Bachelor Thesis

Automated Segmentation of Bones for the Age Assessment in 3D MR Images using Convolutional Neural Networks

Paul-Louis Pröve

mail@plpp.de

supervised by

Prof. Dr. Dennis Säring

dsg@fh-wedel.de

Hamburg,

August 15, 2017

Abstract

The age assessment is a complicated procedure used to determine the chronological age of an individual who lacks legal documentation. While there is no method that provides an exact identification of the age, current practices require high amounts of manual work and invasive X-ray imaging. Previous publications show that the status of growth plates in the knee could be an appropriate indicator for the border to adulthood. As part of a DFG study, noninvasive MRI recordings were collected to investigate this hypothesis further. This thesis provides a step towards a solution by fully automating the extraction of bone in knee MRIs using convolutional neural networks to reduce the data complexity. These results show a dice similarity coefficient of 98% when compared to the manually labeled ground truth and further improve the work of previous studies. This thesis concludes with a proof of concept estimating the age of individuals using the generated bone segmentations. The preliminary results show the potential of this approach achieving a mean difference of 0.48 ± 0.32 years.

Contents

List of Figures

List of Tables

1 Introduction

Recent advances in artificial intelligence have led to fully automated workflows that often exceed human performance. State of the art neural networks can classify images into thousands of categories more accurately and magnitudes faster than humans [1]. They translate texts from multiple languages [2], drive cars autonomously through cities [3], and detect malware in computer systems [4]. In most of these cases, they have been trained on tens of thousands, or even millions, of data samples. Neural networks have also found great success in the field of medical image analysis where data sets are often much smaller. Although the same techniques can be applied, one is often confronted with a different set of challenges.

1.1 Field and Context

One of the main topics of forensic science is the age assessment, which is a difficult procedure to determine the chronological age of an individual lacking legal documentation. It plays an essential role in asylum and criminal proceedings to protect children and afford them with provisions they are entitled to by law. Due to the European migration crisis, this topic has gained much public attention. There is no method that offers an exact identification of the age, but several assessments are currently used in the EU that can be separated into two categories [5].

Personal interviews or examinations are held to gain an observation of physical or psychological features. These practices include a high amount of manual work, and they are also subjective to the person conducting the test. The other type of assessments uses X-rays to observe physical traits like the collar bone, teeth or carpal growth plates [5]. During the X-ray recordings, the patient is exposed to ionizing radiation, which can increase the risk of long term effects like cancer [6]. It is an invasive imaging method that can only be performed as part of a judicial order.

For this reason, there is a high demand for a fully automated, unbiased and noninvasive method for accurately determining the age of a person.

1.2 Research Problem

The above requirements have led to studies evaluating the age assessment based on MRI recordings that are noninvasive. Previous publications [7, 8] show that the closing process of growth plates in the knee aligns with the coming of age of teenagers. Therefore, the status of the growth plates could be an appropriate indicator for the age around the border to adulthood.

However, these studies are often based on qualitative comparisons, with computer-based methods being a more desirable approach. This could further reduce prejudiced results while speeding up the process and creating a standardized measure. In a recent study funded by the German Research Foundation (DFG) new MRI data is collected prospectively to verify this hypothesis scientifically. Due to the underlying technology used in MRI machines, these images show high detail in non-bone tissue. Despite this being an advantage in many other medical applications, it adds complex information outside the scope of the bone and growth plates.

Neural networks are known to be feature selectors, meaning that they will learn to extract information that is relevant to the task [9]. This assumes that the size of the data set and the complexity of the problem enable the network to find correlating features. Based on a series of tests, it was not possible to create a stable algorithm that would predict the age of an individual based on their knee MRI. A bone segmentation is required to reduce the MR images to only relevant information in the frame. Since methods like region growing lead to insufficient results due to similar intensities of other tissue, time-consuming manual modifications are needed.

1.3 Previous Research

A study led by Dodin et al. in 2011, focussed on the same goal by masking the Femur and Tibia bone from other soft tissue. They used the ray casting technique, which disassembles the MR images into several surface layers to find the boundaries of the bones. Afterward, multiple partial maps were merged for the final result [10]. Dam et al. presented another approach in 2015 focusing on the segmentation of cartilages in the knee for the research on osteoarthritis. Their method combined a multi-atlas rigid registration and voxel classification. Besides masking medial and lateral cartilages, they also applied their technique on the Tibia [11]. In a publication from 2014 [12], Stern et al. used carpal MRIs of teenagers to estimate their age based on growth plates in the wrist and fingers. The accuracy of their results is in line with the manual work of a radiologist using common radiographic methods.

1.4 Focus of this Thesis

The focus of this thesis revolves around creating a fully automated workflow that segments the long bones in 3D MRIs of human knees. Convolutional neural networks will be the base tool for this study because they have been setting state of the art results in a vast majority of image segmentation tasks. Lastly, a fully automated proof of concept will be presented that uses the segmented data to assess the age of individuals.

2 Background

This chapter provides an overview of prerequisites and previous work that led to the main task of this thesis. It features a section for the necessary medical knowledge as well as a hierarchical derivation to the appropriate branch of computer science.

2.1 Medical Knowledge

This section will briefly explain how the imaging technology works that was used to create the data set, as well as provide basic information about the structure and growing process of growth plates in the human body.

2.1.1 Magnetic Resonance Imaging

Magnetic Resonance Imaging (MRI) uses magnetic fields and radio frequencies to probe tissue structure. In contrast to X-ray and CT which require the exposure of ionizing radiation, MRI is a noninvasive imaging method. As such it has become an essential diagnostic imaging modality in the medical field [13].

96% of the human body is made up of hydrogen, oxygen, carbon, and nitrogen, all of which are referred to as MR active. These elements have an odd atomic mass number giving the nucleus a spin. Due to the laws of electromagnetic induction, the motion of unbalanced charge produces a magnetic field around itself. Hydrogen is the element used in MRI because its solitary proton in the nucleus gives it a relatively large magnetic moment [13].

The positively charged hydrogen particles in water produce a signal when exposed to a strong external magnetic field. This field is supplied by a magnet in the MRI machine aligning the magnetic field of the hydrogen atoms to its own. Gradient coils are used to cause a linear change in the strength of this field. By alternating the current of these coils on the x, y and z axes, it is possible to calculate a three-dimensional image of the tissue [13].

2.1.2 Growth Plates

Most bones in the human body grow through a chondral process, where cartilages near the end of a long bone are ossified over time. These continuously renewing cartilages are referred to as growth plates, and they are responsible for extending the longitudinal length of the shaft. The growth of a bone comes to an end when cells in the cartilages stop their proliferation, and

the growth plate starts to close. This is correlated with increasing amounts of sex hormones during puberty [14]. On average women will stop growing between age 14 to 17, while for men this happens between age 18 to 22 [15].

Figure 1: Different states of the Tibia

The closing process initializes at the center of the growth plate and extends to the edges. Literature often classifies the current state in open, centrally-closed and closed [16]. Growth plates are visible in T1 weighted MRIs as dark horizontal lines, whereas the surrounding bone appears bright.

2.2 Computer Science

The primary task of this thesis revolves around a particular branch of computer science problems. This section provides a path to where most of the work will take place. Figure 1 visualizes the hierarchical relationship of the following subsections.

