UNIVERSITÄT
KOBLENZ · LANDAU
Fachbereich 4: Informatik

Analysis of medical images using deep learning

Masterarbeit
zur Erlangung des Grades
MASTER OF SCIENCE
im Studiengang Web Science

vorgelegt von

Almat Utegulov

Betreuer: Dr. Sabine Bauer, Institut für Computervisualistik, Fachbereich Informatik, Universität Koblenz-Landau
Erstgutachter: Prof. Dr.-Ing. Dietrich Paulus, Institut für Computervisualistik, Fachbereich Informatik, Universität Koblenz-Landau
Zweitgutachter: Dr. Sabine Bauer, Institut für Computervisualistik, Fachbereich Informatik, Universität Koblenz-Landau

Koblenz, im Februar 2020

Abstract

Since the invention of U-net architecture in 2015, convolutional networks based on its encoder-decoder approach significantly improved results in image analysis challenges. It has been proven that such architectures can also be successfully applied in different domains by winning numerous championships in recent years. Also, the transfer learning technique created an opportunity to push state-of-the-art benchmarks to a higher level. Using this approach is beneficial for the medical domain, as collecting datasets is generally a difficult and expensive process.

In this thesis, we address the task of semantic segmentation with Deep Learning and make three main contributions and release experimental results that have practical value for medical imaging.

First, we evaluate the performance of four neural network architectures on the dataset of the cervical spine MRI scans. Second, we use transfer learning from models trained on the Imagenet dataset and compare it to randomly initialized networks. Third, we evaluate models trained on the bias field corrected and raw MRI data. All code to reproduce results is publicly available online.

Erklärung

Ich versichere, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe und dass die Arbeit in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegen hat und von dieser als Teil einer Prüfungsleistung angenommen wurde. Alle Ausführungen, die wörtlich oder sinngemäß übernommen wurden, sind als solche gekennzeichnet.

Die Vereinbarung der Arbeitsgruppe für Studien- und Abschlussarbeiten habe ich gelesen und anerkannt, insbesondere die Regelung des Nutzungsrechts.

Mit der Einstellung dieser Arbeit in die Bibliothek bin ich einverstanden. ja ☐ nein ☐

Der Veröffentlichung dieser Arbeit im Internet stimme ich zu. ja ☐ nein ☐

Koblenz, den 13th February 2020

Contents

Chapter 1

Introduction

1.1 Problem statement

During the last decades, noninvasive medical imaging technologies became widely used to diagnose diseases, create individual prostheses, intervention planning, and other domains. Processing such images made possible to analyze and study human anatomy, functions without damaging patient tissues. But currently analyzing images is a costly and time-consuming process because of humans involved in the processes.

For example in the case of magnetic resonance images (MRI) segmentation. Processing single brain volume can take several days for a trained expert and thousands of dollars. To optimize this process researches made significant efforts to automate image understating task. Results in this area opened an opportunity to reduce medical expert input from days to hours and therefore dramatically improve the performance of medical experts and make it more affordable to patients.

An important factor for using MRI is that it doesn't emit ionizing radiation as X-ray and computerized tomography (CT) does. Also, this method can better capture soft tissues and see bone tumor and metastases, which cannot be done X-ray and CT [HSST18].

1.2 Research question

In this thesis, I study neural network techniques in the context of image analysis and evaluate vision-based algorithms capable of locating, identifying, and segmenting vertebrae in MR scans. I consider extraction of the spine from the 3-dimensional volume, as a semantic image segmentation problem representing each MRI in the form of 2-dimensional slices for sagittal, coronal and axial planes.

Dataset used in this work is provided by a VisSim research group, it has 14 subjects with corresponding manual vertebral bodies segmentation. As machine learning

algorithms like neural networks generalize better with more training data, different augmentation [HGK18] techniques were used to generate synthetic images of the cervical spine.

There are five main research questions covered in this thesis:

- How various segmentation neural network architectures perform on a dataset of cervical spine scans?
- What would be the influence of MRI artifacts correction on models performance?
- What would be a performance of those networks if with random initialization [HZRS15]?
- How valuable is transfer learning from models trained on the ImageNet dataset of natural images to medical image segmentation?
- How models will perform when trained on sagittal, coronal, axial views?

‘

1.3 Thesis structure

The thesis is structured as follows.

In chapter 2, you will find a general overview of neural networks and its applications specific to image analysis.

In chapter 3 covers the theoretical background of deep learning, how they extract knowledge from data, the definition of transfer learning and details about modern architectures for image segmentation used in this thesis and NN created by me to solve this task.

In chapter 4 you will find details about experimental setup, used preprocessing and data augmentations. Models training detail such as used hardware, hyper-parameters, and results of conducted experiments, models prediction visualization to give better intuition behind numbers. Additionally, I cover edge cases, such as best/worst predictions and unexpected behavior on networks are also reported. A discussion of limitations finishes this chapter.

And finally in part 5 of the thesis I summarize conducted work and discuss ideas for future work.

Chapter 2

Previous work

This chapter will cover previous works that have been conducted in areas that are related to the analysis of medical images using deep neural networks. In the first part, we cover what research was carried out in a field of image segmentation. In the second part, we focus on recent papers and challenges specific to the medical imaging domain. In the last section, we research progress specific to MRI analysis.

2.1 Image analysis

Image analysis can be divided into major categories, here are some examples to give a better intuition behind the task solved in this work. On Figure 2.1 you can see images from COCO [LMB+14] and PASCAL VOC [EVGW+10] dataset. These datasets are used in image analysis competition to compare new approaches in a task like classification, object detection, and segmentation.

Classification challenge goal is to classify what is shown on an image, here you generally have a single object on the foreground which falls to one or more of possible classes, e.g. picture of a car near the building will fall into "car" and "building" classes.

Figure 2.1: COCO [LMB+14] and PASCAL VOC [EVGW+10] dataset samples.

Object detection also knows as classification with localization, is a more sophisticated version of the classification task, here the algorithm also needs to draw bounding boxes around in each classified object on the image.

The next challenge by a level of complexity is semantic segmentation task. Here, in addition, to correctly classifying in each object on the image, each pixel should be assigned to its particular class, in other words, it is pixel-wise classification. For example on the picture of a room in Figure 2.1 walls are given a different class from sofa and window, also import detail is that both sofas are related to the assigned the same class.

And logical next step would be to create masks for each individual object in the image, and this is solved in the "instance segmentation" task. The result is seen in the picture with sheep, where each of them has a separate mask, but at the same time, they relate to a single class.

2.2 Semantic segmentation

Task solved in this work is semantic segmentation for the medical imaging domain and we use a deep learning approach to handle it. In the last years, there was a lot of research happening in this area, the latest breakthrough in the segmentation domain was creating U-net architecture [RFB15] with its encoder-decoder model, which will be covered in Chapter 3.5.1.

2.3 Segmentation of vertebral bodies

Most of the research in the segmentation of vertebral bodies was conducted on 2D images [HCLN09] and a few works with 3D volumes. The main focus of our research is on 2D slices.

Zukić et al. [ZVE+14] presented a method to detect and segment vertebral bodies with minimal user input. Their approach is to use a Viola-Jones algorithm to detect vertebral centers and segment vertebrae in parallel as a second step. For training, they used 26 lumbar datasets containing 234 reference vertebrae and achieved average Dice Similarity Coefficient (DSC) to a manual reference of 79.3%.

Hille et. al [HSST18] reported an approach to segment 3D volumes with minimal user assistance. Their first step is to apply bias field correction to deal with MR artifacts, and after that use appearance-based, VB probability maps to guide segmentation. For training they had MR scans from 48 subjects and 63 more for evaluation, which had 419 vertebral bodies. Performance achieved by Dice overlap similarity is 86.0% and mean Euclidean surface distance error of 1.59 $\pm$ 0.24 mm and a Hausdorff distance of 6.86 mm.

Chu et. al. [CBA+15] used the random forest to get a region of interest (ROI) of vertebral bodies for a subsequent segmentation step on the 3D image. After that results were combined with a learned probability map to segment each vertebral body by a threshold. Their approach achieved an overall Dice similarity of 88.7%.

Athertya et al. [AK+16] created fuzzy C-means clustering for the segmentation of VBs on T1-weights MR scans, combined with post-processing they achieved DSC of 86.7%. Their dataset had 16 subjects.

Gaonkar et. al. in [GXV+17] presented superpixels based multi-parameter ensemble for lumbar spine segmentation. It was followed by morphological post-processing to increase overlap score and resulted in mean DSC of 83%. Their dataset had in 48 sagittal T2 and 15 T1-weighted MR scans, with a spatial resolution of 0.34×0.34 to 1.1×1.1 mm and slice thickness was between 0.5 to 5.0 mm.

