf Stollberger, and Martin Urschler
In: *Scientific Reports*

2019

Integrating Spatial Configuration into Heatmap Regression Based CNNs for Landmark Localization
Christian Payer, Darko Štern, Horst Bischof, and Martin Urschler
In: *Medical Image Analysis*

Integrating Spatial Configuration into Heatmap Regression Based CNNs for Landmark Localization
Christian Payer, Darko Štern, Horst Bischof, and Martin Urschler
In: *Proceedings of International Conference on Medical Imaging with Deep Learning - Extended Abstract Track*
July 2019, London, United Kingdom
Accepted for Poster Presentation

Evaluation of Algorithms for Multi-Modality Whole Heart Segmentation: An Open-Access Grand Challenge
Xiahai Zhuang, Lei Li, Christian Payer, Darko Štern, Martin Urschler, Mattias P. Heinrich, Julien Oster, Chunliang Wang, Orjan Smedby, Cheng Bian, Xin Yang, Pheng-Ann Heng, Aliasghar Mortazi, Ulas Bagci, Guanyu Yang, Chenchen Sun, Gaetan Galisot, Jean-Yves Ramel, Thierry Brouard, Qianqian Tong, Weixin Si, Xiangyun Liao, Guodong Zeng, Zenglin Shi, Guoyan Zheng, Chengjia Wang, Tom MacGillivray, David Newby, Kawal Rhode, Sebastien Ourselin, Raad Mohiaddin, Jennifer Keegan, David Firmin, and Guang Yang
In: *Medical Image Analysis*

Automated Age Estimation from MRI Volumes of the Hand
Darko Štern, Christian Payer, and Martin Urschler
In: *Medical Image Analysis*

Evaluating Spatial Configuration Constrained CNNs for Localizing Facial and Body Pose Landmarks
Christian Payer, Darko Štern, and Martin Urschler
In: *Proceedings of Image and Vision Computing New Zealand*
December 2019, Dunedin, New Zealand
Accepted for Oral Presentation

2020

Coarse to Fine Vertebrae Localization and Segmentation with SpatialConfiguration-Net and U-Net
Christian Payer, Darko Štern, Horst Bischof, and Martin Urschler
In: *Proceedings of the International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*
February 2020, Valetta, Malta
Accepted for Oral Presentation
Winner of the MICCAI 2019 Large Scale Vertebrae Segmentation Challenge

VerSe: A Vertebrae Labelling and Segmentation Benchmark
Anjany Sekuboyina, Amirhossein Bayat, Malek E. Husseini, Maximilian Löffler, Markus Rempfler, Jan Kukačka, Giles Tetteh, Alexander Valentinitsch, Christian Payer, Darko Štern, Martin Urschler, Maodong Chen, Dalong Cheng, Nikolas Lessmann, Yujin Hu, Tianfu Wang, Dong Yang, Daguang Xu, Felix Ambellan, Stefan Zachowk, Tao Jiang, Xinjun Ma, Christoph Angerman, Xin Wang, Qingyue Wei, Kevin Brown, Matthias Wolf, Alexandre Kirszenberg, Élodie Puybareauq, Björn H. Menze and Jan S. Kirschke
In Review for Medical Image Analysis

B.2 Additional Peer-Reviewed Publications

While not being directly being part of my thesis, additionally, my work on other projects for the Institute of Computer Graphics and Vision, the Ludwig Boltzmann Institute for Lung Vascular Research, and the Ludwig Boltzmann Institute for Clinical Forensic Imaging led to the following peer-reviewed publications:

2015

Increased Tortuosity of Pulmonary Arteries in Patients with Pulmonary Hypertension in the Arteries
Michael Pienn, Christian Payer, Andrea Olschewski, Horst Olschewski, Martin Urschler, and Zoltán Bálint
In: *Proceedings of the Conference on Medical Image Understanding and Analysis*
July 2015, Lincoln, United Kingdom
Accepted for Oral Presentation, Best Oral Presentation Award

Automatic Artery-Vein Separation from Thoracic CT Images Using Integer Programming
Christian Payer, Michael Pienn, Zoltán Bálint, Andrea Olschewski, Horst Olschewski, and Martin Urschler
In: *Proceedings of Medical Image Computing and Computer-Assisted Intervention*
October 2015, Munich, Germany
Accepted for Oral Presentation, Acceptance Rate 4.4%, Runners-up for MICCAI Young Scientist Award

2016

Automated Integer Programming Based Separation of Arteries and Veins from Thoracic CT Images
Christian Payer, Michael Pienn, Zoltán Bálint, Alexander Shekhovtsov, Emina Talakic, Eszter Nagy, Andrea Olschewski, Horst Olschewski, and Martin Urschler
In: *Medical Image Analysis*

2017

Generative Adversarial Network based Synthesis for Supervised Medical Image Segmentation
Thomas Neff, Christian Payer, Darko Štern, and Martin Urschler
In: *Proceedings of the OAGM&ARW Joint Workshop*
May 2017, Vienna, Austria
Accepted for Oral Presentation, Best Paper Award

2018

Pulmonary Lobe Segmentation in CT Images Using Alpha-Expansion
Nicola Giuliani, Christian Payer, Michael Pienn, Horst Olschewski, and Martin Urschler
In: *Proceedings of the International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*
January 2018, Funchal, Portugal
Accepted for Poster Presentation

Generative Adversarial Networks to Synthetically Augment Data for Deep Learning based Image Segmentation
Thomas Neff, Christian Payer, Darko Štern, and Martin Urschler
In: *Proceedings of the OAGM Workshop*
May 2018, Hall, Austria
Accepted for Poster Presentation

Instance Segmentation and Tracking with Cosine Embeddings and Recurrent Hourglass Networks
Christian Payer, Darko Štern, Thomas Neff, Horst Bischof, and Martin Urschler
In: *Proceedings of Medical Image Computing and Computer-Assisted Intervention*
September 2018, Granada, Spain
Accepted for Oral Presentation, Acceptance Rate 4.2%

Sparse-View CT Reconstruction Using Wasserstein GANs
Franz Thaler, Kerstin Hammernik, Christian Payer, Martin Urschler, and Darko Štern
In: *International Workshop on Machine Learning for Medical Image Reconstruction*
September 2018, Granada, Spain
Accepted for Poster Presentation

Healthy Lung Vessel Morphology Derived From Thoracic Computed Tomography
Michael Pienn, Caroline Burgard, Christian Payer, Alexander Avian, Martin Urschler, Rudolf Stollberger, Andrea Olschewski, Horst Olschewski, Thorsten Johnson, Felix G. Meinel, and Zoltán Bálint
In: *Frontiers in Physiology*