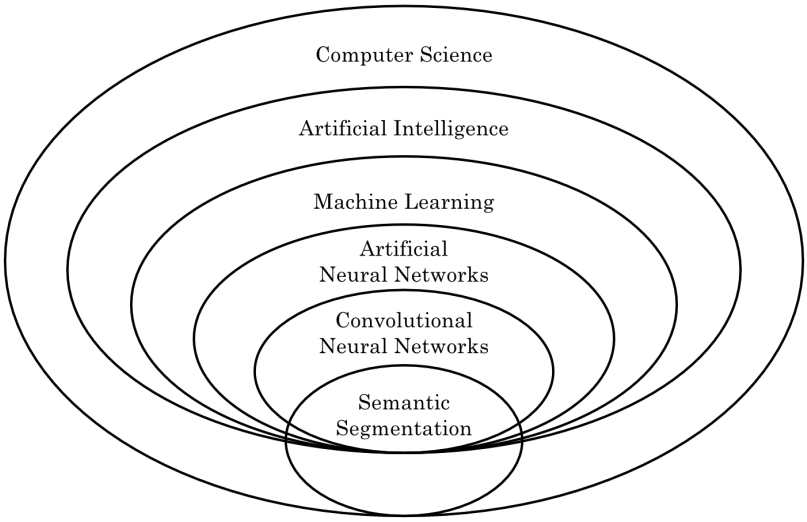

Figure 2: Hierarchical relationship of relevant topics

2.2.1 Artificial Intelligence

Artificial Intelligence (AI) is understood as the effort of automating a given task that typically needs human intelligence to solve. The history of AI started in the 1950s, where a particular type called "symbolic AI" gained popularity. It was believed that human level intelligence could be achieved through hard-coded rules that programmers specified [17].

Taking a complex problem like playing chess and continuing to break it into smaller problems, until they can be solved with known logic. While it was effective for certain tasks, fuzzy problems like image classification, speech recognition or language translation were difficult to tackle. Over the years a new approach was found, which today is referred to as machine learning (ML).

2.2.2 Machine Learning

The concept of classical programming is that an engineer defines a set of rules, called an algorithm, which uses input data to calculate some form of output data [17].

Figure 3: Classical programming pipeline

A machine learning algorithm is an algorithm that can learn from data [18]. It can be used to calculate these rules automatically, so they do not have to be specified by hand. Three components are needed for such an approach.

- Input data the algorithm is supposed to transform
- Output data the algorithm is supposed to predict
- A measurement to validate the performance of a prediction

It works by feeding input and output data into a pipeline, which will learn to transform one into the other. With the advantage that no explicit programming is needed to generate the rules, comes the disadvantage that prior input and output data is required for the initial learning process.

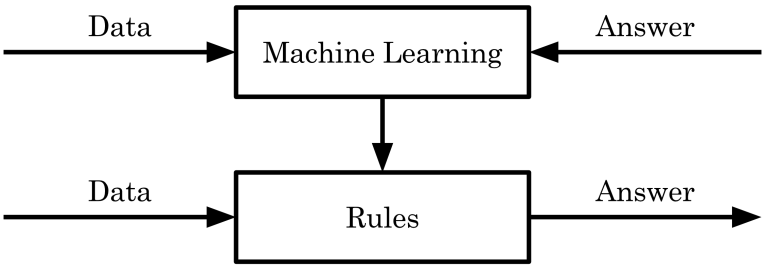

Figure 4: Machine learning pipeline

Machine learning may be applied as an effective method if it is not feasible or possible to define an algorithm by hand and sufficient data is available for training. How much “sufficient” is depends on factors like the type of task, the complexity of the data, the uniformity of the data, the type of machine learning algorithm and others.

There are different subparts to machine learning like supervised and unsupervised learning. Supervised learning is used when it is clear what the output data looks like, whereas unsupervised learning can help to find unknown patterns in the data. Examples of supervised learning techniques include linear regression, naive Bayes, support vector machines, decision trees, random forests, gradient boosting and artificial neural networks (ANNs). Since the primary interest of this study revolves around ANNs, this will be the focus of following chapters.

2.2.3 Artificial Neural Networks

Artificial neural networks are inspired by neurobiological concepts of the human brain. However, they are not models of the human brain. There is no evidence to support that the brain implements anything like the learning mechanisms used in ANNs [17]. Artificial neural networks are mathematical frameworks for learning representations from data and one of, if not the only tool in deep learning (DL) — a subset of machine learning.

The origin of the term deep learning is twofold. On the one hand, it refers to the fact that ANNs can learn "deep" hierarchical information from data. On the other, it describes that they show multiple layers of depth within their architecture. ANNs are as old as machine learning itself, but only gained a lot of their popularity in recent years [17].

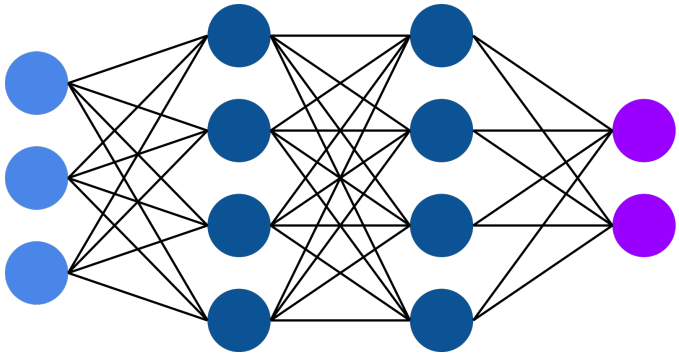

Figure 5: An ANN with 3 input nodes, 2 hidden layers with 4 nodes each and 2 output nodes

These dense layers inside artificial neural networks contain entities called nodes that are jointly linked with one another. Every node in a dense layer is connected to each node in the previous and following layer. This structure gives ANNs the capacity to approximate complex mathematical functions and learn global representations in the data. These nodes, also called parameters, can range to hundreds of millions depending on the task [19].

2.2.4 Convolutional Neural Networks

Convolutional neural networks (CNNs) are a specific type of ANN that uses an operation called convolution in at least one of their layers. The first CNN was introduced by Yann LeCunn [20] in 1990 at which time its popularity was limited. In 2012 a CNN called AlexNet [21] won the prestigious ImagetNet competition, and these types of networks have been winning ever since.

A convolution is a mathematical operation on two functions of real-valued argument. In imaging terminology, the first function refers to the input and the second function describes the kernel. The output of this operation is called a feature map [18].

Convolutions are frequently used in image processing, which is also why they were introduced to visual tasks in the field of deep learning. They allow learning local patterns in the data instead of treating the input features in a global manner like dense layers do.

Another type of layer that is frequently used in CNN architectures perform a pooling operation. By doing so, the spatial resolution is reduced, and only the most relevant features are kept. This is important to maintain a manageable network size.

2.2.5 Semantic Segmentation

CNNs can help to solve different types of supervised machine learning problems of which regression, classification, and segmentation are the most common.

A regression describes the prediction of one or multiple continuous outputs. An example for this would be the age prediction of a person based on their knee MRI. Regression is also an important part of object detections, where it is applied to determine coordinates in an image.

A classification sorts the input into one or multiple categories, like predicting if the growth plate is open, centrally-closed or closed. It can be seen as n parallel regressions where n equals the total number of classes. The continuous output for each class is the network's confidence of the input belonging to this category. For 1 out of n classifications, the most probable category is predicted. For m out of n classifications, a threshold defines at which point a category is predicted.

A segmentation creates an image of identical dimensions as the input and masks specific regions within. Every channel of the output mask belongs to a particular category that is segmented. A segmentation can be seen as a classification of each pixel/voxel. For this study, the Femur, Tibia, and Fibula needed to be masked from the rest of the MRI content. This is called a semantic segmentation because the process is based on a semantic meaning of objects in the image.

Figure 6: An example of semantic segmentation

3 Methods

The purpose of this chapter is twofold. It will discuss the design and processing decisions that were made for the development of this project, while also giving critical insight into how these applied technologies work. The sections for setup and data set describe the working environment for this study. Preprocessing, architecture, training and postprocessing show in chronological order how the development was addressed.

3.1 Setup

The workstation included an Intel i5 processor, 16GB of RAM and most importantly a NVIDIA GeForce GTX1060/6GB. Neural Networks can be trained more efficiently on GPUs than CPUs. This is because the simpler but highly parallelized architecture of graphics chips plays in favor for the needed calculations in deep learning [22].