Korez et. al [KLPV16] proposed an automated method to segment VBs using a combination of designed 3D CNN architecture which guides deformable models towards VB boundaries. For training, they used 61 VBs from 3D MR spine images of 23 subjects and reported an average of 93.4 ± 1.7% DSC.

Neubert et al. [NFE+12] achieved a Dice overlap of 91% using statistical shape analysis and registration of grey level intensity profiles. Their automated approach was used to extract lumbar and thoracic intervertebral discs and vertebral bodies into three-dimensional segmentation.

Chapter 3

Theoretical background

In this chapter, we walk through the background information required for a better thesis understanding. Starting from a cervical spine will with neural networks, computer vision tasks, finishing with the transfer learning approach and modern architectures used for semantic segmentation.

3.1 Cervical spine

For model training we use the dataset of Magnetic Resonance (MR) images of human cervical spine. This is a upper section of spinal cord and divided into 3 regions. Upped has C_0, C_1, C_2 vertebral bodies, middle has C_3, C_4, C_5 and the lower cervical is C_6, C_7. Location of these VBs is shown on the Figure 3.1, here C_0 is connected to the head, followed by C_1 and so on [PW90].

Also we use the coordinate system shown in Figure 3.2 to refer MR volume dimensions. Those three planes views are called *sagittal, coronal* and *axial* in next chapters.

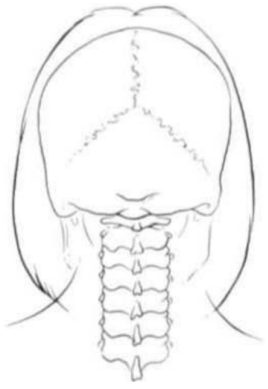


Figure 3.1: Cervical spine location [PW90].

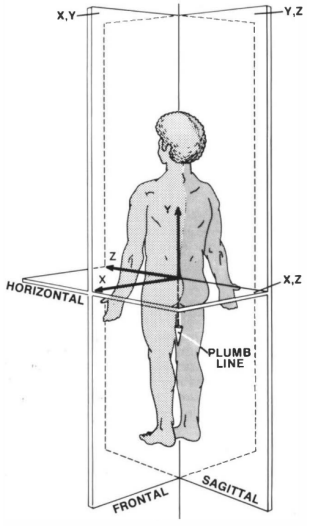


Figure 3.2: Coordinate system used in medical imaging [PW90]. Horizontal is also known as *axial* or *traverse* plane. Frontal is also known as *coronal* plane.

3.2 Convolutional neural networks

Convolutional Neural Networks (CNNs/ConvNets) assume that input is an image. Main building blocks of ConvNets are *convolutional*, *pooling/upsampling* and *normalization* layers. They receive different parameters and stack using various connection approaches to form an NN architecture. To score created network performance against the target value, they have a loss function in the last layer (e.g. softmax, sigmoid).

3.2.1 Convolutional layer

Convolutional layer has 3 dimensions: height (h), width (w) and depth (d). Depth, in this case, is a number of input layer channels. For RGB it is $h \times w \times 3$, for gray-scale image it is $h \times w \times 1$. It should not be confused with the full depth of the neural network, which is a total number of learnable layers in the network.

Note that convolutional layers are made of neurons that have weights and biases trainable with gradient descent. They do most of the computations happening in CNNs. On the other hand pooling/upsampling and normalization layers are fixed functions.

In Figure 3.3 you can see a learning unit of the convolutional layer which is called **kernel**. Each layer consists of K kernels (filters), generally, with a small receptive field, $n \times m \times d$. For example typical kernel for gray-scale image input has shape of $3 \times 3 \times 1$. Each filter is convolved through the input, performs a dot product optionally followed with a non-linearity (e.g. ReLU), and forms a 2-dimensional matrix called **activation map**. These K maps stack together and serve as an input to the next layers which should have $n \times m \times K$ dimensional filters. This operation is significantly more efficient than for regular fully-connected layers because convolutional kernels are connected to a small region of the layer before it, rather than to all neurons [Kar].

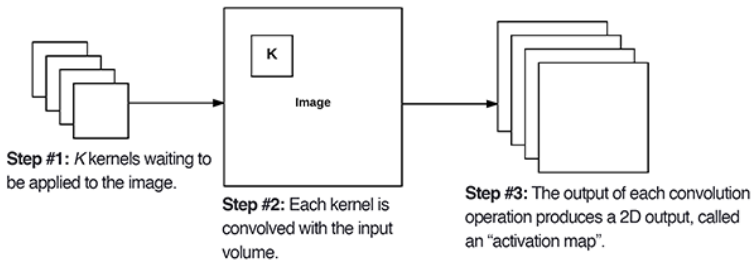


Figure 3.3: Application of learnable convolutional kernels to the input.

3.2.2 Pooling layer

Pooling layers are used in ConvNets to reduce the number of parameters in the network and control overfitting. It has 2 hyper-parameters – *kernel size*, *stride (step)* and

transformation function, which can be $max, average, max + average, etc.$. The most common form, which is used in CNN architectures described later, is 2×2 kernel with stride 2, it independently resizes channels inside of the input using max function. Pooling layer with such hyper-parameters accepts input of $h \times w \times d$ dimensions and returns an activation map with $h/2 \times w/2 \times d$ shape. As you can see the number of channels does not change. Max pooling operation is shown in Figure 3.4.

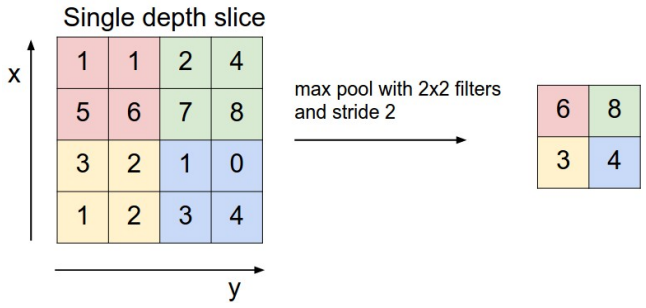


Figure 3.4: Max pooling operation with 2×2 filter and stride 2 discards 75% of input information [Kar].

The upsampling layer makes a reverse transformation of the pooling operation. It accepts *factor* as a hyper-parameter, commonly its value is 2. In this case $h \times w \times d$ dimensional input will result in $h * 2 \times w * 2 \times d$ output. New matrix elements are commonly filled with corresponding pixel values or zeros.

3.3 Transfer learning

Usually, people don't train the entire CNN from scratch, because it needs a lot of data to reach top performance. As it is a rare case to have that much data, especially in the medical domain, a common practice is to reuse weights from models pre-trained on large datasets, such as ImageNet (1.2 million images) as an initialization or feature extractor for another task – this approach is called transfer learning. It can be used in different ways [Kar]:

The first strategy is to treat the convolutional network as *fixed feature extractor* – useful when original and target datasets are similar or share the same classes. For example original dataset with trees and flower images, and target plant photos made with another camera/light conditions/new flower types/etc. In this case, you need to substitute the last layers (one or more) with the desired "head" and treat the rest of CNN as a fixed feature extractor. Network, in this case, will train only new layers on the target dataset.

The second strategy is to use pre-trained weights as an initialization, substitute network head as in the previous case and continue training on a new dataset with back-propagation – this is called *fine-tuning*. It is useful when original and target datasets are not similar but can share some primitive shapes, colors, and textures. An example could be images of nature and fashion designer clothes. As this our case, we investigate if fine-tuning from natural images will work for the MRI scans dataset. We will use two models trained on ImageNet for fine-tuning:

- Inception v4 [SIVA17],
- ResNet-50 [HZRS16].

Recent research from [RZKB19] raised a question about the usefulness of models trained on natural image datasets in transfer learning to the medical domain. They demonstrate that deep neural networks architectures winning competitions in object detection and classification are over-fitted to this task and have little value for the medical domain and models with a fraction of their parameters can achieve similar performance while trained from scratch.

3.4 Overlap-based metric

The agreement between ground-truth segmentation and model prediction is evaluated using the Dice Similarity Coefficient (DSC) [Dic45], in terms of F_1-*score*. It is employed on each batch of m slices and ranges from 0 (no segmentation overlap) to 1 (perfect agreement):

$$F_1 = \frac{1}{m}\sum_{i=1}^{m}\left(\frac{2TP}{2TP+FP+FN}\right)_i, \tag{3.1}$$

representing DSC as follows:

$$DSC = \frac{2TP}{2TP+FP+FN}, \tag{3.2}$$

taking into calculation false-positive (FP), false-negative (FN), true-positive (TP) and true-negative (TN) predictions. Alternative way of presenting DSC is as follows:

$$DSC = \frac{2\mathbf{N}(A \cap B)}{N(A)+N(B)}. \tag{3.3}$$

Here A – ground truth segmentation, B – model under evaluation, N(A) and N(B) – a number of pixels obtained by each technique, A∩B – the pixel-wise intersection between structures.