2019

Quantitative CT-derived Vessel Metrics in Idiopathic Pulmonary Fibrosis: A Structure-function Study
Joseph Jacob, Michael Pienn, Christian Payer, Martin Urschler, Maria Kokosi, Anand Devaraj, Athol U. Wells, and Horst Olschewski
In: *Respirology*

Segmenting and Tracking Cell Instances with Cosine Embeddings and Recurrent Hourglass Networks
Christian Payer, Darko Štern, Marlies Feiner, Horst Bischof, and Martin Urschler
In: *Medical Image Analysis*
Runners-up for Medical Image Analysis Best Paper Award

Matwo-CapsNet: A Multi-Label Semantic Segmentation Capsules Network
Savinien Bonheur, Darko Štern, Christian Payer, Michael Pienn, Horst Olschewski, Martin Urschler
In: *Proceedings of Medical Image Computing and Computer-Assisted Intervention*
October 2019, Granada, Spain
Accepted for Poster Presentation

Bibliography

[1] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mane, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viegas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X. (2016). TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. *arXiv Prepr. arXiv1603.04467*. (page 74)

[2] Amato, V., Giannachi, L., Irace, C., and Corona, C. (2010). Accuracy of Pedicle Screw Placement in the Lumbosacral Spine Using Conventional Technique: Computed Tomography Postoperative Assessment in 102 Consecutive Patients. *J. Neurosurg. Spine*, 12(3):306–313. (page 136)

[3] Andriluka, M., Roth, S., and Schiele, B. (2012). Discriminative Appearance Models for Pictorial Structures. *Int. J. Comput. Vis.*, 99(3):259–280. (page 44)

[4] Avendi, M., Kheradvar, A., and Jafarkhani, H. (2016). A Combined Deep-Learning and Deformable-Model Approach to Fully Automatic Segmentation of the Left Ventricle in Cardiac MRI. *Med. Image Anal.*, 30:108–119. (page 128)

[5] Ba, J. L., Kiros, J. R., and Hinton, G. E. (2016). Layer Normalization. *arXiv Prepr. arXiv1607.06450*. (page 27)

[6] Beichel, R., Bischof, H., Leberl, F., and Sonka, M. (2005). Robust Active Appearance Models and Their Application to Medical Image Analysis. *IEEE Trans. Med. Imaging*, 24(9):1151–1169. (page 3)

[7] Betts, J. G., Desaix, P., Johnson, E., Johnson, J. E., Korol, O., Kruse, D. H., Poe, B., Wise, J. A., Womble, M., and Young, K. A. (2013). *Anatomy & Physiology*. OpenStax College, 1st edition. (page 2, 14)

[8] Bier, B., Unberath, M., Zaech, J.-N., Fotouhi, J., Armand, M., Osgood, G., Navab, N., and Maier, A. (2018). X-ray-transform Invariant Anatomical Landmark Detection for Pelvic Trauma Surgery. In *Proc. Med. Image Comput. Comput. Interv.*, pages 55–63. (page 53)

[9] Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer-Verlag New York, 1st edition. (page 7, 8, 9, 10, 11, 14, 16)

[10] Breiman, L. (2001). Random Forests. *Mach. Learn.*, 45(1):5–32. (page 13, 47)

[11] Carman, D. L., Browne, R. H., and Birch, J. G. (1990). Measurement of Scoliosis and Kyphosis Radiographs. Intraobserver and Interobserver Variation. *J. Bone Jt. Surg.*, 72(3):328–333. (page 4)

[12] Chen, H., Shen, C., Qin, J., Ni, D., Shi, L., Cheng, J. C. Y., and Heng, P.-A. (2015). Automatic Localization and Identification of Vertebrae in Spine CT via a Joint Learning Model with Deep Neural Networks. In *Proc. Med. Image Comput. Comput. Interv.*, pages 515–522. (page 53, 96, 137, 150)

[13] Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., and Yuille, A. L. (2014). Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs. *Int. Conf. Learn. Represent. arXiv1412.7062*. (page 34)

[14] Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., and Yuille, A. L. (2018). DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. *IEEE Trans. Pattern Anal. Mach. Intell.*, 40(4):834–848. (page 54)

[15] Chen, X. and Yuille, A. L. (2014). Articulated Pose Estimation by a Graphical Model with Image Dependent Pairwise Relations. In *Adv. Neural Inf. Process. Syst.*, pages 1736–1744. (page 50, 117, 118)

[16] Cireşan, D. C., Giusti, A., Gambardella, L. M., and Schmidhuber, J. (2012a). Deep Neural Networks Segment Neuronal Membranes in Electron Microscopy Images. In *Adv. Neural Inf. Process. Syst.*, pages 2843–2851. (page 1)

[17] Cireşan, D. C., Meier, U., Masci, J., Gambardella, L. M., and Schmidhuber, J. (2011). Flexible, High Performance Convolutional Neural Networks for Image Classification. In *Proc. Int. Jt. Conf. Artif. Intell.*, pages 1237–1242. (page 36, 153)

[18] Cireşan, D. C., Meier, U., and Schmidhuber, J. (2012b). Multi-Column Deep Neural Networks for Image Classification. In *Proc. Conf. Comput. Vis. Pattern Recognit.*, pages 3642–3649. IEEE. (page 1, 27)

[19] Clevert, D.-A., Unterthiner, T., and Hochreiter, S. (2015). Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs). *Int. Conf. Learn. Represent. arXiv1511.07289*. (page 19)

[20] Cootes, T. F., Edwards, G. J., and Taylor, C. J. (1998). Active Appearance Models. In *Proc. Eur. Conf. Comput. Vis.*, pages 484–498. (page 42)

[21] Cootes, T. F., Ionita, M. C., Lindner, C., and Sauer, P. (2012). Robust and Accurate Shape Model Fitting Using Random Forest Regression Voting. In *Proc. Eur. Conf. Comput. Vis.*, pages 278–291. (page 42, 47, 48, 49, 115, 116)

[22] Cootes, T. F., Taylor, C. J., Cooper, D. H., and Graham, J. (1995). Active Shape Models-Their Training and Application. *Comput. Vis. Image Underst.*, 61(1):38–59. (page 4, 40, 41, 46)

[23] Cortes, C. and Vapnik, V. (1995). Support-Vector Networks. *Mach. Learn.*, 20(3):273–297. (page 13)

[24] Criminisi, A., Robertson, D., Konukoglu, E., Shotton, J., Pathak, S., White, S., and Siddiqui, K. (2013). Regression Forests for Efficient Anatomy Detection and Localization in Computed Tomography Scans. *Med. Image Anal.*, 17(8):1293–1303. (page 47, 49, 123)

[25] Cristinacce, D. and Cootes, T. F. (2008). Automatic Feature Localisation with Constrained Local Models. *Pattern Recognit.*, 41(10):3054–3067. (page 42, 46)