The computer ran Ubuntu 16.04 with the NVIDIA CUDA and cuDNN libraries installed to take advantage of the GPU. As the primary programming language Python 2 was chosen due to its simple syntax and popularity in the deep learning field. The code was briefly tested with Python 3 as well and seemed to work. Keras was used as the framework for training models because its high level syntax allows fast prototyping and testing. The development environment was a Jupyter Notebook, which allowed a flexible execution of code snippets instead of running the entire program for every single change.

The processing of medical image data needed a library that could handle these formats. SimpleITK is a Python binding to the ITK library written in C++. It includes many tools for image processing and is especially popular in the medical field. Other libraries were also used for smaller tasks.

3.2 Data Set

The data set is a collection of three dimensional, T1 weighted and non-fat-saturated MR images showing the right knee. The number of available samples grew during the project. For most of the development time, it included 150 samples that came from 3 different sources. All images were provided in the MHD file format, which is very common in the medical field. It features lossless compression, as well as technical meta information about each image. The resolution of these samples differed between sources.

Source	Prospective	View	Samples	Maps	Resolution
Class 1	Yes	Coronal	65	36	800x800x41
Class 2	Yes	Coronal	80	40	512x512x24
Class 3	No	Sagittal	5	0	multiple

Table 1: Details of the data set separated by source

Class 3 data featured a sagittal perspective, while all other samples were taken coronal. This mattered because the resolution of all images was roughly 20 times lower on the z-axis. The 80 class 2 images originated in a previous study that investigated the condition of growth plates led by Jopp et al. A total of 65 class 1 images were taken as part of the DFG project, which showed the highest resolution due to a different MRI machine.

Figure 7: Six different slices of one 3D MRI

The images above move through the slices of one 3D MRI. The Femur already becomes visible in the first frame, whereas the Tibia starts to show in the second frame. The Fibula becomes visible in the lower left corner of the fourth image and can be seen more clearly in the fifth.

A total of 76 segmentation maps were initially labeled by semi-automatic region growing followed by a manual adaption. The labeling process took 2 hours per 3D image, and each map showed Femur, Tibia, and Fibula separately. Not the entire image was segmented, but only a square window around the point where Femur and Tibia meet.

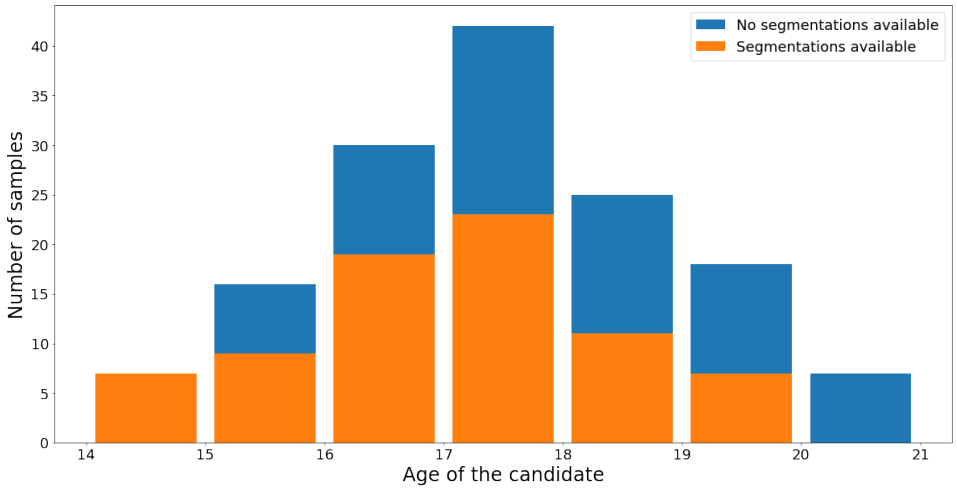

Figure 8: Age distribution in the data set

The candidates chosen for the MRI recordings were all german males between the age of 14 and 21. Their age was normally distributed, and a majority of the candidates were recorded twice roughly a year apart.

3.3 Preprocessing

The parameters in a Neural Network commonly range from tens of thousands to hundreds of millions. This complexity allows the model to learn on its own what features of an image are relevant for any given task. It works in conjunction with the fact that high volumes of data are available for the training. Because of the small data set that was available for this study, several types of preprocessing were applied to the images. For the most part, these techniques remove irrelevant information and reduce variance between multiple samples. Other preparation methods experimented with the difference between 2D and 3D data as well as the influence of separate segmentation channels on the output.

3.3.1 Training, Validation, and Testing Sets

The data was split into three subsets of which each was used for a different purpose (training, validation and testing). The training set is commonly the largest of the three and contains the data that is applied to the actual learning process. It's the only portion of the data the network will draw direct conclusions from.

The validation data is used to regularly measure the performance of the model and check whether overfitting occurs or not. If the accuracy of the validation set drops below the results of the training data, the network is starting to memorize the data it knows rather than generalizing on the concept.

The third subset is referred to as the testing data. In contrast to the validation set, it's only used once in the very end, to give a final score. The idea is that by building a model based on the validation results a certain amount of information bleed occurs, where the network will implicitly learn from the validation data. In order to prevent biased results, the testing data is used as a last performance reference.

- Traning Set: 74% of the data
- Validation Set: 13% of the data
- Test Set: 13% of the data

3.3.2 Cropping, Resizing, and Resampling

The framing included vast areas of the upper and lower leg to be visible in the picture. Since these were not seen as relevant, they were cropped out. An existing algorithm was used to detect the center where Tibia and Femur meet and only use a square window around this point.

Although there is no theoretical size limitation to using convolutional neural networks, it is desirable to reduce the spatial resolution and therefore decrease the number of calculations. 224 x 224 pixels for each slice still gave enough detail to identify the shape of Femur, Tibia, and Fibula, while also being a common resolution for CNNs.

Resizing the z-axis was problematic because the resolution was roughly 20 times lower. When scaling along this dimension, the segmentation maps of different slices started to blend and create merged maps of multiple layers. In order to get the images from two different main sources on the same scale, two different approaches were tried.

The 41 slices per image of the class 1 data were padded with empty pixels to 48 slices. Afterward, every second 2D image was resampled to the same 24 slices the data from Jopp et al. showed. It was not possible to do it the other way around and upscale 24 slices to 48, because the interpolated segmentation maps may have falsified the data.

Throwing away perfectly good data is very uncommon in the machine learning field, especially if data sets are small. However, one slice shares a lot of similar information with its neighboring slices and can be seen as a sort of

image augmentation between the two. By using the same spatial resolution for both sources, the balance between the amount of class 1 and 2 data was kept. A total of 76 images with a resolution of 24 x 224 x 224 were now available for training.

The second approach converted the volumetric data to 2D while using all 41 slices from the class 1 data and all 24 slices from class 2. No samples were thrown away, but the data was no longer volumetric.

3.3.3 Normalization

The normalization of images refers to the process of transforming all samples on the same scale of values. Two techniques for this are popular in the field of deep learning. The first one is called feature scaling, where every sample is normalized between 0 and 1.

$$x' = \frac{x - min(x)}{max(x) - min(x)}$$

The second technique calculates the standard score, where the mean is subtracted from the samples and then divided by the standard deviation.

$$x' = \frac{x - \mu}{\sigma}$$

In this case, the mean and standard deviation are not calculated for every image, but for all training samples. This centers the intensities around the average brightness of the training data. When normalizing validation and test data, one would also use the mean and standard deviation from the training set because those are the measures the architecture is expecting.

The standard score presents a more desirable approach because its mean centering helps to prevent that parameters get abnormally large or get set to zero during training. It is also less sensitive to outliers e.g. extremely bright spots in the image, which frequently appeared in the form of high intensities.

In contrast to the regular use of the standard score, every image was normalized on its own mean and standard deviation instead of calculating it on the entire training data. Class 2 featured roughly five times higher intensities than the class 1 data because they were recorded on different MRI machines. Figure 9 shows two spikes in mean intensities that had to be dealt with. By normalizing each 3D image on its own values, these two distributions were unified. Furthermore, this procedure will normalize future samples on the same scale the network expects no matter what the bit depth of an image is.