3.5 Architectures

In this chapter, we describe CNN architectures for semantic segmentation which were used for evaluation on our dataset of MR scans. These models achieved top results in various challenges for different dataset types such as natural, medical and satellite images. Besides four approaches adopted from other researches, we designed own convolutional network for segmentation.

3.5.1 U-net

First model under evaluation is U-net [RFB15], shown in Figure 3.5. After its creation in 2015 many winning solutions in segmentation challenges [**?**] used its "encoder-decoder" approach.

Encoder part of the model, which extracts features and captures context, and symmetric decoder part which upsamples the resulting mask for the final prediction and enables precise localization. The concatenation of encoder and decoder convolutions of the same shape improves the gradient flow and passes features from lower layers to adjust prediction details, which are lost during the contraction phase.

Authors also used strong augmentations to increase the dataset size and reached first place in ISBI challenge for the segmentation of neuronal structures in electron microscopic stacks.

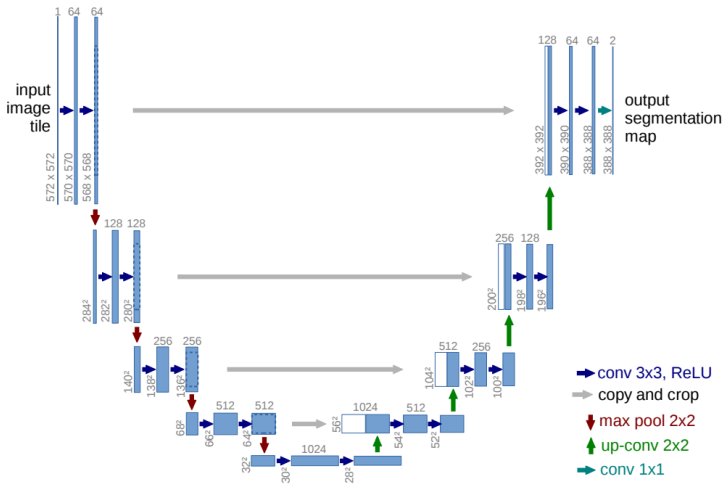


Figure 3.5: U-net architecture. Blue box – means feature map, and number on top of each box is the number of channels. Feature map x-y dimensions are provided at the lower-left edge of the box. White boxes represent copied feature maps. The arrows denote the different operations [RFB15].

3.5.2 Pyramid scene parsing network

The second model for evaluation is the Pyramid Scene Parsing Network (PSPNet) [ZSQ+17], illustrated in Figure 3.6. This architecture achieved state-of-the-art results on ImageNet scene parsing challenge 2016 [RDS+15], PASCAL VOC 2012 [EVGW+10] and Cityscapes [COR+16] benchmarks.

Authors introduced *pyramid pooling module* which was empirically proven to be an effective global contextual prior and addresses the issue of not sufficiently incorporated global scene information in the last layers [ZZP+19]. This module joins features of different scales and sub-regions to makes them accessible in the last layer for pixel-wise classification.

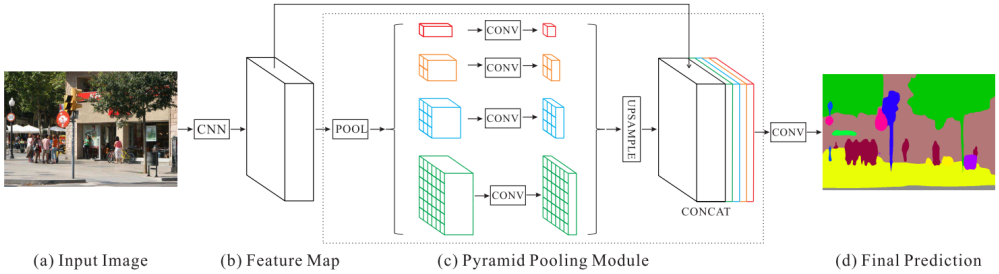


Figure 3.6: Overview of PSPNet. (a) – input image is supplied to CNN, (b) – arbitrary CNN, in this case, authors used FCN [LSD15], extract last convolutional layer activation maps, (c) – pyramid parsing module collects different sub-region representations, Final feature maps are formed by upsampling and concatenation pyramid layers. (d) – stacked feature maps are passed to the convolutional layer to get final segmentation prediction [ZSQ+17].

3.5.3 Linknet

Third evaluation candidate is Linknet [CC17]. Authors proposed architecture, shown on Figure 3.7, which is light weight and efficient at the same time. They achieved state-of-the-art performance on CamVid [BFC09] and comparable results on Cityscapes [COR+16] datasets.

They proposed to bypass spacial information directly from encoder to decoder, such a way of improving segmentation quality and processing speed. Using their approach improved over the issue of information loss during pooling or strided convolution operations in the encoder. Also, usage of indices passing directly to the decoder was thus making the network keep excessive parameters.

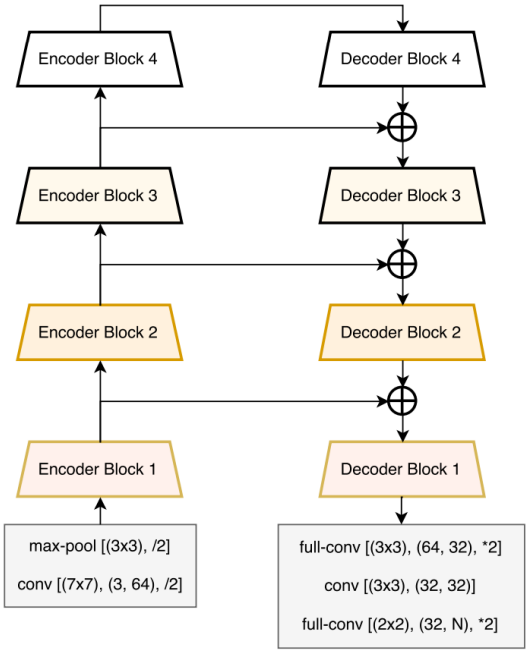


Figure 3.7: Linknet architecture [CC17], Left half of the network is the encoder, right is the decoder. Here, *conv* – means convolution, *full-conv* – denotes full-convolution [LSD15], */2* – down-sampling by a factor of 2, and $*2$ means up-sampling by a factor of 2. Each *conv* layer is followed by standard combination of *batch normalization* + *ReLU activation*.

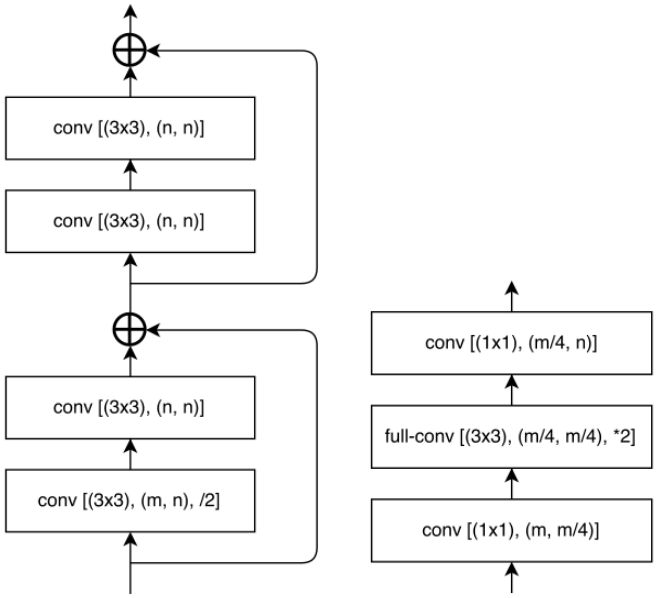


Table 3.1: Linknet encoder block (left) and decoder block (right) [CC17].

In Table 3.1 you can see the structure of Linknet encoder and decoder blocks. The novelty of this architecture is in a way those building blocks are connected. As during encoding phase spatial information, if lost, authors passed each encoder input to corresponding decoder output, such a way, recovering lost spatial details and improving the results of the upsampling step.

3.5.4 Feature pyramid network

Forth segmentation model under evaluation is Feature Pyramid Network (FPN) [LDG+17]. Authors constructed feature pyramids have strong semantics at all scales using a pyramidal hierarchy of CNNs with a little additional computational cost. Their approach combined with Faster R-CNN detector [RHGS15] achieved a state-of-the-art single-model result on the COCO detection benchmark [LMB+14].

FPN architecture is shown in Figure 3.8. Here "bottom-up" path is connected to "top-down" with additional skip connections. It combines low-resolution, semantically strong features with high-resolution, semantically weak features.