[26] Demirjian, A., Goldstein, H., and Tanner, J. M. (1973). A New System of Dental Age Assessment. *Hum. Biol.*, 45(2):211–27. (page 119)

[27] Dollár, P., Welinder, P., and Perona, P. (2010). Cascaded Pose Regression. In *Proc. Comput. Vis. Pattern Recognit.*, pages 1078–1085. (page 50)

[28] Donner, R., Menze, B. H., Bischof, H., and Langs, G. (2013). Global Localization of 3D Anatomical Structures by Pre-Filtered Hough Forests and Discrete Optimization. *Med. Image Anal.*, 17(8):1304–1314. (page 44, 45, 47, 49, 149)

[29] Duchi, J., Hazan, E., and Singer, Y. (2011). Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. *J. Mach. Learn. Res.*, 12:2121–2159. (page 23)

[30] Dumoulin, V. and Visin, F. (2016). A Guide to Convolution Arithmetic for Deep Learning. *arXiv Prepr. arXiv1603.07285*. (page 28)

[31] Ebner, T., Štern, D., Donner, R., Bischof, H., and Urschler, M. (2014). Towards Automatic Bone Age Estimation from MRI: Localization of 3D Anatomical Landmarks. In *Proc. Med. Image Comput. Comput. Interv.*, pages 421–428. (page 47, 48, 49, 77, 87, 90, 91, 92, 150, 152)

[32] Eichner, M. and Ferrari, V. (2009). Better Appearance Models for Pictorial Structures. In *Procedings Br. Mach. Vis. Conf.*, pages 3.1–3.11. British Machine Vision Association. (page 44)

[33] Erol, A., Bebis, G., Nicolescu, M., Boyle, R. D., and Twombly, X. (2007). Vision-based Hand Pose Estimation: A Review. *Comput. Vis. Image Underst.*, 108(1-2):52–73. (page 40)

[34] Felzenszwalb, P. F. and Huttenlocher, D. P. (2005). Pictorial Structures for Object Recognition. *Int. J. Comput. Vis.*, 61(1):55–79. (page 4, 42, 43, 44, 46)

[35] Fischler, M. A. and Elschlager, R. A. (1973). The Representation and Matching of Pictorial Structures. *IEEE Trans. Comput.*, C-22(1):67–92. (page 42, 43)

[36] Forsberg, D., Lundström, C., Andersson, M., Vavruch, L., Tropp, H., and Knutsson, H. (2013). Fully Automatic Measurements of Axial Vertebral Rotation for Assessment of Spinal Deformity in Idiopathic Scoliosis. *Phys. Med. Biol.*, 58(6):1775–1787. (page 136)

[37] Galisot, G., Brouard, T., and Ramel, J.-Y. (2018). Local Probabilistic Atlases and a Posteriori Correction for the Segmentation of Heart Images. In *Stat. Atlases Comput. Model. Hear. ACDC MMWHS Challenges.*, pages 207–214. Springer International Publishing. (page 134)

[38] Gertych, A., Zhang, A., Sayre, J., Pospiech-Kurkowska, S., and Huang, H. (2007). Bone Age Assessment of Children Using a Digital Hand Atlas. *Comput. Med. Imaging Graph.*, 31(4-5):322–331. (page 76, 86)

[39] Ghesu, F. C., Georgescu, B., Grbic, S., Maier, A., Hornegger, J., and Comaniciu, D. (2018). Towards Intelligent Robust Detection of Anatomical Structures in Incomplete Volumetric Data. *Med. Image Anal.*, 48:203–213. (page 54, 152)

[40] Ghesu, F. C., Georgescu, B., Zheng, Y., Grbic, S., Maier, A., Hornegger, J., and Comaniciu, D. (2017). Multi-Scale Deep Reinforcement Learning for Real-Time 3D-Landmark Detection in CT Scans. *IEEE Trans. Pattern Anal. Mach. Intell.*, Early Acce:1–14. (page 54, 152)

[41] Girshick, R. (2015). Fast R-CNN. In *Proc. Int. Conf. Comput. Vis.*, pages 1440–1448. IEEE. (page 1)

[42] Glocker, B., Feulner, J., Criminisi, A., Haynor, D. R., and Konukoglu, E. (2012). Automatic Localization and Identification of Vertebrae in Arbitrary Field-of-View CT Scans. In *Proc. Med. Image Comput. Comput. Interv.*, pages 590–598. (page 44, 45, 47, 48, 49, 152)

[43] Glocker, B., Konukoglu, E., and Haynor, D. R. (2016). Random Forests for Localization of Spinal Anatomy. In *Med. Image Recognition, Segmentation Parsing*, pages 93–110. Elsevier. (page 47)

[44] Glocker, B., Zikic, D., Konukoglu, E., Haynor, D. R., and Criminisi, A. (2013). Vertebrae Localization in Pathological Spine CT via Dense Classification from Sparse Annotations. In *Proc. Med. Image Comput. Comput. Interv.*, pages 262–270. (page 40, 45, 47, 77, 82, 95, 96, 136, 137, 150, 152)

[45] Glorot, X. and Bengio, Y. (2010). Understanding the Difficulty of Training Deep Feedforward Neural Networks. In *Proc. Int. Conf. Artif. Intell. Stat.*, pages 249–256. (page 26)

[46] Glorot, X., Bordes, A., and Bengio, Y. (2011). Deep Sparse Rectifier Neural Networks. In *Proc. Int. Conf. Artif. Intell. Stat.*, pages 315–323. (page 18)

[47] Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press. (page 7, 12, 16, 28, 33, 34)

[48] Greulich, W. W. and Pyle, S. I. (1959). *Radiographic Atlas of Skeletal Development of the Hand and Wrist*. Stanford University Press, Stanford, CA, 2nd edition. (page 119)

[49] Hahnloser, R. H. R., Sarpeshkar, R., Mahowald, M. A., Douglas, R. J., and Seung, H. S. (2000). Digital Selection and Analogue Amplification Coexist in a Cortex-Inspired Silicon Circuit. *Nature*, 405(6789):947–951. (page 1, 18)

[50] Halevy, A., Norvig, P., and Pereira, F. (2009). The Unreasonable Effectiveness of Data. *IEEE Intell. Syst.*, 24(2):8–12. (page 12)

[51] He, K., Zhang, X., Ren, S., and Sun, J. (2015). Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. In *Proc. Int. Conf. Comput. Vis.*, pages 1026–1034. IEEE. (page 19, 26, 79, 80, 81, 132)

[52] He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep Residual Learning for Image Recognition. In *Proc. Comput. Vis. Pattern Recognit.*, pages 770–778. IEEE. (page 16, 26, 33, 70, 88, 153)