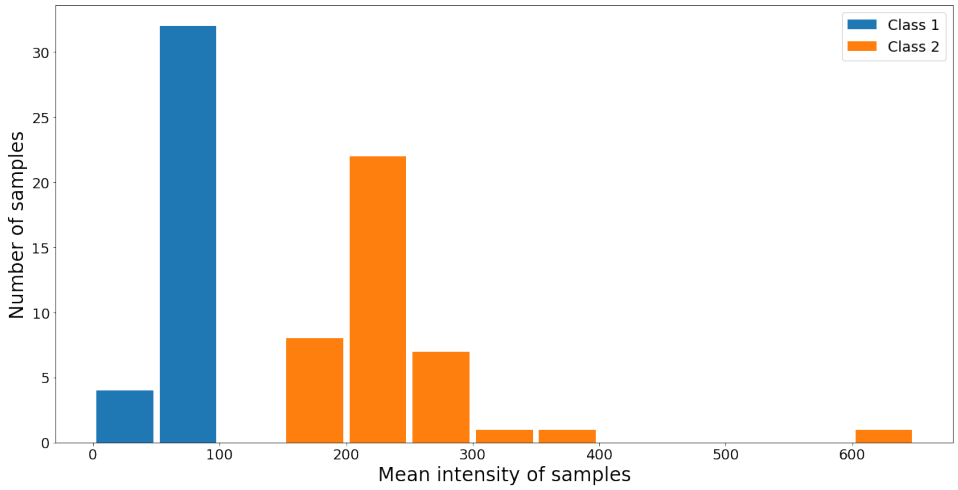

Figure 9: Intensity distribution in the data set

This approach also has a downside. The network can no longer use the mean intensity and the standard deviation as a feature to learn from because every image will be identical according to these measures. However, these two features showed no correlation to the accuracy of a prediction, which is why it was decided to use this approach.

3.3.4 N4 Bias Field Correction

Due to the inhomogeneous magnetic field used in MRI machines, a bias field is present as a low-frequency non-uniform intensity in the acquired images. Several methods have been developed in the past of which N4ITK is the de facto standard in the field [23]. As the name suggests, it is included in the Insight Toolkit (ITK) for which SimpleITK delivered an available Python binding. Using the N4 Bias Field Correction with its default settings on a single 24 x 224 x 224 image took between 60 and 120 seconds.

Figure 10: Raw image, corrected image and intensity differences

3.3.5 Augmentation

Image augmentation is a popular approach to virtually increase the size of the data set. The general idea is that a neural network will overfit more when learning one sample n times, in comparison to learning n alternations of this sample just once. It helps to generalize on new images instead of memorizing patterns in the training data.

Lossless augmentation techniques are those that don't change the values and relative localities in a sample. In 2D these include vertical and horizontal flips, as well diagonal flips if width and height are equal. Note that any multiples of 90-degree rotations can also be created using a combination of these flips. Although the samples are changed as a whole, they do not vary concerning their contained values.

The term "lossless" only refers to the technical change and not necessarily to the semantic change. Vertically flipping a slice of this data set switches the absolute positions of Femur and Tibia. This might make it harder for the network to distinguish between the two.

Lossy augmentation methods include a variety of image transformations. Common choices include:

- Horizontal shifts
- Vertical shifts
- Rotations
- Shear mapping
- Brightness adjustments
- Contrast adjustments
- Gamma adjustments

For this study horizontal flips were implemented to give the impression that images from both knees were available. In addition to this, these images were shifted 24 pixels on the horizontal axis to add another type of augmentation.

Interestingly, these methods did not improve the accuracy of the model. The horizontal flips even hurt the performance and were therefore removed. The horizontal shifts were kept in the training set, so the network would perform better on unseen data that was not perfectly aligned.

3.3.6 2D and 3D data

Although convolutional neural networks are most commonly connected with 2D data such as photos or drawings, they can be applied in any dimensional space. In Natural Language Processing (NLP) 1D convolutions are often used on sentences and in finance they can be applied to time series forecasting. In the medical field where a variety of volumetric data exists, 3D convolutions have become a popular choice for building deep learning solutions.

In the context of neural networks, one has to differentiate between volumetric data with one channel and 2D data that consists of multiple color channels. Both are an example of 3D data, but the volumetric images feature three spatial dimensions, whereas photos feature only two. Convolutions commonly traverse the spatial dimensions, which means photos with multiple channels are usually not subject to 3D convolutions. Instead, various input channels are fed into a 2D CNN.

Although the data set consisted of volumetric MRI data, the z-axis showed 20 times fewer voxels than the other two dimensions. Not knowing the influence of this situation both 3D and 2D architectures were investigated for this project. The 3D model didn't use any MaxPooling on the already small z-axis. A kernel size of 3 x 3 x 3 was used similarly to the 3D version of U-Net [24]. The 2D data was created by using each slice as a single input image.

As described in 3.3.2 the 3D convolutional network used 24 x 224 x 224 voxel images from both the class 1 and 2 data. In contrast, the 2D network used all of the available slices from each source as separate images.

While in theory, the spatial information of the z-axis gave the 3D network more contextual information of every slice, the 2D model performed better at the end. The latter was also less computationally expensive and allowed working with data where only single slices were available per sample.

3.3.7 Separate Bone Maps

The initial segmentation maps came with three separate values for Femur, Tibia, and Fibula. With this information, it was possible to train a network that would segment the three bones while still differentiating between them.

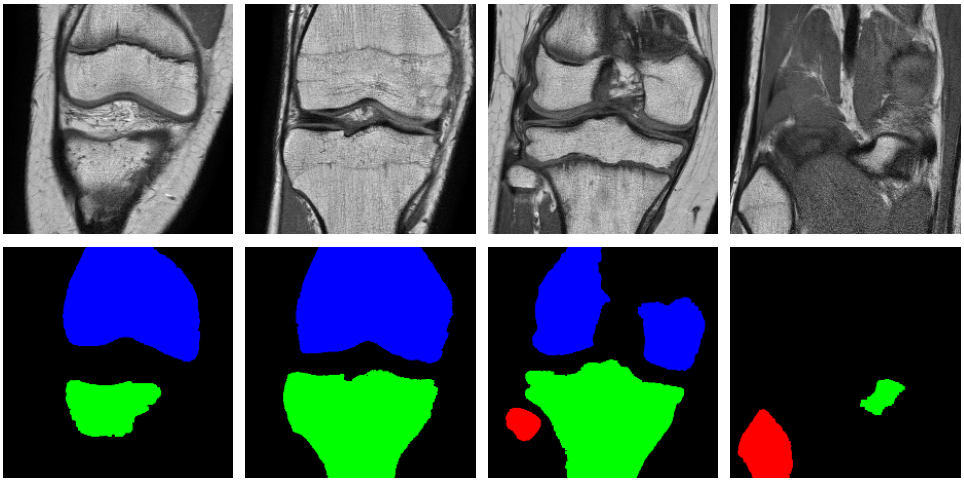

Figure 11: Examples of slices with their ground truth segmentations

However, this led to problems with the Fibula, which was not recognized by any of the applied architectures. The channel was always predicted empty. Its appearance in the images accounted for 3.4% of the segmented area, whereas the Tibia made up 41% and the Femur accounted for the remaining 55.6%. This led the network to learn that an empty prediction of this channel is valid in 96.6% of the cases, which is very accurate on its own. Instead of spending capacity on improving a channel that only makes up a tiny fraction of the result, the network improved the performance of Femur and Tibia.

The only successful method was training a separate model for each of the three bones. The network started to predict empty segmentation maps in the beginning because the actual masks only made up a fraction of the frame. The only way to improve from here was to find the Fibula in the non-empty slices and segment its region. This was different from the tests before, where the unsatisfactory performance of the Fibula was balanced by making more accurate segmentations of the other two bones.