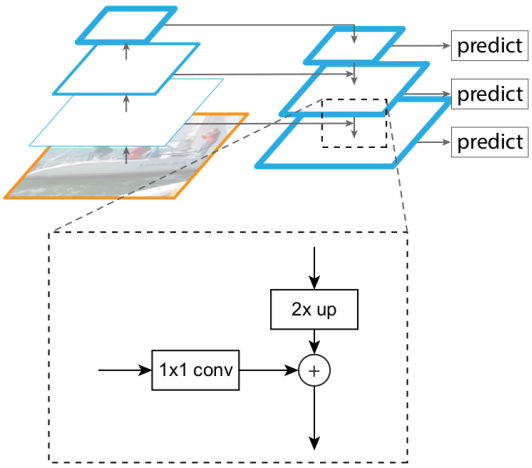


Figure 3.8: Feature Pyramid Network building block illustrating the skip connection and the top-down pathway, merged by addition [LDG+17].

As the network is fully convolutional, it accepts an image of arbitrary size as input and produces feature maps scaled -down/-up by a factor of 2 on each pyramid level.

"Bottom-up" path on the left half of the Figure 3.8, makes feed-forward computation of backbone (encoder) architecture. As encoder networks generally have multiple stages of convolutional layers that produce feature maps of the same size, FPN only takes the last layer of each stage. The motivation behind it is that deeper layers in the network have the strongest features.

"Top-down" pathway with lateral connections is on the right half of the network. It scales feature maps back to the original size on each stage and skip-connections enrich activation maps with features from the bottom-up path that have the same size.

3.5.5 Custom convolutional networks

In this section, handcrafted convolutional networks created to compete with stated previously are described. We use *Convolution – Batch normalization – ReLU* combination as a building block for these networks. We divided them into 3 versions, here *v1* is an early experiment in creating own segmentation architecture.

CNN version	Architecture
v1	(conv64-bn-relu) (conv128-bn-relu) (conv128-1) sigmoid
v2	(conv32-bn-relu) (conv64-bn-relu) (conv128-bn-relu) (conv256-bn-relu) (conv512-bn-relu) (conv512x1) sigmoid
v3 Large32	(conv32-bn-relu) maxpool (conv64-bn-relu) maxpool (conv128-bn-relu) maxpool (conv256-bn-relu) maxpool (conv512-bn-relu) maxpool (conv256-bn-relu) upsample (conv256-bn-relu) upsample (conv128-bn-relu) (conv64-bn-relu) upsample (conv32-bn-relu) sigmoid
v3 Large64	(conv64-bn-relu) maxpool (conv128-bn-relu) maxpool (conv256-bn-relu) maxpool (conv512-bn-relu) maxpool (conv512-bn-relu) upsample (conv256-bn-relu) upsample (conv128-bn-relu) upsample (conv64-bn-relu) sigmoid
v3 Small32	(conv32-bn-relu) maxpool (conv64-bn-relu) maxpool (conv128-bn-relu) maxpool (conv256-bn-relu) maxpool (conv256-bn-relu) upsample (conv128-bn-relu) upsample (conv64-bn-relu) upsample (conv32-bn-relu) sigmoid
v3 Small64	(conv64-bn-relu) maxpool (conv128-bn-relu) maxpool (conv256-bn-relu) maxpool (conv512-bn-relu) maxpool (conv512-bn-relu) upsample (conv256-bn-relu) upsample (conv128-bn-relu) upsample (conv64-bn-relu) sigmoid

Table 3.2: Custom CNNs details.

Looking forward to results, best DSC achieved by the custom convolutional network was 64% for sagittal view, considering that other models perform similar or worse on other views, we decided to postpone further development and focus more on established approaches.

Chapter 4

Experiments

4.1 Experimental setup

In this chapter we start with section 4.1.1 describing implementation details, such as programming language, framework, and packages used for training CNNs, etc. After that in section 4.1.2 we cover the size and parameters of the dataset and overview augmentations applied to artificially increase its size. Section 4.1.3 is about dataset preprocessing and covers various normalization approaches we adopted for this task. Section 4.1.4 describes image transformations used during models training. After that in section 4.2 you will find a description of how previous sections are combined into neural networks training pipeline and which hyper-parameters were used for experiments. Final results for NN models are listed and visualized in section 4.3. And this chapter ends on part 4.4 where we discuss limitations of achieved results.

4.1.1 Frameworks and libraries

As a framework to define and optimize neural networks we used Pytorch [PGM+19], which is in an open-source project backed by Facebook and available to the public free of charge under a BSD license. To handle routine training operations we extended python library Segmentation models [Yak19] to be able to work with medical images and custom models. As an augmentation framework, we used Albumentation [BPK+18] package which has many implemented image transformations. For MR image N4 bias field correction [TAC+10] we used ITK [JMIC13] library. Python 3.6 [VRDJ95] is used as programming language, and many scientific operations where handled with NumPy [vCV11] package. Code used for training and final models weights reported in this thesis are available online[1].

[1] https://github.com/utegulovalmat/cervical-spine-segmentation

4.1.2 Dataset details

The dataset used for experiments was provided by the VisSim research group. It has 14 unique patient studies of the cervical spine. Subjects were split at random into 3 groups: 12 of them were for a train set, 1 for the validation and 1 for the test.

Each scan has 7 segmented VBs, an average volume of $512 \times 512 \times 235$ slices and image pixel spacing of $0.35mm \times 0.35mm \times 0.7mm$, resulting in 2686 slices with segmented VB for sagittal, 2276 for coronal and 2130 for axial planes. Scan details of study subjects are stated in Table 4.1.

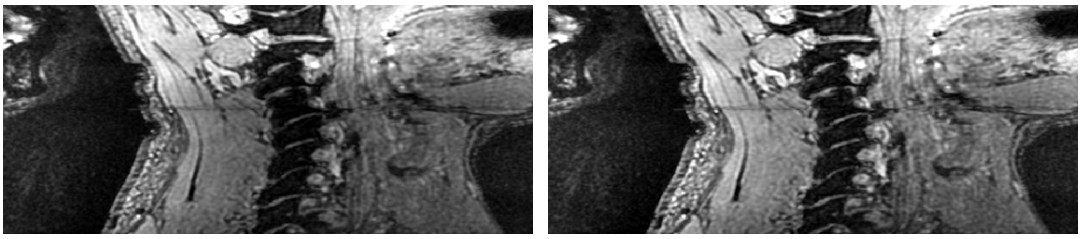
Patient	Image dimensions	Image spacing, mm	Used for
D0030100301	512×512×232	0.35×0.35×0.7	train
D0040100402	512×512×174	0.35×0.35×0.9	train
D0040100403	512×512×216	0.35×0.35×0.8	train
D0060100602	512×512×228	0.35×0.35×0.7	train
D0060100702	512×512×248	0.35×0.35×0.7	train
D0060100802	512×512×228	0.35×0.35×0.7	train
D0060100902	512×512×248	0.35×0.35×0.7	train
D0060101002	512×512×248	0.35×0.35×0.7	train
D0060101202	512×512×278	0.35×0.35×0.7	train
D0060101402	512×512×228	0.35×0.35×0.7	train
D0060101502	512×512×244	0.35×0.35×0.7	train
D0060101902	512×512×236	0.35×0.35×0.7	train
D0060102002	512×512×248	0.35×0.35×0.7	validation
D0060102102	512×512×238	0.35×0.35×0.7	test

Table 4.1: Characterization of all datasets used in experiments.

4.1.3 Preprocessing

Dataset scans have individual pixel values varied between [0; 3600] for different types of tissue, where 0 represented background. To standardize data we used preprocessing pipeline before supplying images to model training.

In the first step, we used N4 bias-field correction [TAC+10] algorithm to handle non-uniformity within each scan, as it showed improvements in segmentation tasks for VBs [HSST18] and other domain e.g. brain lesion segmentation [TAC+10]. As using such preprocessing will reduce the information available in the volume, we also trained networks on data without this normalization and measure its value. In Figure 4.1 you can see a sample slice before and after N4 bias field correction.



(a) Raw slice **(b)** N4 corrected slice

Figure 4.1: Example slice before (a) and after (b) N4 bias field correction.

The second step is normalization, which refers to normalizing the data dimensions so that they are of approximately the same scale [Kar]. We applied normalized the 3D scans to zero mean and unit variance using, Z-score normalization, with the mean value taken as average pixel intensity of segmented vertebral bodies.