[53] Heimann, T. and Meinzer, H. P. (2009). Statistical Shape Models for 3D Medical Image Segmentation: A Review. *Med. Image Anal.*, 13(4):543–563. (page 3)

[54] Heinrich, M. P. and Oster, J. (2018). MRI Whole Heart Segmentation Using Discrete Nonlinear Registration and Fast Non-local Fusion. In *Stat. Atlases Comput. Model. Hear. ACDC MMWHS Challenges.*, pages 233–241. Springer International Publishing. (page 134)

[55] Hinton, G. E., Srivastava, N., and Swersky, K. (2012). Lecture 6a: Overview of Mini-Batch Gradient Descent. (page 23)

[56] Hochreiter, S. (1991). *Untersuchungen zu dynamischen neuronalen Netzen*. PhD thesis. (page 17, 26)

[57] Hornik, K., Stinchcombe, M., and White, H. (1989). Multilayer Feedforward Networks are Universal Approximators. *Neural Networks*, 2(5):359–366. (page 17)

[58] Huang, G., Liu, Z., van der Maaten, L., and Weinberger, K. Q. (2017). Densely Connected Convolutional Networks. In *Proc. Comput. Vis. Pattern Recognit.*, pages 2261–2269. IEEE. (page 26, 153)

[59] Huang, S.-H., Chu, Y.-H., Lai, S.-H., and Novak, C. L. (2009). Learning-Based Vertebra Detection and Iterative Normalized-Cut Segmentation for Spinal MRI. *IEEE Trans. Med. Imaging*, 28(10):1595–1605. (page 46)

[60] Hubel, D. H. and Wiesel, T. N. (1962). Receptive Fields, Binocular Interaction and Functional Architecture in the Cat's Visual Cortex. *J. Physiol.*, 160(1):106–154. (page 28)

[61] Ibragimov, B., Likar, B., Pernuš, F., and Vrtovec, T. (2012). A Game-Theoretic Framework for Landmark-Based Image Segmentation. *IEEE Trans. Med. Imaging*, 31(9):1761–1776. (page 40, 44)

[62] Ibragimov, B., Likar, B., Pernuš, F., and Vrtovec, T. (2014). Shape Representation for Efficient Landmark-Based Segmentation in 3-D. *IEEE Trans. Med. Imaging*, 33(4):861–874. (page 3, 93, 94, 150)

[63] Ioffe, S. and Szegedy, C. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In *Proc. Int. Conf. Mach. Learn.*, volume 37, pages 448–456. (page 27)

[64] Janssens, R., Zeng, G., and Zheng, G. (2018). Fully Automatic Segmentation of Lumbar Vertebrae from CT Images Using Cascaded 3D Fully Convolutional Networks. In *Proc. Int. Symp. Biomed. Imaging*, pages 893–897. IEEE. (page 136)

[65] Johnson, H. J. and Christensen, G. E. (2002). Consistent Landmark and Intensity-Based Image Registration. *IEEE Trans. Med. Imaging*, 21(5):450–461. (page 3)

[66] Kamnitsas, K., Ledig, C., Newcombe, V. F., Simpson, J. P., Kane, A. D., Menon, D. K., Rueckert, D., and Glocker, B. (2017). Efficient Multi-Scale 3D CNN with Fully Connected CRF for Accurate Brain Lesion Segmentation. *Med. Image Anal.*, 36:61–78. (page 54)

[67] Kang, D. (2012). Heart Chambers and Whole Heart Segmentation Techniques: Review. *J. Electron. Imaging*, 21(1):010901. (page 128)

[68] Kaplowitz, P., Srinivasan, S., He, J., McCarter, R., Hayeri, M. R., and Sze, R. (2011). Comparison of Bone Age Readings by Pediatric Endocrinologists and Pediatric Radiologists Using two Bone Age Atlases. *Pediatr. Radiol.*, 41(6):690–693. (page 119)

[69] Karim, R., Blake, L.-E., Inoue, J., Tao, Q., Jia, S., Housden, R. J., Bhagirath, P., Duval, J.-L., Varela, M., Behar, J. M., Cadour, L., van der Geest, R. J., Cochet, H., Drangova, M., Sermesant, M., Razavi, R., Aslanidi, O., Rajani, R., and Rhode, K. (2018). Algorithms for Left Atrial Wall Segmentation and Thickness - Evaluation on an Open-Source CT and MRI Image Database. *Med. Image Anal.*, 50:36–53. (page 128)

[70] Kelm, M. B., Wels, M., Zhou, K. S., Seifert, S., Suehling, M., Zheng, Y., and Comaniciu, D. (2013). Spine Detection in CT and MR Using Iterated Marginal Space Learning. *Med. Image Anal.*, 17(8):1283–1292. (page 46)

[71] Kingma, D. P. and Ba, J. (2015). Adam: A Method for Stochastic Optimization. *Int. Conf. Learn. Represent. arXiv1412.6980*. (page 1, 23, 24, 79, 123, 132, 141)

[72] Kitamura, Y., Li, Y., Ito, W., and Ishikawa, H. (2016). Data-Dependent Higher-Order Clique Selection for Artery–Vein Segmentation by Energy Minimization. *Int. J. Comput. Vis.*, 117(2):142–158. (page 3)

[73] Klambauer, G., Unterthiner, T., Mayr, A., and Hochreiter, S. (2017). Self-Normalizing Neural Networks. In *Adv. Neural Inf. Process. Syst.* (page 19, 27, 153)

[74] Klinder, T., Ostermann, J., Ehm, M., Franz, A., Kneser, R., and Lorenz, C. (2009). Automated Model-Based Vertebra Detection, Identification, and Segmentation in CT images. *Med. Image Anal.*, 13(3):471–482. (page 3, 44, 137)

[75] Knez, D., Likar, B., Pernus, F., and Vrtovec, T. (2016). Computer-Assisted Screw Size and Insertion Trajectory Planning for Pedicle Screw Placement Surgery. *IEEE Trans. Med. Imaging*, 35(6):1420–1430. (page 136)

[76] Korez, R., Likar, B., Pernuš, F., and Vrtovec, T. (2016). Model-Based Segmentation of Vertebral Bodies from MR Images with 3D CNNs. In *Proc. Med. Image Comput. Comput. Interv.*, pages 433–441. (page 137)

[77] Köstinger, M., Wohlhart, P., Roth, P. M., and Bischof, H. (2011). Annotated Facial Landmarks in the Wild: A Large-Scale, Real-World Database for Facial Landmark Localization. In *Proc. Int. Conf. Comput. Vis. Work.*, pages 2144–2151. (page 115)

[78] Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. In *Adv. Neural Inf. Process. Syst.*, pages 1097–1105. (page 1, 13, 16, 25, 27, 34)