For another experiment, the three channels were merged to create a network that would segment any long bone in the image. Not only did this solve the problem with the Fibula as well, but the predictions of Femur and Tibia were just as accurate. The network didn't need to learn any more that the global position in the image accounts for whether or not an object should be segmented. Instead, it could segment any long bone in any position in the frame.

While building the architecture in the following sections, this merged channel approach was used. It helped to speed up the design process because different settings only needed to be trained once instead of once for every bone.

3.4 Architecture

The majority of classification CNNs use a recurrent combination of convolutional and pooling layers [21, 25, 26, 27]. By combining the local learning patterns of convolutions with the spatial reductions of pooling, the input is compressed to a dense representation of its most important features.

As mentioned in 2.2.5 a segmentation can be seen as a classification for every pixel. As such, early CNN segmentation models used small patches of the input image only to predict a single pixel through a classification pipeline. Afterward, a full segmentation map was assembled using each of these pixels. This process was very slow, and it also prevented the network to have a field of view larger than the inserted patch.

An improvement for segmentations came through architectures referred to as encoder-decoder models. The encoding process describes the same spatial compression used in classification networks. Afterward, in the decoding step, the spatial resolution is brought back to its original shape and further processed by additional convolutions. This yields huge speed improvements, while also increasing the accuracy of the prediction. Encoder-decoder networks are the de facto standard in the current field of image segmentation [28, 29, 30, 31, 32, 33, 24, 27, 34], and they were the initial starting point for building architecture.

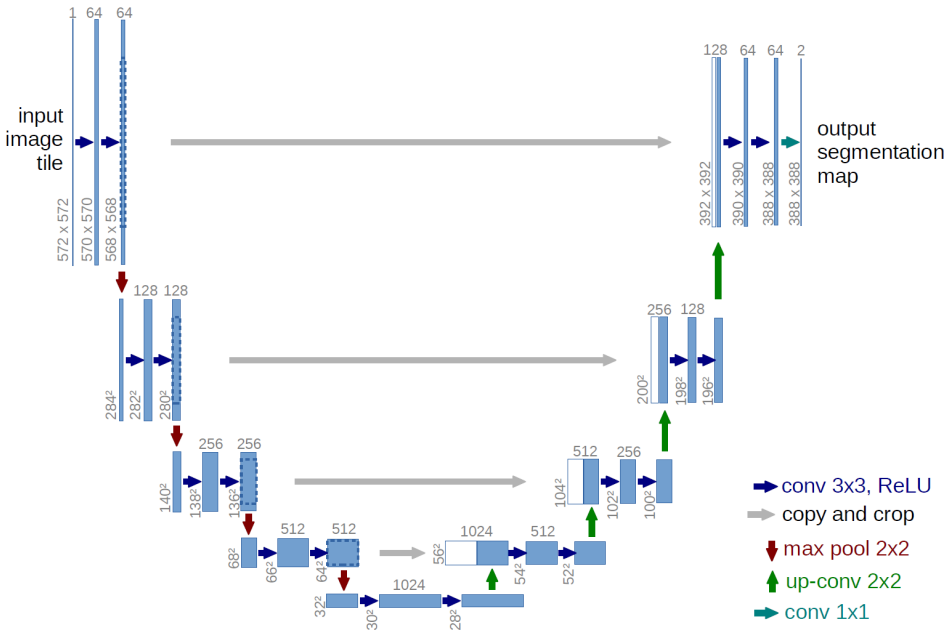

Figure 12: Visualization of the U-Net architecture

Figure 12 shows a particular implementation of an encoder-decoder model known as U-Net [27]. It has been successfully used on a variety of projects in and outside the medical field. The contracting side on the left shows a similar architecture to classification CNNs, after which the process is mirrored on the expanding side.

The following subchapters will discuss key elements that go into the design process of such an architecture. It was also analyzed how technologies that were introduced after the U-Net paper could improve the performance on this data set.

3.4.1 Channels, Growth Rate, and Depth

The number of parameters in a neural network has a high correlation with its learning capacity. By adding more nodes that can be adjusted during training, the model can approximate a more complex function that transforms the input into the output. The downside is that a larger parameter count will also increase the possibility of overfitting the data. A convention in the field of CNNs is to gradually increase the number of channels, while the spatial resolution is reduced due to the use of MaxPooling [21][25][26][27]. U-Net also shows this behavior on the left side of its architecture.

A test was set up that compared the original U-Net against five smaller variants. They varied in the total number of parameters and how the number of channels changed from one layer to the next.

Name	C1	C2	C3	C4	C5	C6	C7	C8	C9	Para.
Model A	32	32	32	32	32	32	32	32	32	195k
Model B	16	24	36	54	81	54	36	24	16	332k
Model C	48	48	48	48	48	48	48	48	48	437k
Model D	8	16	32	64	128	64	32	16	8	491k
Model E	32	48	72	108	159	108	72	48	32	1.33m
U-Net	64	128	256	512	1024	512	256	128	64	31.0m

Table 2: Convolutional blocks with their respective output channels

Training U-Net was very slow in two regards. Each training step lasted six times longer compared to the smallest model, while it also took 25 times more training steps to reach the same results. Since it was also overfitting to a significant amount, it was not trained until the end. The second largest model E also overfitted on the training data and never achieved comparable results to the other four models.

Model D started with eight channels which were then increased by a factor of 2, whereas B started out larger but only increased the channels by a rate of 1.5. Although being the smaller model, B showed a higher accuracy in the end. It was concluded that the initial number of output channels has a larger effect on the result than the growth rate.

This theory was supported by the results from A and C, both of which kept their initial number of channels throughout the network. These two showed the highest scores in comparison to the other models while maintaining their relatively small size. Since their results were identical, model A was kept because of its faster training speed. It was interesting to see that the smallest model performed best across the candidates.

The original U-Net featured a depth of 5 convolutional blocks until reaching the bottom of the "U"-shape. It was also investigated that a depth of 6 would not improve performance, while a depth of 4 decreased the accuracy. Therefore, Model A was left unchanged.

3.4.2 Skip Connections

U-Net uses skip connections to copy values from the left side to the corresponding layer on the right side. This improves performance because otherwise spatial information would get lost while contracting the input. An experiment showed that removing these paths indeed hurt the performance badly.

ResNet [25] is another popular classification architecture that was the first to use residual connections. These are another type of skip connection that bridge the beginning and end of a convolutional block. According to the original paper, they are also one of the main reasons CNNs can be expanded to depths of thousands of layers. Since their use in the industry has found great success in many models, they were also applied in this network. However, they did not have an impact on the performance or the training speed, which is why they were not used in the end.

3.4.3 Dropout

Dropout is a popular regularization technique that randomly zeros out a fraction of the nodes during training [35]. It is understood that this helps the model to generalize better and reduce overfitting on a given data set. Well known image classification architectures like VGG16 [19], SqueezeNet [26] or AlexNet [21] use Dropout near the end of the network. Similarly, U-Net uses Dropout at the end of the contracting path to prevent overfitting.

Since a single unit of dropout with a rate of 0.5 is common in other architectures, this was also chosen as a first option. Other tests included adding multiple dropout units between the convolutions on the contracting side, which led to slower training and lower scores. Adding dropout on the expanding side is rather unusual and also didn't perform well in the tests. In the end, the first candidate was chosen for future training.