In equations 4.1 and 4.2, Z-score normalization uses intensities inside the VB masks – M, for the MR image – I, to determine the mean – μ, and standard deviation – σ, that is:

$$\mu = \frac{1}{|M|} \sum_{\mathbf{m} \in M} I(\mathbf{m}), \tag{4.1}$$

and:

$$\sigma = \sqrt{\frac{\sum_{\mathbf{b} \in M} (I(\mathbf{m}) - \mu)^2}{|M| - 1}}. \tag{4.2}$$

Then the Z-score normalized image is defined using the following equation 4.3:

$$I_{z-\text{ score}}(\mathbf{x}) = \frac{I(\mathbf{x}) - \mu}{\sigma}. \tag{4.3}$$

Third step, result were supplied to linear transformation to normalize data in range $[0; 1]$ and represent slices as standard gray-scale images. In equation 4.4 the normalized

image is defined as $\hat{I}$, $\min(I)$ is minimal intensity of pixel in the volume, and $\max(I)$ maximal pixel value:

$$\hat{I} = \frac{I - \min(I)}{\max(I) - \min(I)}. \tag{4.4}$$

The last step was to pad 2D slices with 0s to 512×512 pixels height and width, to standardize the neural network input layer.

4.1.4 Augmentation

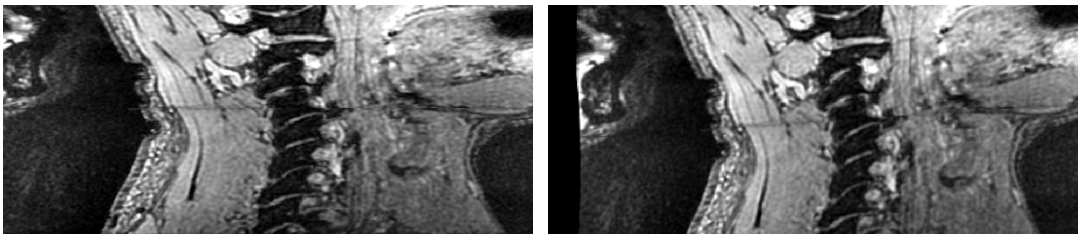
Augmentation is a process of changing some image parameters while keeping it recognizable. For example in natural images, changing the color of a car will not change the image class, it will stay car. This technique is used to artificially increase the number of dataset samples and is especially useful for the medical image analysis domain, because of the difficult and expensive data collection process.

As the used dataset has only 14 subjects and in order to train models generalize better, in this work we used various augmentation approaches. The list of transformations and probability of applying them individually are listed in Table 4.2.

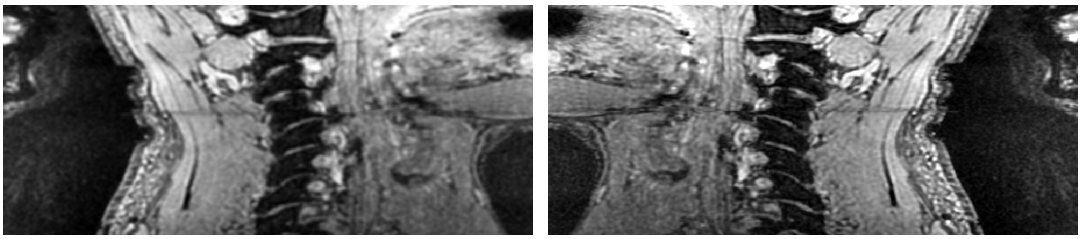
Transformation	**Hyper-parameters**	**Probability**
Horizontal flip		0.5
Blur	3px kernel	0.5
Gaussian blur	3px kernel	0.5
Motion blur	3px kernel	0.5
Gaussian noise	0.01	0.5
Elastic transformation [SSP03]		0.5
Random rotation	$\pm 20^{\circ}$	0.5
Random scale	$\pm 10\%$	0.5
Random shift	up tp 32px	0.5

Table 4.2: Probability of applying individual augmentations.

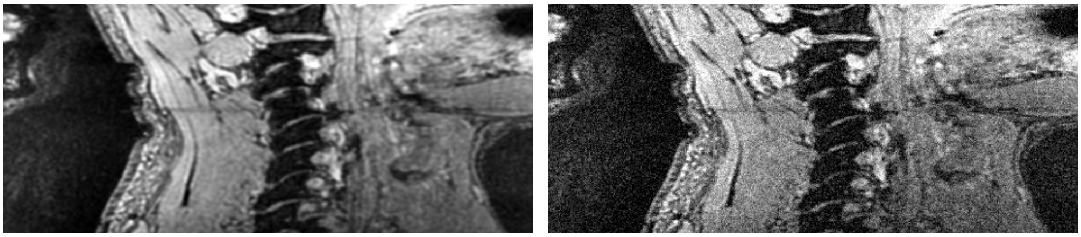
For augmentations we used implementation from [BPK+18]. In Figure 4.2 you can see example transformations applied to the dataset during training phase.



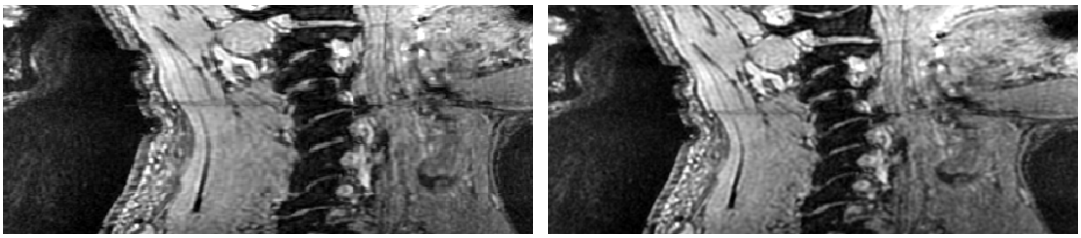
(a) Original image **(b)** Elastic transformation



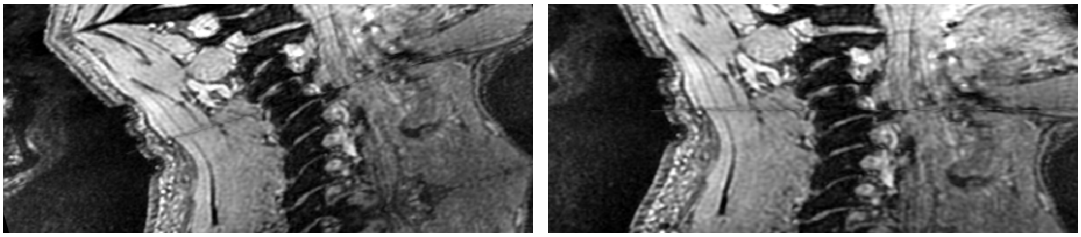
(c) Blur **(d)** Horizontal flip



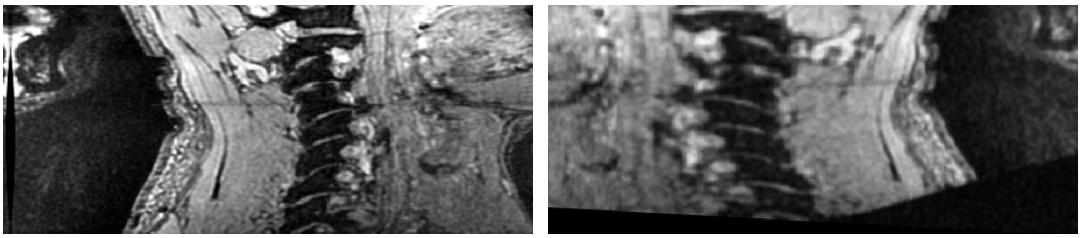
(e) Gaussian blur **(f)** Gaussian noise



(g) Grid distortion **(h)** Motion blur



(i) Random rotation **(j)** Random scale



(k) Random shift **(l)** All transformations at once

Figure 4.2: Transformations applied to artificially increase dataset size.

4.2 Training

In this chapter, the workflow and experimental setup used to train neural networks is described in details.

In Figure 4.3 you can see the training process applied to neural networks. Hyper-parameters that were used to train them were set as follows.

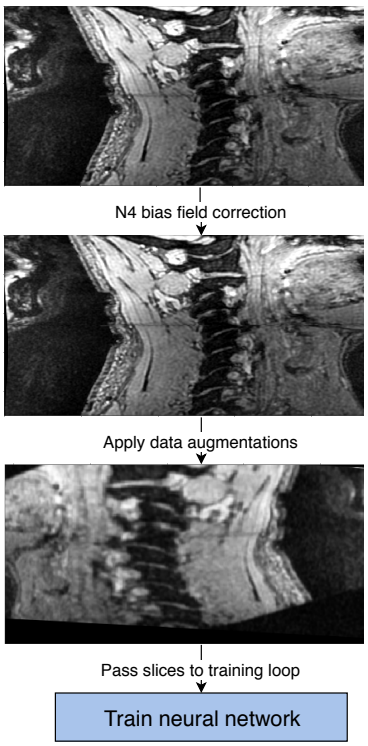


Figure 4.3: Training pipeline.

To train networks, GPU from Nvidia GeForce GTX 1080 was used, with driver version 418.56, CUDA 10.1 and 8199 MB of memory.