[79] LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep Learning. *Nature*, 521(7553):436–444. (page 10, 49, 65)

[80] LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998a). Gradient-based Learning Applied to Document Recognition. *Proc. IEEE*, 86(11):2278–2323. (page 1, 27)

[81] LeCun, Y., Bottou, L., Orr, G. B., and Müller, K. R. (1998b). Efficient BackProp. In *Neural Networks: Tricks of the Trade*, pages 9–50. Springer Berlin Heidelberg. (page 26)

[82] Lee, S. C., Shim, J. S., Seo, S. W., Lim, K. S., and Ko, K. R. (2013). The Accuracy of Current Methods in Determining the Timing of Epiphysiodesis. *Bone Joint J.*, 95-B(7):993–1000. (page 119)

[83] Liao, H., Mesfin, A., and Luo, J. (2018). Joint Vertebrae Identification and Localization in Spinal CT Images by Combining Short- and Long-Range Contextual Information. *IEEE Trans. Med. Imaging*, 37(5):1266–1275. (page 54, 96, 97, 137, 150)

[84] Lin, M., Chen, Q., and Yan, S. (2014). Network In Network. *Int. Conf. Learn. Represent. arXiv1312.4400*. (page 33)

[85] Lindner, C., Bromiley, P. A., Ionita, M. C., and Cootes, T. F. (2015). Robust and Accurate Shape Model Matching Using Random Forest Regression-Voting. *IEEE Trans. Pattern Anal. Mach. Intell.*, 37(9):1862–1874. (page 40, 42, 47, 48, 49, 62, 77, 87, 90, 93, 94, 115, 116, 149, 150)

[86] Lindner, C., Wang, C.-W., Huang, C.-T., Li, C.-H., Chang, S.-W., and Cootes, T. F. (2016). Fully Automatic System for Accurate Localisation and Analysis of Cephalometric Landmarks in Lateral Cephalograms. *Sci. Rep.*, 6:33581. (page 90)

[87] Liu, D., Zhou, K. S., Bernhardt, D., and Comaniciu, D. (2010). Search Strategies for Multiple Landmark Detection by Submodular Maximization. In *Proc. Comput. Vis. Pattern Recognit.*, pages 2831–2838. (page 46)

[88] Long, J., Shelhamer, E., and Darrell, T. (2015). Fully Convolutional Networks for Semantic Segmentation. In *Proc. Comput. Vis. Pattern Recognit.*, pages 3431–3440. IEEE. (page 1, 33, 50, 60, 148)

[89] Lowekamp, B. C., Chen, D. T., Ibáñez, L., and Blezek, D. (2013). The Design of SimpleITK. *Front. Neuroinform.*, 7. (page 73)

[90] Maas, A. L., Hannun, A. Y., and Ng, A. Y. (2013). Rectifier Nonlinearities Improve Neural Network Acoustic Models. In *Proc. Int. Conf. Mach. Learn.*, volume 28, page 6. (page 19, 81)

[91] Mader, A. O., Lorenz, C., von Berg, J., and Meyer, C. (2019). Automatically Localizing a Large Set of Spatially Correlated Key Points: A Case Study in Spine Imaging. In *Proc. Med. Image Comput. Comput. Interv.*, pages 384–392. (page 54)

[92] Martin, D. D., Wit, J. M., Hochberg, Z., van Rijn, R. R., Fricke, O., Werther, G., Cameron, N., Hertel, T., Wudy, S. A., Butler, G., Thodberg, H. H., Binder, G., and Ranke, M. B. (2011). The Use of Bone Age in Clinical Practice - Part 2. *Horm. Res. Paediatr.*, 76(1):10–16. (page 119)

[93] McCulloch, W. S. and Pitts, W. (1943). A Logical Calculus of the Ideas Immanent in Nervous Activity. *Bull. Math. Biophys.*, 5(4):115–133. (page 13)

[94] Mendis, S., Puska, P., and Norrving, B. (2011). Global Atlas on Cardiovascular Disease Prevention and Control. *World Heal. Organ.* (page 128)

[95] Mitchell, T. M. (1997). *Machine Learning*. McGraw-Hill, 1st edition. (page 8)

[96] Mortazi, A., Burt, J., and Bagci, U. (2018). Multi-Planar Deep Segmentation Networks for Cardiac Substructures from MRI and CT. In *Stat. Atlases Comput. Model. Hear. ACDC MMWHS Challenges.*, pages 199–206. Springer International Publishing. (page 134)

[97] Nair, V. and Hinton, G. E. (2010). Rectified Linear Units Improve Restricted Boltzmann Machines. In *Proc. Int. Conf. Mach. Learn.*, pages 807–814. (page 18)

[98] Nesterov, Y. (1983). A Method of Solving A Convex Programming Problem With Convergence rate O(1/kˆ2). In *Sov. Math. Dokl.*, volume 27, pages 372–376. (page 23, 79, 80, 81, 141)

[99] Neumayer, B., Schloegl, M., Payer, C., Widek, T., Tschauner, S., Ehammer, T., Stollberger, R., and Urschler, M. (2018). Reducing Acquisition Time for MRI-based Forensic Age Estimation. *Sci. Rep.*, 8(1):2063. (page 122)

[100] Newell, A., Yang, K., and Deng, J. (2016). Stacked Hourglass Networks for Human Pose Estimation. In *Proc. Eur. Conf. Comput. Vis.*, pages 483–499. (page 52, 60, 64, 70, 79, 115, 117, 118, 148)

[101] Ngo, T. A., Lu, Z., and Carneiro, G. (2017). Combining Deep Learning and Level Set for the Automated Segmentation of the Left Ventricle of the Heart from Cardiac Cine Magnetic Resonance. *Med. Image Anal.*, 35:159–171. (page 128)

[102] Odena, A., Dumoulin, V., and Olah, C. (2016). Deconvolution and Checkerboard Artifacts. *Distill*. (page 33)

[103] Oktay, O., Ferrante, E., Kamnitsas, K., Heinrich, M., Bai, W., Caballero, J., Cook, S. A., de Marvao, A., Dawes, T., O'Regan, D. P., Kainz, B., Glocker, B., and Rueckert, D. (2018a). Anatomically Constrained Neural Networks (ACNNs): Application to Cardiac Image Enhancement and Segmentation. *IEEE Trans. Med. Imaging*, 37(2):384–395. (page 54)

[104] Oktay, O., Rueckert, D., Bai, W., Guerrero, R., Rajchl, M., de Marvao, A., O'Regan, D. P., Cook, S. A., Heinrich, M. P., and Glocker, B. (2017). Stratified Decision Forests for Accurate Anatomical Landmark Localization in Cardiac Images. *IEEE Trans. Med. Imaging*, 36(1):332–342. (page 3, 40)