3.4.4 Activation Functions

Activation functions sit between layers in a network to introduce a nonlinearity. Otherwise, the operations could be regenerated to a simple linear transformation and remove the benefits of building a deep model. The rectified linear unit or ReLU [36] is the default recommendation for an activation function in modern neural networks. It's defined as the maximum of 0 and the input value and can be described as a nonlinear function made up of two linear pieces. Because of this, it preserves many of the properties that make models easy to optimize and generalize well on new data [18].

$$relu(x) = \begin{cases} 0 & \text{if } x < 0 \\ x & \text{if } x \geq 0 \end{cases}$$

$$elu(x) = \begin{cases} e^x - 1 & \text{if } x < 0 \\ x & \text{if } x \geq 0 \end{cases}$$

Since the introduction of ReLU other variants have built on its success like PReLU [1], LReLU and ELU [37]. The latter titled exponential linear unit shows the same linear behavior for positive values but adds an exponential function to negative inputs instead of erasing its value. This has shown further improvements by keeping the mean of values centered at 0 while only being slightly more computationally expensive. ELU showed superior results in comparison to ReLU and was chosen for all future tests.

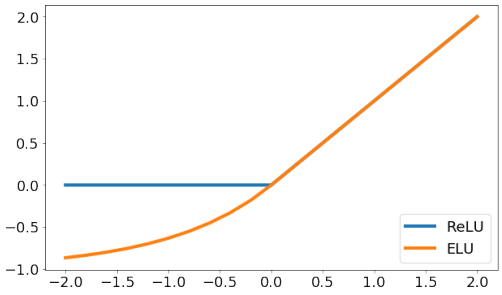

Figure 13: Visualization of ReLU and ELU

3.5 Training

The training is where the actual learning takes place, and its process is inspired by how humans improve their performance on tasks. When opposed to a difficult question the first step would be a mere guess of what the answer might be. In a second phase, this information is compared to the right answer and processed by the brain accordingly.

Similarly, the training of a neural network consists of two parts. In the first step, the input is fed forward through the network, and an answer prediction is made. In the very beginning, this is just a random guess because ANNs are initialized with random values.

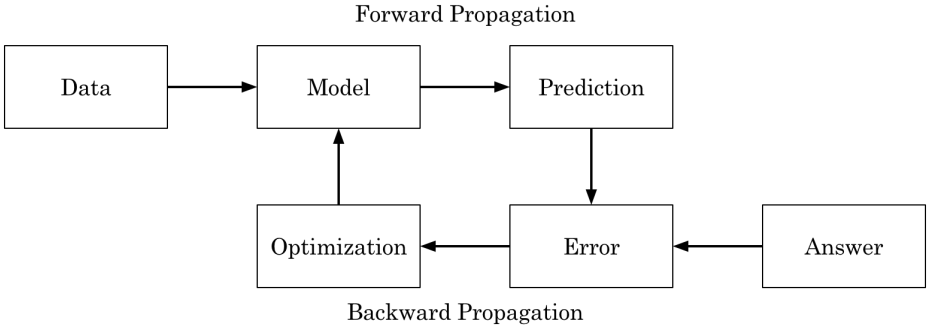

Figure 14: The iterative process of forward and backward propagation

The prediction is then compared to the right answer, and the error is calculated. This metric can be fed back through the network to optimize the parameters in the model. These two steps are called forward and backward propagation.

3.5.1 Metrics and Loss Functions

Metrics are used in deep learning to measure the performance of a model. For example, the accuracy is often chosen to describe how well a neural network is doing on a classification task. An accuracy of 0.9 indicates that 9 out of 10 samples are classified correctly.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

In the formula above T and F indicate whether a prediction was true or false. P and N stand for a positive or negative outcome.

The result of a loss function is a metric that will be minimized during the backward propagation process. It needs to be a differentiable function, which is why the accuracy cannot be used as a loss function. It is a binary metric that works with true or false values and not with probabilities.

In situations like these, a surrogate function is used that has a high correlation with the target metric. For classification problems, this is often the cross entropy. Because a segmentation can be seen as a classification for every output pixel, it was also chosen as a candidate for this study.

$$\text{Cross Entropy } (p, q) = -\sum\ p(x)\ log\ q(x)$$

Another option was the F_1 score, which is a particular implementation of the F-Measure when β is 1. It is also known as the dice similarity coefficient (DSC).

$$F_\beta \text{ Score} = \frac{(1+\beta^2)TP}{(1+\beta^2)TP + \beta^2 FN + FP}$$

Although the F_1 score is commonly applied as a binary measure and therefore not differentiable, a "soft" version can be used that accounts for continuous probabilities. To use it as a loss function where 0 describes the best possible outcome the F_1 loss was defined as $1 - F_1$ score.

$$F_\beta \text{ Loss} = 1 - F_\beta \text{ Score}$$

The value of β can be adjusted to change the emphasize between precision and recall. Precision describes how much of the predicted area was true, whereas recall describes how much of the ground truth was recognized by the model. This can be useful for data sets with large imbalances between classes, like this study showed between Fibula, Femur, and Tibia.

The F_2 score was tried for the three channel segmentation, where the Fibula could not be recognized. Although this model showed better scores regarding the recall measure, the Fibula was still not segmented.

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

$$\text{Total Error} = \frac{FP + FN}{TP + TN + FP + FN}$$

Both loss functions were used in separate runs to compare the results of the F_1 score and the cross entropy. The F_1 trained model showed better results regarding the F_1 score and the cross entropy model showed better results on the cross entropy loss. In regards to Precision, Recall, Accuracy, and Intersection-over-Union (IoU) the performance of the F_1 trained network showed higher scores.

$$\text{IoU} = \frac{TP}{TP + FP + FN}$$

Another test run used both loss function at the same time. This was done by feeding the output of the next-to-last layer into two separate segmentation outputs. Each of these last layers was trained with the F_1 loss and cross entropy loss respectively. All previous layers were therefore trained by the sum of both loss functions. This architecture matched the best results of the previous runs in a single model but also increased the training time by 30%.

Evaluating these two outputs on the other metrics played in favor for the F_1 loss output. Combining both maps didn't improve the scores any further. Since the cross entropy was chosen as a surrogate function and not as a performance metric, all future tests used only the F_1 loss. This delivered faster training than the dual output method.

3.5.2 Optimizer

The previous chapter provided an overview of the loss function, which measures the performance of a prediction during the training process. This chapter discusses how the result can be used to execute the actual optimization step.

The derivative of a single variable function defines the slope of this function at any given point. Knowing this, one can tell in which direction the original function declines. Adjusting the independent variable along the descent of a loss function will minimize the error in respect to its dependent variable.

The gradient is the derivative for functions of multi-dimensional inputs, such as the loss functions used in deep learning. A process that minimizes its result is called gradient descent. While it is possible to determine its minimum analytically, it is intractable for artificial neural networks due to the high number of parameters [17].

Instead, the stochastic gradient descent (SGD) is used which will take a random batch of the training data and iteratively adjust the parameters in small steps. SGD is the basis for all common ANNs, but over the years different variants were introduced to the field.

The Adam optimizer [38] is such a variant, which enhances SGD amongst other things by using what's called an "adaptive momentum estimation." This is also where its name derives from. It adaptively adjusts the learning rate which defines how much the parameters will be changed in one training step. By incorporating the previous and current shape of the slope, Adam can tremendously speed up the training.

3.5.3 Batch Size

The number of random samples per training step is referred to as the batch size. In the past, it was believed that larger batches led to something called the generalization gap [39], where the accuracy of a model would drop if it was trained on unusually large batches. Recent work [40] suggests other reasons for this decrease in accuracy. While common batch sizes range from 32 to 256, Goyal et al. showed accurate results using 8192 images per batch when training a model on ImageNet [41].

Using batches smaller than 32 samples can introduce a different kind of problem. Having too few data points that don't represent the mean of the data well, may lead to slow and unstable training.

Based on hardware limitations the largest possible batch sizes ranged from 24 to 48 samples depending on the current architecture. Whatever could be fit into memory was used for these experiments. Exceptions occurred when working with 3D convolutions in which case the batch size had to be limited to just 4 samples.

3.5.4 Learning Rate Policy

One full iteration over the training samples is referred to as an epoch, and the learning rate policy describes how the learning rate is changed from one epoch to another. With the introduction of adaptive optimizers like Adam, there has been a lower emphasize on this topic because the learning rate is modified during training. Even though this reduces the number of possible defects, training time can often be saved with the right initial learning rate.