The batch size used for all models training was 8 images to fit it into GPU memory. For CNN optimization Adam optimizer was used with a learning rate of $1e-4$. In general, architectures were trained around 12-15 epochs until no improvement happens in validation set loss. The average training time for single models was around 2.5 hours.

In Figure 4.4 you can that training metrics (blue line) monotonically decreasing to 0, it means that the model is capable to learn data representation from training examples and minimize the loss function, in other words, it can memorize the dataset if trained long enough.

Clearly, such an overfitted model cannot make good predictions on unseen data, that is why after each training epoch it is evaluated against the validation set, in order to

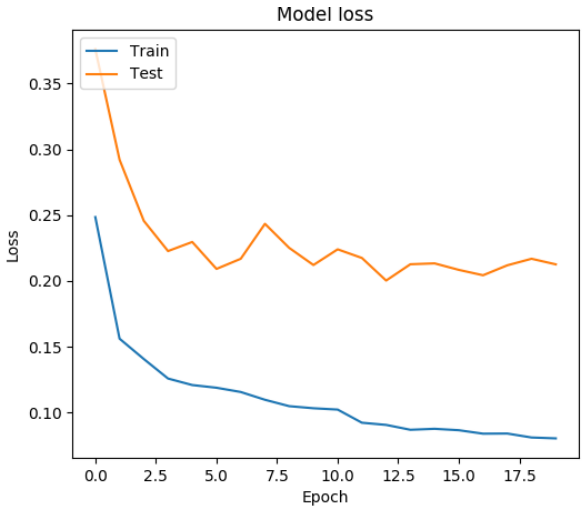


Figure 4.4: Loss for train and validation datasets during model training process.

estimate performance on new data. If validation metrics have now improvements for 7 epochs further training was stopping to reduce GPU utilization time.

Also on the graph, we can see that validation set loss follows training with a consistent gap, it represents that the trained model can generalize on unseen data. The reason behind it may be the usage of augmentations to supply new training samples on each epoch.

4.3 Results

In this section, we showcase results achieved during experiments with visualization of performance metrics for better intuition.

Model	Weights	Corrected	Sagittal, %	Coronal, %	Axial, %
U-Net	Random	Yes	75	77	74
U-Net	Random	No	84	76	74
U-Net	ResNet-50	Yes	**87**	79	77
U-Net	ResNet-50	No	84	77	77
U-Net	Inception-v4	Yes	79	81	79
U-Net	Inception-v4	No	80	75	79
FPN	Random	Yes	83	71	70
FPN	Random	No	83	70	71
FPN	ResNet-50	Yes	86	84	78
FPN	ResNet-50	No	80	75	**80**
FPN	Inception-v4	Yes	79	**88**	74
FPN	Inception-v4	No	79	83	73
Linknet	Random	Yes	83	75	74
Linknet	Random	No	84	77	79
Linknet	ResNet-50	Yes	86	86	73
Linknet	ResNet-50	No	86	80	73
Linknet	Inception-v4	Yes	76	83	73
Linknet	Inception-v4	No	82	78	78
PSPNet	Random	Yes	79	74	72
PSPNet	Random	No	82	77	74
PSPNet	ResNet-50	Yes	80	78	72
PSPNet	ResNet-50	No	74	76	72
PSPNet	Inception-v4	Yes	78	82	75
PSPNet	Inception-v4	No	74	78	77

Table 4.3: Performance of models trained and evaluated on same planes. Model – backbone architecture name in the first column. Weights – convolutional kernels used for initialization. N4 correction – used bias field corrected MRI. Sagittal, coronal, axial – DSC for models train on given planes, where 0 is no overlap of prediction with ground truth segmentation, and 100 is a perfect overlap.

In Table 4.3 you can see in the first column CNN architecture used for evaluation. Second, it has a convolutional kernel initialization approach used, such as transfer learning with ResNet-50 and Inception-v4 models or random weights from Kaim-

ing [HZRS15]. As you can see all models surpassed 70% overlap and the best score achieved is 88%.

As shown in Table 4.4 overall models perform best on the sagittal plane with mean DSC of 81±3.8%. Results for other views are a bit lower, for coronal 78±4.4% and for axial 75±2.9%.

Plane	**DSC, %**
Sagittal	81±3.8
Coronal	78±4.4
Axial	75±2.9

Table 4.4: Performance of models for each plane.

Taking model trained on corrected images DSC results for sagittal 81±4.0%, coronal 80±5.1%, and axial 74±2.6%. Compared to models trained on MRI scans without preprocessing gives mean DSC for sagittal 81±3.8%, coronal 77±3.1%, axial 76±3.1% as shown in Table 4.5.

As you can see training on unprocessed images results in better predictions, only models trained on corrected coronal plane images did show slightly better results. So the hypothesis that MRI artifacts correction will have a positive impact on model performance has strong evidence that it is not true. A possible reason is that the N4 algorithm removes significant features during image gradient smoothing.

Plane	**Raw**	**Corrected**
Sagittal	**81±3.8%**	81±4.0%
Coronal	77±3.1%	**80±5.1%**
Axial	**76±3.1%**	74±2.6%

Table 4.5: DSC performance of models trained corrected and raw MRI scans.

As for the custom networks, the best results were for the 3rd version and shown in Table 4.6 below.

Model name	**DSC, %**
v3 Small32	58
v3 Small64	**64**
v3 Large32	59
v3 Large64	61

Table 4.6: Results for custom CNNs trained on a sagittal view of raw MRI scans.

Comparing statistics for networks initialized randomly or using transfer learning we can see in Table 4.7. We can observe significantly performance boost for fine-tuned models trained on coronal and slightly better on axial views. As for the random wights, DSC is up for 1% on the sagittal plane, but taking overlapping intervals of standard deviation into account, we may consider them performing similarly to fine-tuned models.

Initialized	Sagittal, %	Coronal, %	Axial, %
Random	**82±3.1**	75±2.8	74±2.7
Transfer learning	81±4.2	**80±3.9**	**76±2.8**

Table 4.7: The DSCs for networks initialized randomly or with pre-trained encoder.

And in Table 4.8 you can see metrics for encoders performance on different views. Given mean DSCs are close for each other, especially considering overlapping standard deviation intervals.

Encoder	Sagittal, %	Coronal, %	Axial, %
ResNet-50	**83±4.5**	79±3.9	75±3.1
Inception v4	78±2.4	**81±4.0**	**76±2.6**

Table 4.8: The DSCs for networks initialized ResNet-50 and Inception v4.

In Figure 4.5 you can see distribution prediction DSCs per slice for the test subject sagittal view made by U-net ResNet-50. Here the full bar stands for the perfect agreement of the model with ground truth and empty bar for zero overlaps.

As you can see predictions for first and last slices mostly show that there is a perfect match with manual segmentation, which in this case means that there are no vertebral bodies on those images. At the same time predictions on the borders of the spinal cord for the appearance of VB but manual segmentation doesn't have it. An overall prediction quality for slices with VBs can vary within ±4% for consecutive images, we expect the picture will become more uniform with dataset growth.

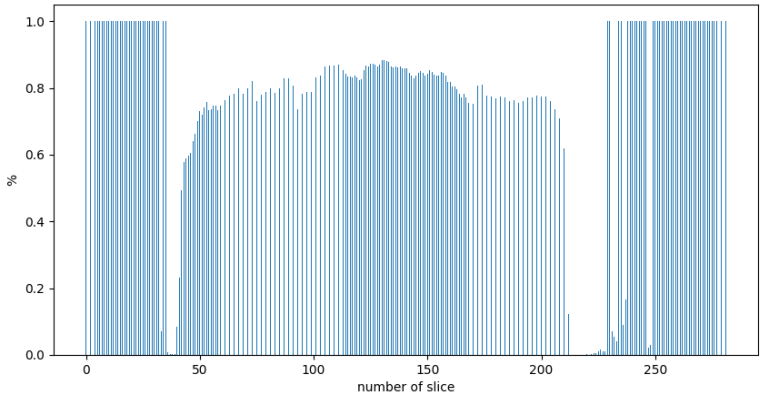


Figure 4.5: DSC per slice for test volume prediction on a sagittal view.

To have a better intuition behind the number in the results table here you can see examples of slices with best and worst predictions below in Tables 4.9 and 4.10.