[105] Oktay, O., Schlemper, J., Folgoc, L. L., Lee, M., Heinrich, M., Misawa, K., Mori, K., McDonagh, S., Hammerla, N. Y., Kainz, B., Glocker, B., and Rueckert, D. (2018b). Attention U-Net: Learning Where to Look for the Pancreas. *Proc. 1st Conf. Med. Imaging with Deep Learn.*, pages 1–10. (page 153)

[106] Payer, C., Štern, D., Bischof, H., and Urschler, M. (2016). Regressing Heatmaps for Multiple Landmark Localization Using CNNs. In *Proc. Med. Image Comput. Comput. Interv.*, pages 230–238. (page 53, 54, 55, 72, 74, 77, 80, 85, 87, 88, 90, 91, 92, 103, 118, 137, 150)

[107] Payer, C., Štern, D., Bischof, H., and Urschler, M. (2018). Multi-label Whole Heart Segmentation Using CNNs and Anatomical Label Configurations. In Pop, M., Sermesant, M., Jodoin, P.-M., Lalande, A., Zhuang, X., Yang, G., Young, A., and Bernard, O., editors, *Stat. Atlases Comput. Model. Hear. ACDC MMWHS Challenges.*, pages 190–198. Springer International Publishing. (page 55, 128)

[108] Payer, C., Štern, D., Bischof, H., and Urschler, M. (2019a). Integrating Spatial Configuration into Heatmap Regression Based CNNs for Landmark Localization. *Med. Image Anal.*, 54:207–219. (page 55, 80, 81, 85, 87, 88, 150)

[109] Payer, C., Štern, D., Bischof, H., and Urschler, M. (2020). Coarse to Fine Vertebrae Localization and Segmentation with SpatialConfiguration-Net and U-Net. In *Proc. Int. Jt. Conf. Comput. Vision, Imaging Comput. Graph. Theory Appl.*, volume 5, pages 124–133. SCITEPRESS - Science and Technology Publications. (page 55, 137)

[110] Payer, C., Stern, D., and Urschler, M. (2019b). Evaluating Spatial Configuration Constrained CNNs for Localizing Facial and Body Pose Landmarks. In *Proc. Int. Conf. Image Vis. Comput. New Zeal.*, pages 1–6. IEEE. (page 54, 55, 114)

[111] Pekar, V., Bystrov, D., Heese, H. S., Dries, S. P. M., Schmidt, S., Grewer, R., den Harder, C. J., Bergmans, R. C., Simonetti, A. W., and van Muiswinkel, A. M. (2007). Automated Planning of Scan Geometries in Spine MRI Scans. In *Proc. Med. Image Comput. Comput. Interv.*, pages 601–608. (page 46)

[112] Peng, P., Lekadir, K., Gooya, A., Shao, L., Petersen, S. E., and Frangi, A. F. (2016). A Review of Heart Chamber Segmentation for Structural and Functional Analysis Using Cardiac Magnetic Resonance Imaging. *Magn. Reson. Mater. Physics, Biol. Med.*, 29(2):155–195. (page 128)

[113] Peng, Z., Zhong, J., Wee, W., and Lee, J.-h. (2005). Automated Vertebra Detection and Segmentation from the Whole Spine MR Images. In *Proc. Eng. Med. Biol.*, pages 2527–2530. IEEE. (page 46)

[114] Pfister, T., Charles, J., and Zisserman, A. (2015). Flowing ConvNets for Human Pose Estimation in Videos. In *Proc. Int. Conf. Comput. Vis.*, pages 1913–1921. (page 5, 50, 51, 52, 60, 63, 70, 74, 79, 148, 149)

[115] Pfister, T., Simonyan, K., Charles, J., and Zisserman, A. (2014). Deep Convolutional Neural Networks for Efficient Pose Estimation in Gesture Videos. In *Asian Conf. Comput. Vis.*, pages 538–552. (page 49)

[116] Potesil, V., Kadir, T., Platsch, G., and Brady, M. (2015). Personalized Graphical Models for Anatomical Landmark Localization in Whole-Body Medical Images. *Int. J. Comput. Vis.*, 111(1):29–49. (page 44)

[117] Ronneberger, O., Fischer, P., and Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. In *Proc. Med. Image Comput. Comput. Interv.*, pages 234–241. (page 33, 36, 52, 55, 60, 64, 70, 80, 137, 140, 148, 149, 153)

[118] Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning Representations by Back-Propagating Errors. *Nature*, 323(6088):533–536. (page 13, 24)

[119] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Fei-Fei, L. (2015). ImageNet Large Scale Visual Recognition Challenge. *Int. J. Comput. Vis.*, 115(3):211–252. (page 1, 10, 49)

[120] Salimans, T. and Kingma, D. P. (2016). Weight Normalization: A Simple Reparameterization to Accelerate Training of Deep Neural Networks. In *Adv. Neural Inf. Process. Syst.* (page 27)

[121] Sapp, B. and Taskar, B. (2013). MODEC: Multimodal Decomposable Models for Human Pose Estimation. In *Proc. Comput. Vis. Pattern Recognit.*, pages 3674–3681. (page 115, 117)

[122] Sekuboyina, A., Bayat, A., Husseini, M. E., Löffler, M., Rempfler, M., Kukačka, J., Tetteh, G., Valentinitsch, A., Payer, C., Urschler, M., Chen, M., Cheng, D., Lessmann, N., Hu, Y., Wang, T., Yang, D., Xu, D., Ambellan, F., Zachowk, S., Jiang, T., Ma, X., Angerman, C., Wang, X., Wei, Q., Brown, K., Wolf, M., Kirszenberg, A., Puybareauq, É., Menze, B. H., and Kirschke, J. S. (2020). VerSe: A Vertebrae Labelling and Segmentation Benchmark. *arXiv Prepr. arXiv2001.09193*. (page 137, 143, 144)

[123] Sekuboyina, A., Rempfler, M., Kukačka, J., Tetteh, G., Valentinitsch, A., Kirschke, J. S., and Menze, B. H. (2018). Btrfly Net: Vertebrae Labelling with Energy-Based Adversarial Learning of Local Spine Prior. *Proc. Med. Image Comput. Comput. Interv.*, pages 649–657. (page 53, 96, 97, 136, 137, 150)

[124] Sekuboyina, A., Valentinitsch, A., Kirschke, J. S., and Menze, B. H. (2017). A Localisation-Segmentation Approach for Multi-label Annotation of Lumbar Vertebrae using Deep Nets. *arXiv Prepr. arXiv1703.04347*. (page 136)

[125] Shankar, S. (2020). Types of machine learning algorithms. `https://medium.com/swlh/types-of-machine-learning-algorithms-62608e83d709`. Accessed: 2020-09-07. (page 9)