Ten epochs were run at different learning rates to compare initial results and to examine the point at which the model wouldn't converge at all. 0.002 was the highest rate at which the model started training, but 0.001 resulted in the best score. Adding a decay that reduces the learning rate manually over time did not improve the results with the use of Adam.

3.5.5 Early Stopping

Neural networks will continuously minimize the loss on the training set. This result needs to be validated on data the network hasn't seen before. At a certain point during training, the performance on the validation set will start to decrease because the model is overfitting on the training data. The number of iterations to reach this point is dependent on many hyperparameters, as well as the random values the network has been initialized with. As such, it's difficult to calculate how long the training will need to reach its peak.

Early stopping is a simple technique that will end the training process as soon as the model stops improving on the validation data. To do this, a patience value is defined how long the network should continue training after the score has stopped increasing. This is important because not every epoch will lead to a new best score on the validation data.

For test runs in this study, a patience of 9 was selected, which meant the training would stop after 10 epochs without improvement. Depending on the architecture and other hyperparameters this point was reached after 30 to 60 epochs. For the last training with the final set of hyperparameters, the patience was increased to 19, which didn't improve the accuracy. This was also a verification that the initial value of 9 was a good fit for this problem.

3.6 Postprocessing

After the training had finished, it was investigated how the results on the validation data could be improved any further. Since the predicted segmentations showed a small number of values between 0 and 1, a threshold of 0.3 showed the best results to create a fully binary map. Furthermore, any 2D slice that showed a prediction smaller than 2% of the total frame was predicted as empty instead. This helped against the false prediction of tiny fractions in upper and lower layers. For the segmentation of the Fibula alone, this last step was not executed.

3.7 Summary

In summary, the architecture features 4 "Down Blocks" on the contracting side that are mirrored through 4 "Up Blocks" on the expanding side. Each level is connected to the other side to share information at the current spatial resolution.

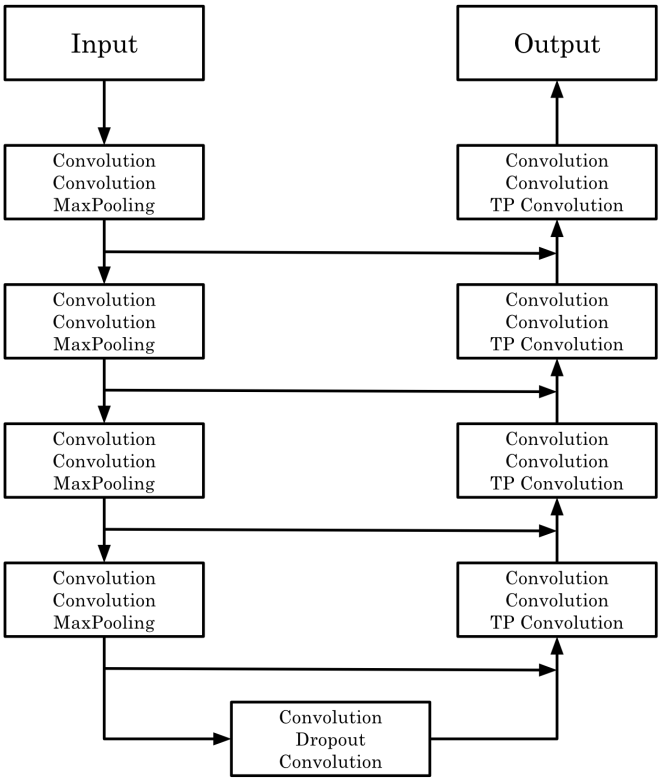

Figure 15: The proposed architecture for the segmentation

A Down Block consists of 2 convolutional layers each followed by an ELU activation function, after which A MaxPooling layer halves the width and height of the image. The Up Block looks similar but features a transposed convolutional layer in the beginning to upscale the resolution and no MaxPooling in the end. The Middle Block features a Dropout layer with a value of 0.5, which is surrounded by convolutional layers as well. All convolutional layers have 32 channels. The output layer includes another convolution that reduces the channels to 1, which represents the segmentation map.

Other options that were tried but didn't improve performance included using convolutional blocks with a stride of 2 instead of MaxPooling to down sample the images. Batch normalization was also applied but resulted in a high amount of overfitting the network never recovered from. The fact that it did not improve the performance came as a surprise because it usually improves any model. A possible reason for this could be the small batch size in combination with a high variance of intensities in the data. If the mean of each batch varies significantly from the mean of the entire training data, it may hurt the performance of batch normalization.

4 Results

The results in this chapter are based on data the network has not been trained with. It was also made sure that there was no overlap in images of people that were recorded multiple times. Five images were chosen randomly from each of the two sources in the very beginning. Since the Epi data featured 41 slices and the Jopp data featured 24 slices, 325 2D samples were available for evaluation. Each slice contained 50,176 pixel values, making a total of 16.3 million predictions that were analyzed.

The network architecture was developed using a single segmentation channel that merged Femur, Tibia and Fibula maps. However, tests were also run to validate the performance for each bone on its own and also combine the three separate predictions into a multi channel segmentation. The same architecture and training procedure was used for this task.

4.1 Numeric Evaluation

The proposed model achieves a DSC score of 98.0% and an IoU of 96.0%. Precision and Recall are perfectly balanced, suggesting that predictions are neither too optimistic or pessimistic. The error shows a small value of 1.2%.

	DSC	IoU	Precision	Recall	Error
Merged	0.980	0.960	0.980	0.980	0.012
Femur	0.981	0.963	0.979	0.984	0.006
Tibia	0.977	0.955	0.976	0.977	0.006
Fibula	0.953	0.911	0.954	0.952	0.001
Combined	0.979	0.958	0.977	0.981	0.004
Femur [10]	0.940	-	-	-	-
Tibia [10]	0.920	-	-	-	-
Tibia [11]	0.975	-	-	-	-

Table 3: Numeric evaluation of segmentations

Results on Femur and Tibia alone are comparable to the merged approach, whereas the Fibula segmentation shows lower scores. This could be because the Fibula is only visible in a minority of slices, making it a somewhat unbalanced task. Combining the three separate segmentations to a single model gives comparable results as well. The error is reduced by a factor of 3, which is expected because the channels are increased by 3. Compared to previous studies, the proposed model shows slightly better results than the multi-atlas segmentation and significantly higher scores than the ray casting technique.

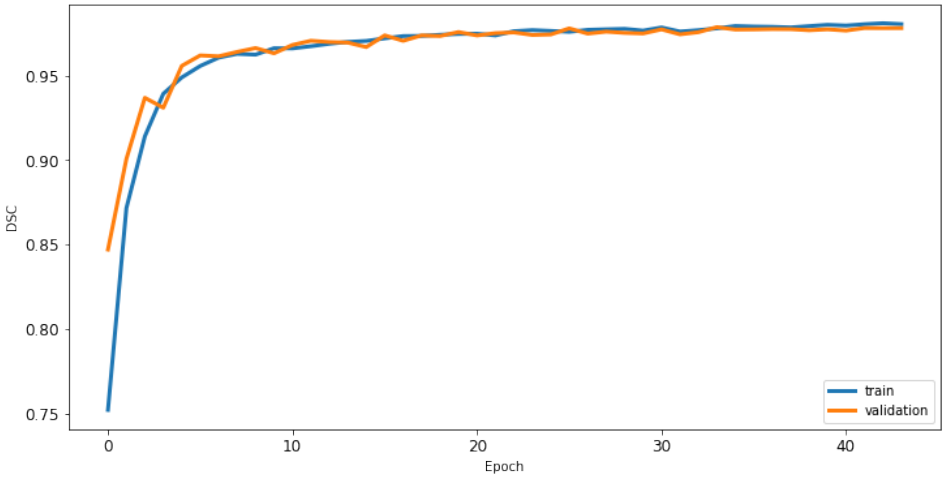

Figure 16: Visual representation of the performance during training

The graph above shows the performance of the merged model during the learning process on the training and validation data.