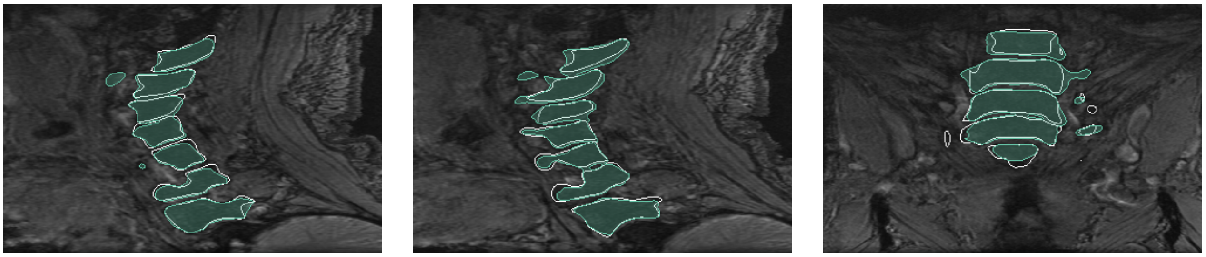


Table 4.9: Best predictions with DSC 89% on the left, 89% in the middle and DSC 87% on the right.

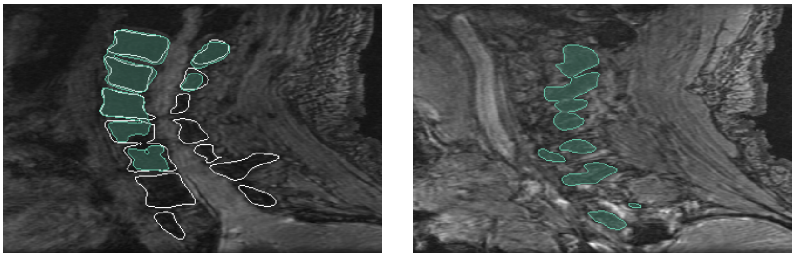


Table 4.10: Sample predictions with DSC 65% on the left, for some reason on this particular slice model didn't recognize some VBs, but on the slices before and after. DSC 0% on the right.

4.4 Discussion of limitations

Reported architectures were trained and evaluated on MRI scans made on a specific version of medical equipment. Because of scanners' technical and scanning parameters, trained models provided in this thesis may show sub-optimal results on images from other devices. To improve on other machine or scanner mode, and get reproduce reported model performance, there will be a necessity to fine-tune models on new scans with corresponding manual segmentation from a given machine. This procedure will adjust the weights of neural networks to generalize better on new MRI device setup.

Another limiting factor would be the presence of pathological VBs, e.g. with fractures or metastases. Because such training examples are not present in the VisSim dataset and their influence on tissue shape and pixel intensities are expected to reduce model prediction quality.

Given previous factors, one should consider numbers reported by different segmentation approaches with caution and take into account that on the similar task of thoracolumbar spine segmentation on MR data inter-rater variability ranges between mean DSCs of 88.4% and 91% [HSST18, DLSH02]. As our dataset didn't have segmentation from an independent second rater, it was not possible to compare the algorithm-rater difference with inter-rater differences.

Chapter 5

Conclusion and future work

5.1 Conclusion

In this thesis, we built an end-to-end pipeline for evaluation and development of neural networks with MRI scans. Summarizing answers for research questions:

Conclution 1. Reported results performance of 4 established and 1 hand-crafted deep CNNs in a task of cervical spine segmentation. We can see that FPN and U-net fine-tuned from encoders trained on ImageNet have performance close to state-of-the-art even without any post-processing.

Conclution 2. As for the value of MR artifact correction, we demonstrate that raw images, in general, have slightly better performance for sagittal and axial planes, and worse for the coronal axis. Also, 2 out of 3 models with the best results from Conclusion 1 were trained on N4 corrected scans.

Conclutions 3 and 4. Also FPN, U-net, PSPNet, and Linknet were test with random initialization using [HZRS15] and compared to metrics from fine-tuned networks. For the sagittal view, random weights did 1% better, but coronal and axial planes were 5% and 2% worse. As the numbers show, transfer learning from ImageNet does improve results on medical data by up to 5 percent in the best case.

Conclution 5. And models trained on sagittal tend to perform better than other planes. A larger number of training samples and more vertebral bodies shape variability may be a reason behind it. Models trained on coronal are very close to the sagittal view but significantly outperform those from the axial plane. This may be happening because of twice fewer data present in the dataset, because image spacing for an axial plane is 0.70, compared to 0.35 for sagittal and coronal.

5.2 Future work

As [RZKB19] reported and we had some evidence that transfer learning from models trained of natural images is not valuable an idea for future work is use models trained on medical data.

Also getting more intuition about NNs learned features would be valuable for future progress and first step in this direction would be to visualize convolutional kernels trained for various networks.

Another idea is to ensemble predictions from different architectures and have a pixel-wise voting, architectures may learn different features and predictions for uncertain areas of one model, could be solved by other models.

Also in the thesis no post-processing was employed and incorporating practices used by other researchers can lead significant improvements, example further steps would be smoothing borders, filling holes or interpolation between neighbor slices.

And besides models trained here, there are many other neural architectures and building-blocks available, e.g. dilated-convolutions or new state-of-the-art network EfficientDet [TPL19] released recently by Google researchers.

As having precise VBs border shape is very important, one can look at module reported on ICCV 2019 [TAJF19] which focuses on improving it.

List of Tables

List of Figures

Bibliography

[AK+16] ATHERTYA, Jiyo ; KUMAR, G S. u. a.: Fuzzy clustering based segmentation of vertebrae in T1-weighted spinal MR images. In: *arXiv preprint arXiv:1605.02460* (2016)

[BFC09] BROSTOW, Gabriel J. ; FAUQUEUR, Julien ; CIPOLLA, Roberto: Semantic object classes in video: A high-definition ground truth database. In: *Pattern Recognition Letters* 30 (2009), Nr. 2, S. 88–97

[BPK+18] BUSLAEV, A. ; PARINOV, A. ; KHVEDCHENYA, E. ; IGLOVIKOV, V. I. ; KALININ, A. A.: Albumentations: fast and flexible image augmentations. In: *ArXiv e-prints* (2018)

[CBA+15] CHU, Chengwen ; BELAVÝ, Daniel L. ; ARMBRECHT, Gabriele ; BANSMANN, Martin ; FELSENBERG, Dieter ; ZHENG, Guoyan: Fully automatic localization and segmentation of 3D vertebral bodies from CT/MR images via a learning-based method. In: *PloS one* 10 (2015), Nr. 11, S. e0143327

[CC17] CHAURASIA, Abhishek ; CULURCIELLO, Eugenio: Linknet: Exploiting encoder representations for efficient semantic segmentation. In: *2017 IEEE Visual Communications and Image Processing (VCIP)* IEEE, 2017, S. 1–4

[COR+16] CORDTS, Marius ; OMRAN, Mohamed ; RAMOS, Sebastian ; REHFELD, Timo ; ENZWEILER, Markus ; BENENSON, Rodrigo ; FRANKE, Uwe ; ROTH, Stefan ; SCHIELE, Bernt: The cityscapes dataset for semantic urban scene understanding. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, S. 3213–3223

[Dic45] DICE, Lee R.: Measures of the amount of ecologic association between species. In: *Ecology* 26 (1945), Nr. 3, S. 297–302

[DLSH02] DAVATZIKOS, Christos ; LIU, Dengfeng ; SHEN, Dinggang ; HERSKOVITS, Edward H.: Spatial normalization of spine MR images for

statistical correlation of lesions with clinical symptoms. In: *Radiology* 224 (2002), Nr. 3, S. 919–926

[EVGW+10] EVERINGHAM, Mark ; VAN GOOL, Luc ; WILLIAMS, Christopher K. ; WINN, John ; ZISSERMAN, Andrew: The pascal visual object classes (voc) challenge. In: *International journal of computer vision* 88 (2010), Nr. 2, S. 303–338

[GXV+17] GAONKAR, Bilwaj ; XIA, Yihao ; VILLAROMAN, Diane S. ; KO, Allison ; ATTIAH, Mark ; BECKETT, Joel S. ; MACYSZYN, Luke: Multi-parameter ensemble learning for automated vertebral body segmentation in heterogeneously acquired clinical MR images. In: *IEEE journal of translational engineering in health and medicine* 5 (2017), S. 1–12

[HCLN09] HUANG, S. ; CHU, Y. ; LAI, S. ; NOVAK, C. L.: Learning-Based Vertebra Detection and Iterative Normalized-Cut Segmentation for Spinal MRI. In: *IEEE Transactions on Medical Imaging* 28 (2009), Oct, Nr. 10, S. 1595–1605. http://dx.doi.org/10.1109/TMI.2009.2023362. – DOI 10.1109/TMI.2009.2023362. – ISSN 1558–254X

[HGK18] HERNÁNDEZ-GARCÍA, Alex ; KÖNIG, Peter: Further advantages of data augmentation on convolutional neural networks. In: *International Conference on Artificial Neural Networks* Springer, 2018, S. 95–103