[126] Shelhamer, E., Long, J., and Darrell, T. (2017). Fully Convolutional Networks for Semantic Segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(4):640–651. (page 33, 50, 60, 70, 148)

[127] Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., van den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T., and Hassabis, D. (2016). Mastering the Game of Go with Deep Neural Networks and Tree Search. *Nature*, 529(7587):484–489. (page 10)

[128] Simonyan, K. and Zisserman, A. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition. *Int. Conf. Learn. Represent. arXiv1409.1556*. (page 26, 33, 59, 63)

[129] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958. (page 35, 80, 81, 123)

[130] Štern, D., Ebner, T., and Urschler, M. (2016a). From Local to Global Random Regression Forests: Exploring Anatomical Landmark Localization. In *Proc. Med. Image Comput. Comput. Interv.*, pages 221–229. Springer. (page 44, 45, 47, 48, 49, 77, 87, 90, 150)

[131] Štern, D., Kainz, P., Payer, C., and Urschler, M. (2017). Multi-factorial Age Estimation from Skeletal and Dental MRI Volumes. In *Int. Work. Mach. Learn. Med. Imaging*, volume 10541 LNCS, pages 61–69. (page 120)

[132] Štern, D., Likar, B., Pernuš, F., and Vrtovec, T. (2011). Parametric Modelling and Segmentation of Vertebral Bodies in 3D CT and MR Spine Images. *Phys. Med. Biol.*, 56(23):7505–7522. (page 3, 137)

[133] Štern, D., Payer, C., Giuliani, N., and Urschler, M. (2019a). Automatic Age Estimation and Majority Age Classification From Multi-Factorial MRI Data. *IEEE J. Biomed. Heal. Informatics*, 23(4):1392–1403. (page 120)

[134] Štern, D., Payer, C., Lepetit, V., and Urschler, M. (2016b). Automated Age Estimation from Hand MRI Volumes Using Deep Learning. In *Proc. Med. Image Comput. Comput. Interv.*, pages 194–202. Springer. (page 4, 77, 120)

[135] Štern, D., Payer, C., and Urschler, M. (2019b). Automated Age Estimation from MRI Volumes of the Hand. *Med. Image Anal.*, 58:101538. (page 120, 122, 123, 125)

[136] Sun, Y., Wang, X., and Tang, X. (2013). Deep Convolutional Network Cascade for Facial Point Detection. In *Proc. Comput. Vis. Pattern Recognit.*, pages 3476–3483. (page 49, 50, 59)

[137] Szegedy, C., Wei Liu, Yangqing Jia, Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2015). Going Deeper with Convolutions. In *Conf. Comput. Vis. Pattern Recognit.*, pages 1–9. IEEE. (page 26)

[138] Tan, L. K., McLaughlin, R. A., Lim, E., Abdul Aziz, Y. F., and Liew, Y. M. (2018). Fully Automated Segmentation of the Left Ventricle in Cine Cardiac MRI Using Neural Network Regression. *J. Magn. Reson. Imaging*, 48(1):140–152. (page 128)

[139] Tanner, J. M., Whitehouse, R. H., Cameron, N., Marshall, W. A., Healy, M. J. R., and Goldstein, H. (1983). *Assessment of Skeletal Maturity and Prediction of Adult Height (TW2 Method)*. Academic Press, 2nd edition. (page 119, 120)

[140] Tian, Y., Zitnick, C. L., and Narasimhan, S. G. (2012). Exploring the Spatial Hierarchy of Mixture Models for Human Pose Estimation. In *Proc. Eur. Conf. Comput. Vis.*, pages 256–269. (page 44)

[141] Toews, M. and Arbel, T. (2007). A Statistical Parts-based Model of Anatomical Variability. *IEEE Trans. Med. Imaging*, 26(4):497–508. (page 44)

[142] Tompson, J., Jain, A., LeCun, Y., and Bregler, C. (2014). Joint Training of a Convolutional Network and a Graphical Model for Human Pose Estimation. In *Adv. Neural Inf. Process. Syst.*, pages 1799–1807. (page 5, 50, 55, 59, 60, 70, 72, 79, 117, 118, 137, 148, 149, 150)

[143] Tong, Q., Ning, M., Si, W., Liao, X., and Qin, J. (2018). 3D Deeply-Supervised U-Net Based Whole Heart Segmentation. In *Stat. Atlases Comput. Model. Hear. ACDC MMWHS Challenges.*, pages 224–232. Springer International Publishing. (page 134)

[144] Tönnis, D. (1985). Indications and Time Planning for Operative Interventions in Hip Dysplasia in Child and Adulthood. *Z. Orthop. Ihre Grenzgeb.*, 123(4):458–61. (page 4)

[145] Toshev, A. and Szegedy, C. (2014). DeepPose: Human Pose Estimation via Deep Neural Networks. In *Proc. Comput. Vis. Pattern Recognit.*, pages 1653–1660. (page 40, 49, 50, 59, 117)

[146] Ulyanov, D., Vedaldi, A., and Lempitsky, V. (2016). Instance Normalization: The Missing Ingredient for Fast Stylization. *arXiv Prepr. arXiv1607.08022*. (page 27)

[147] Urschler, M., Ebner, T., and Štern, D. (2018). Integrating Geometric Configuration and Appearance Information into a Unified Framework for Anatomical Landmark Localization. *Med. Image Anal.*, 43:23–36. (page 44, 45, 47, 48, 49, 62, 77, 87, 90, 91, 92, 94, 149, 150, 152)

[148] Urschler, M., Zach, C., Ditt, H., and Bischof, H. (2006). Automatic Point Landmark Matching for Regularizing Nonlinear Intensity Registration: Application to Thoracic CT Images. In *Proc. Med. Image Comput. Comput. Interv.*, pages 710–717. (page 3)

[149] Vapnik, V. N. (2000). *The Nature of Statistical Learning Theory*. Springer New York, New York, NY. (page 12)

[150] Viola, P. and Jones, M. J. (2004). Robust Real-Time Face Detection. *Int. J. Comput. Vis.*, 57(2):137–154. (page 47)

[151] Wang, C. and Smedby, Ö. (2018). Automatic Whole Heart Segmentation Using Deep Learning and Shape Context. In *Stat. Atlases Comput. Model. Hear. ACDC MMWHS Challenges.*, pages 242–249. Springer International Publishing. (page 134)

[152] Wang, C. W., Huang, C. T., Lee, J. H., Li, C. H., Chang, S. W., Siao, M. J., Lai, T. M., Ibragimov, B., Vrtovec, T., Ronneberger, O., Fischer, P., Cootes, T. F., and Lindner, C. (2016a). A Benchmark for Comparison of Dental Radiography Analysis Algorithms. *Med. Image Anal.*, 31:63–76. (page 4, 40, 77, 93, 94)

[153] Wang, W. W. J., Xia, C. W., Zhu, F., Zhu, Z. Z., Wang, B., Wang, S. F., Yan Yeung, B. H., Man Lee, S. K., Yiu Cheng, J. C., and Qiu, Y. (2009). Correlation of Risser Sign, Radiographs of Hand and Wrist With the Histological Grade of Iliac Crest Apophysis in Girls With Adolescent Idiopathic Scoliosis. *Spine (Phila. Pa. 1976).*, 34(17):1849–1854. (page 119)

[154] Wang, Y., Yao, J., Roth, H. R., Burns, J. E., and Summers, R. M. (2016b). Multi-atlas Segmentation with Joint Label Fusion of Osteoporotic Vertebral Compression Fractures on CT. In *Comput. Methods Clin. Appl. Spine Imaging*, pages 74–84. (page 137)

[155] Wei, S.-E., Ramakrishna, V., Kanade, T., and Sheikh, Y. (2016). Convolutional Pose Machines. In *Proc. Comput. Vis. Pattern Recognit.*, pages 4724–4732. (page 51, 52, 54, 60, 64, 79, 117)

[156] Werbos, P. J. (1981). Applications of Advances in Nonlinear Sensitivity Analysis. In *Syst. Model. Optim.*, pages 762–770. Springer-Verlag, Berlin/Heidelberg. (page 24)

[157] Wiesler, S. and Ney, H. (2011). A Convergence Analysis of Log-Linear Training. In *Adv. Neural Inf. Process. Syst.*, pages 657–665. (page 26)

[158] World Health Organization (2017). Cardiovascular Diseases (CVDs) : Fact Sheet No. 317. Technical report. (page 128)

[159] Xie, S., Girshick, R., Dollar, P., Tu, Z., and He, K. (2017). Aggregated Residual Transformations for Deep Neural Networks. In *Proc. Comput. Vis. Pattern Recognit.*, pages 5987–5995. IEEE. (page 34)

[160] Yang, D., Xiong, T., Xu, D., Huang, Q., Liu, D., Zhou, S. K., Xu, Z., Park, J. H., Chen, M., Tran, T. D., Chin, S. P., Metaxas, D., and Comaniciu, D. (2017). Automatic Vertebra Labeling in Large-Scale 3D CT Using Deep Image-to-Image Network with

Message Passing and Sparsity Regularization. In *Proc. Inf. Process. Med. Imaging*, pages 633–644. (page 53, 96, 97, 137, 150)

[161] Yang, H. and Patras, I. (2013). Sieving Regression Forest Votes for Facial Feature Detection in the Wild. In *Proc. Int. Conf. Comput. Vis.*, pages 1936–1943. (page 44, 47, 115, 116)

[162] Yang, H., Sun, J., Li, H., Wang, L., and Xu, Z. (2016). Deep Fusion Net for Multi-atlas Segmentation: Application to Cardiac MR Images. In *Proc. Med. Image Comput. Comput. Interv.*, pages 521–528. (page 128)

[163] Yang, X., Bian, C., Yu, L., Ni, D., and Heng, P.-A. (2018a). 3D Convolutional Networks for Fully Automatic Fine-Grained Whole Heart Partition. In *Stat. Atlases Comput. Model. Hear. ACDC MMWHS Challenges.*, pages 181–189. Springer International Publishing. (page 134)

[164] Yang, X., Bian, C., Yu, L., Ni, D., and Heng, P.-A. (2018b). Hybrid Loss Guided Convolutional Networks for Whole Heart Parsing. In *Stat. Atlases Comput. Model. Hear. ACDC MMWHS Challenges.*, pages 215–223. Springer International Publishing. (page 134)

[165] Yaniv, Z., Lowekamp, B. C., Johnson, H. J., and Beare, R. (2018). SimpleITK Image-Analysis Notebooks: a Collaborative Environment for Education and Reproducible Research. *J. Digit. Imaging*, 31(3):290–303. (page 73)

[166] Yu, F. and Koltun, V. (2016). Multi-Scale Context Aggregation by Dilated Convolutions. *Int. Conf. Learn. Represent. arXiv1511.07122*. (page 34)

[167] Zeiler, M. D. and Fergus, R. (2014). Visualizing and Understanding Convolutional Networks. In *Proc. Eur. Conf. Comput. Vis.*, pages 818–833. (page 65, 66)

[168] Zhang, A., Sayre, J. W., Vachon, L., Liu, B. J., and Huang, H. K. (2009). Racial Differences in Growth Patterns of Children Assessed on the Basis of Bone Age. *Radiology*, 250(1):228–235. (page 76, 86)

[169] Zhang, J., Liu, M., and Shen, D. (2017). Detecting Anatomical Landmarks from Limited Medical Imaging Data Using Two-Stage Task-Oriented Deep Neural Networks. *IEEE Trans. Image Process.*, 26(10):4753–4764. (page 40, 53, 59)

[170] Zhou, Y.-T. and Chellappa, R. (1988). Computation of Optical Flow Using a Neural Network. In *IEEE Int. Conf. Neural Networks*, pages 71–78. IEEE. (page 31)

[171] Zhou, Z., Siddiquee, M. M. R., Tajbakhsh, N., and Liang, J. (2020). UNet++: Redesigning Skip Connections to Exploit Multiscale Features in Image Segmentation. *IEEE Trans. Med. Imaging*, 39(6):1856–1867. (page 153)

[172] Zhuang, X. (2013). Challenges and Methodologies of Fully Automatic Whole Heart Segmentation: A Review. *J. Healthc. Eng.*, 4(3):371–408. (page 128)

[173] Zhuang, X., Li, L., Payer, C., Štern, D., Urschler, M., Heinrich, M. P., Oster, J., Wang, C., Smedby, Ö., Bian, C., Yang, X., Heng, P.-A., Mortazi, A., Bagci, U., Yang, G., Sun, C., Galisot, G., Ramel, J.-Y., Brouard, T., Tong, Q., Si, W., Liao, X., Zeng, G., Shi, Z., Zheng, G., Wang, C., MacGillivray, T., Newby, D., Rhode, K., Ourselin, S., Mohiaddin, R., Keegan, J., Firmin, D., and Yang, G. (2019). Evaluation of Algorithms for Multi-Modality Whole Heart Segmentation: An Open-Access Grand Challenge. *Med. Image Anal.*, 58:101537. (page 128, 131, 134)

[174] Zhuang, X. and Shen, J. (2016). Multi-Scale Patch and Multi-Modality Atlases for Whole Heart Segmentation of MRI. *Med. Image Anal.*, 31:77–87. (page 128)

I'll be back.

T-800 (⋆2026 – †1984)