[HSST18] HILLE, Georg ; SAALFELD, Sylvia ; SEROWY, Steffen ; TÖNNIES, Klaus: Vertebral body segmentation in wide range clinical routine spine MRI data. In: *Computer methods and programs in biomedicine* 155 (2018), S. 93–99

[HZRS15] HE, Kaiming ; ZHANG, Xiangyu ; REN, Shaoqing ; SUN, Jian: Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In: *Proceedings of the IEEE international conference on computer vision*, 2015, S. 1026–1034

[HZRS16] HE, Kaiming ; ZHANG, Xiangyu ; REN, Shaoqing ; SUN, Jian: Deep residual learning for image recognition. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, S. 770–778

[JMIC13] JOHNSON, Hans J. ; MCCORMICK, M. ; IBÁÑEZ, L. ; CONSORTIUM, The Insight S. ; KITWARE, INC. (Hrsg.): *The ITK Software Guide*. Third. Kitware, Inc., 2013. http://www.itk.org/ItkSoftwareGuide.pdf. – In press

[Kar] KARPATHY, Andrej: *CS231n Convolutional Neural Networks for Visual Recognition*. http://cs231n.github.io/neural-networks-1/, . – Accessed: 2019-12-29

[KLPV16] KOREZ, Robert ; LIKAR, Boštjan ; PERNUŠ, Franjo ; VRTOVEC, Tomaž: Model-based segmentation of vertebral bodies from MR images with 3D CNNs. In: *International Conference on Medical Image Computing and Computer-Assisted Intervention* Springer, 2016, S. 433–441

[LDG+17] LIN, Tsung-Yi ; DOLLÁR, Piotr ; GIRSHICK, Ross ; HE, Kaiming ; HARIHARAN, Bharath ; BELONGIE, Serge: Feature pyramid networks for object detection. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, S. 2117–2125

[LMB+14] LIN, Tsung-Yi ; MAIRE, Michael ; BELONGIE, Serge ; HAYS, James ; PERONA, Pietro ; RAMANAN, Deva ; DOLLÁR, Piotr ; ZITNICK, C L.: Microsoft coco: Common objects in context. In: *European conference on computer vision* Springer, 2014, S. 740–755

[LSD15] LONG, Jonathan ; SHELHAMER, Evan ; DARRELL, Trevor: Fully convolutional networks for semantic segmentation. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, S. 3431–3440

[NFE+12] NEUBERT, Aleš ; FRIPP, Jurgen ; ENGSTROM, Craig ; SCHWARZ, Raphael ; LAUER, Lars ; SALVADO, Olivier ; CROZIER, Stuart: Automated detection, 3D segmentation and analysis of high resolution spine MR images using statistical shape models. In: *Physics in Medicine & Biology* 57 (2012), Nr. 24, S. 8357

[PGM+19] PASZKE, Adam ; GROSS, Sam ; MASSA, Francisco ; LERER, Adam ; BRADBURY, James ; CHANAN, Gregory ; KILLEEN, Trevor ; LIN, Zeming ; GIMELSHEIN, Natalia ; ANTIGA, Luca ; DESMAISON, Alban ; KOPF, Andreas ; YANG, Edward ; DEVITO, Zachary ; RAISON, Martin ; TEJANI, Alykhan ; CHILAMKURTHY, Sasank ; STEINER, Benoit ; FANG, Lu ; BAI, Junjie ; CHINTALA, Soumith: PyTorch: An Imperative Style, High-Performance Deep Learning Library. Version: 2019. http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning.pdf. In: WALLACH, H. (Hrsg.) ; LAROCHELLE, H. (Hrsg.) ; BEYGELZIMER, A. (Hrsg.) ; ALCH-BUC, F. (Hrsg.) ; FOX, E. (Hrsg.) ; GARNETT, R. (Hrsg.): *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, 8024–8035

[PW90] PANJABI, Manohar M. ; WHITE, AA: Clinical biomechanics of the spine. (1990), S. 86–100

[RDS+15] RUSSAKOVSKY, Olga ; DENG, Jia ; SU, Hao ; KRAUSE, Jonathan ; SATHEESH, Sanjeev ; MA, Sean ; HUANG, Zhiheng ; KARPATHY, Andrej ; KHOSLA, Aditya ; BERNSTEIN, Michael ; BERG, Alexander C. ; FEI-FEI, Li: ImageNet Large Scale Visual Recognition Challenge. In: *International Journal of Computer Vision (IJCV)* 115 (2015), Nr. 3, S. 211–252. http://dx.doi.org/10.1007/s11263-015-0816-y. – DOI 10.1007/s11263–015–0816–y

[RFB15] RONNEBERGER, Olaf ; FISCHER, Philipp ; BROX, Thomas: U-net: Convolutional networks for biomedical image segmentation. In: *International Conference on Medical image computing and computer-assisted intervention* Springer, 2015, S. 234–241

[RHGS15] REN, Shaoqing ; HE, Kaiming ; GIRSHICK, Ross ; SUN, Jian: Faster r-cnn: Towards real-time object detection with region proposal networks. In: *Advances in neural information processing systems*, 2015, S. 91–99

[RZKB19] RAGHU, Maithra ; ZHANG, Chiyuan ; KLEINBERG, Jon ; BENGIO, Samy: Transfusion: Understanding Transfer Learning for Medical Imaging. Version: 2019. http://papers.nips.cc/paper/8596-transfusion-understanding-transfer-learning-for-medic pdf. In: WALLACH, H. (Hrsg.) ; LAROCHELLE, H. (Hrsg.) ; BEYGELZIMER, A. (Hrsg.) ; ALCH-BUC, F. (Hrsg.) ; FOX, E. (Hrsg.) ; GARNETT, R. (Hrsg.): *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, 3342–3352

[SIVA17] SZEGEDY, Christian ; IOFFE, Sergey ; VANHOUCKE, Vincent ; ALEMI, Alexander A.: Inception-v4, inception-resnet and the impact of residual connections on learning. In: *Thirty-first AAAI conference on artificial intelligence*, 2017

[SSP03] SIMARD, P. Y. ; STEINKRAUS, D. ; PLATT, J. C.: Best practices for convolutional neural networks applied to visual document analysis. In: *Seventh International Conference on Document Analysis and Recognition, 2003. Proceedings.*, 2003. – ISSN null, S. 958–963

[TAC+10] TUSTISON, Nicholas J. ; AVANTS, Brian B. ; COOK, Philip A. ; ZHENG, Yuanjie ; EGAN, Alexander ; YUSHKEVICH, Paul A. ; GEE, James C.: N4ITK: improved N3 bias correction. In: *IEEE transactions on medical imaging* 29 (2010), Nr. 6, S. 1310

[TAJF19] TAKIKAWA, Towaki ; ACUNA, David ; JAMPANI, Varun ; FIDLER, Sanja: Gated-SCNN: Gated Shape CNNs for Semantic Segmentation. In: *ICCV* (2019)

[TPL19] TAN, Mingxing ; PANG, Ruoming ; LE, Quoc V.: Efficientdet: Scalable and efficient object detection. In: *arXiv preprint arXiv:1911.09070* (2019)

[vCV11] VAN DER WALT, S. ; COLBERT, S. C. ; VAROQUAUX, G.: The NumPy Array: A Structure for Efficient Numerical Computation. In: *Computing in Science Engineering* 13 (2011), March, Nr. 2, S. 22–30. `http://dx.doi.org/10.1109/MCSE.2011.37`. – DOI 10.1109/MCSE.2011.37. – ISSN 1558–366X

[VRDJ95] VAN ROSSUM, Guido ; DRAKE JR, Fred L.: *Python tutorial*. Centrum voor Wiskunde en Informatica Amsterdam, The Netherlands, 1995

[Yak19] YAKUBOVSKIY, Pavel: *Segmentation Models*. `https://github.com/qubvel/segmentation_models`. Version: 2019

[ZSQ+17] ZHAO, Hengshuang ; SHI, Jianping ; QI, Xiaojuan ; WANG, Xiaogang ; JIA, Jiaya: Pyramid scene parsing network. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, S. 2881–2890

[ZVE+14] ZUKIĆ, Dženan ; VLASÁK, Aleš ; EGGER, Jan ; HOŘÍNEK, Daniel ; NIMSKY, Christopher ; KOLB, Andreas: Robust detection and segmentation for diagnosis of vertebral diseases using routine MR images. In: *Computer Graphics Forum* Bd. 33 Wiley Online Library, 2014, S. 190–204

[ZZP+19] ZHOU, Bolei ; ZHAO, Hang ; PUIG, Xavier ; XIAO, Tete ; FIDLER, Sanja ; BARRIUSO, Adela ; TORRALBA, Antonio: Semantic understanding of scenes through the ade20k dataset. In: *International Journal of Computer Vision* 127 (2019), Nr. 3, S. 302–321