Data	Epoch 1	Epoch 5	Epoch 10	Epoch 20	Epoch 40
Training Set	0.753	0.949	0.966	0.975	0.981
Validation Set	0.847	0.956	0.963	0.976	0.979
Test Set	-	-	-	-	0.980

Table 4: Performance of the merged model on different sets over time

The 98% mark is never reached on the validation data, but only on the final evaluation of the test set. Towards the end, the performance of the training data pulls slightly ahead of the validation results.

4.2 Visual Evaluation

The visual evaluation will focus on the results of the combined network since its performance is on par with the one channel model while offering more information about the associated bones.

The network shows excellent performance on slices that are located near the middle of the 3D MRIs often with DSC scores of over 99%.

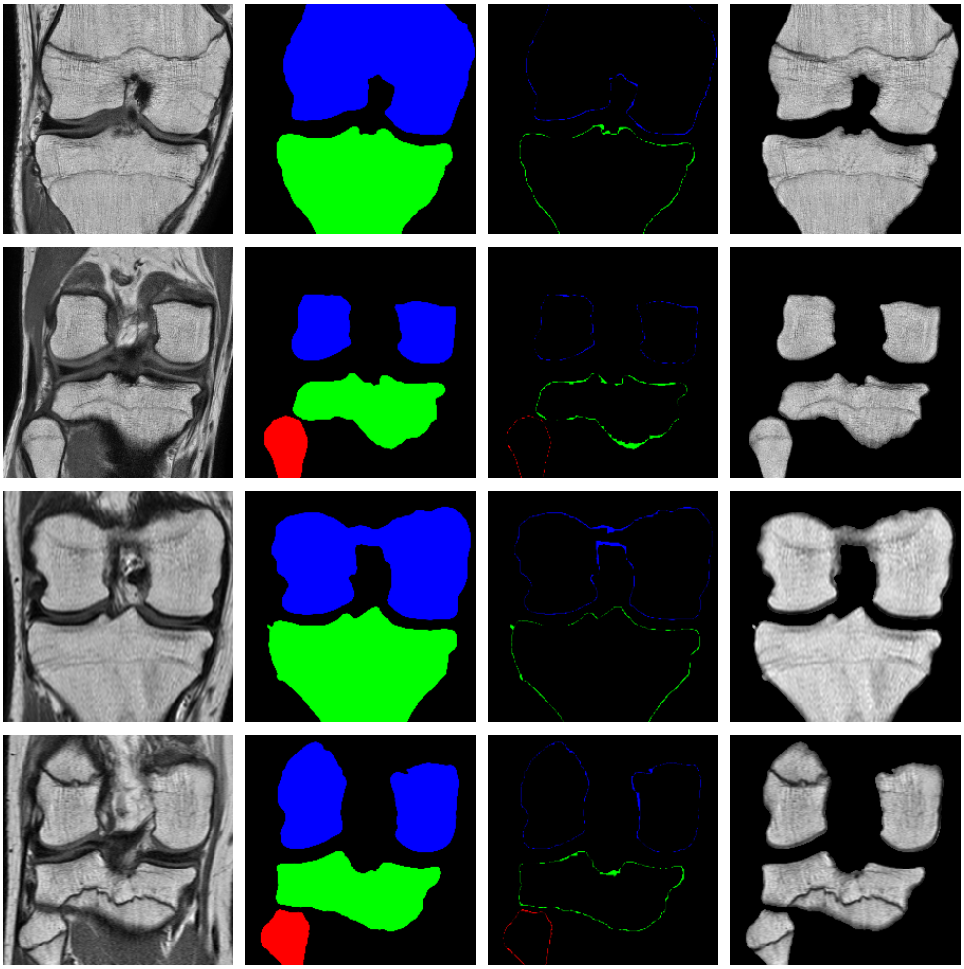

Figure 17: Input, prediction, difference to ground truth and applied mask (from left to right)

Perfect DSC scores of 100% are achieved through empty segmentations that the network correctly predicted as such.

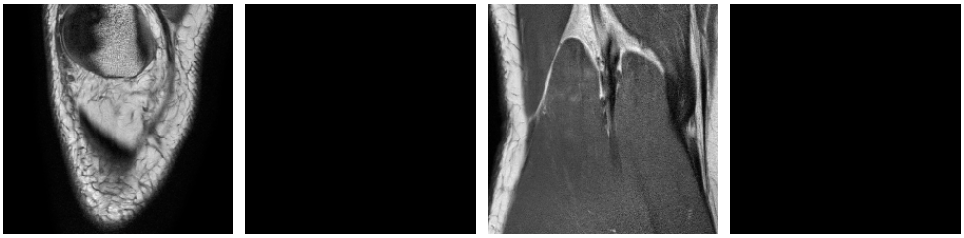

Figure 18: Correctly segmented upper and lower slices

The most imprecise results showed a DSC of 0% where ground truth and prediction did not overlap at all.

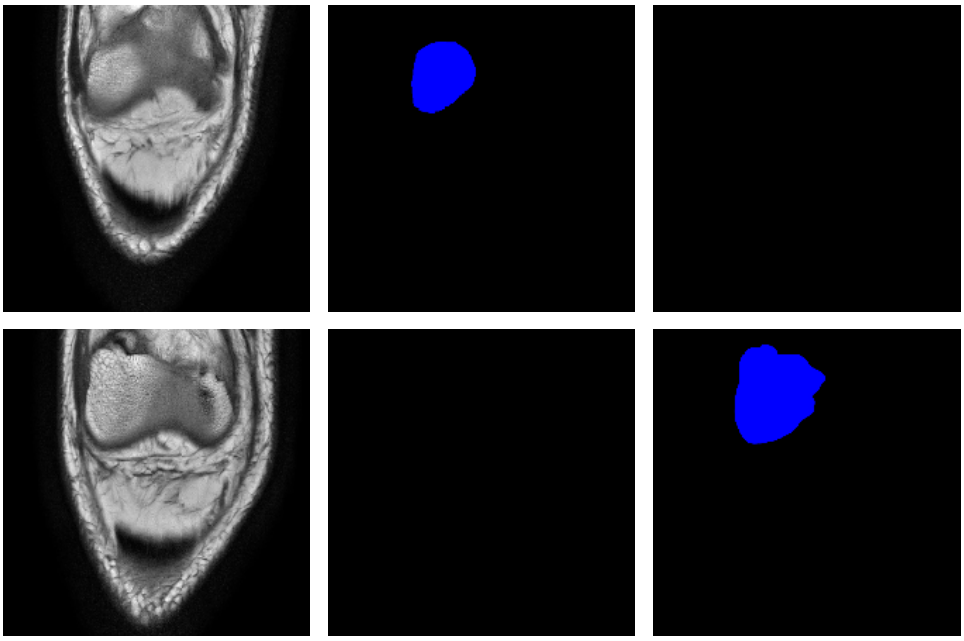

Figure 19: Input, ground truth and prediction of false segmentations

In these examples, the ground truth defines the Femur to be visible in the upper image but not in the one below it. The prediction disagrees with it in both cases. These are two examples that could be seen as noise in the ground truth data or to be debatable at least.

4.3 Model Exploration

Neural networks are often considered black boxes because it is difficult to get an intuitive understanding how decisions are calculated. This is a crucial problem especially in the medical field, where a prediction may approximate certain health conditions about a patient.

Two factors help in this study to pull away from this problem. Firstly, a segmentation is an image to image pipeline making it less abstract what kind of transformations are happening. Visualizing input and output gives an intuitive understanding how one may have emerged out of the other.

Secondly, convolutional neural networks take away most of this black box reasoning [17] because every layer in the network can be visualized. The following images give an insight what a sample looks like after different convolutional blocks.

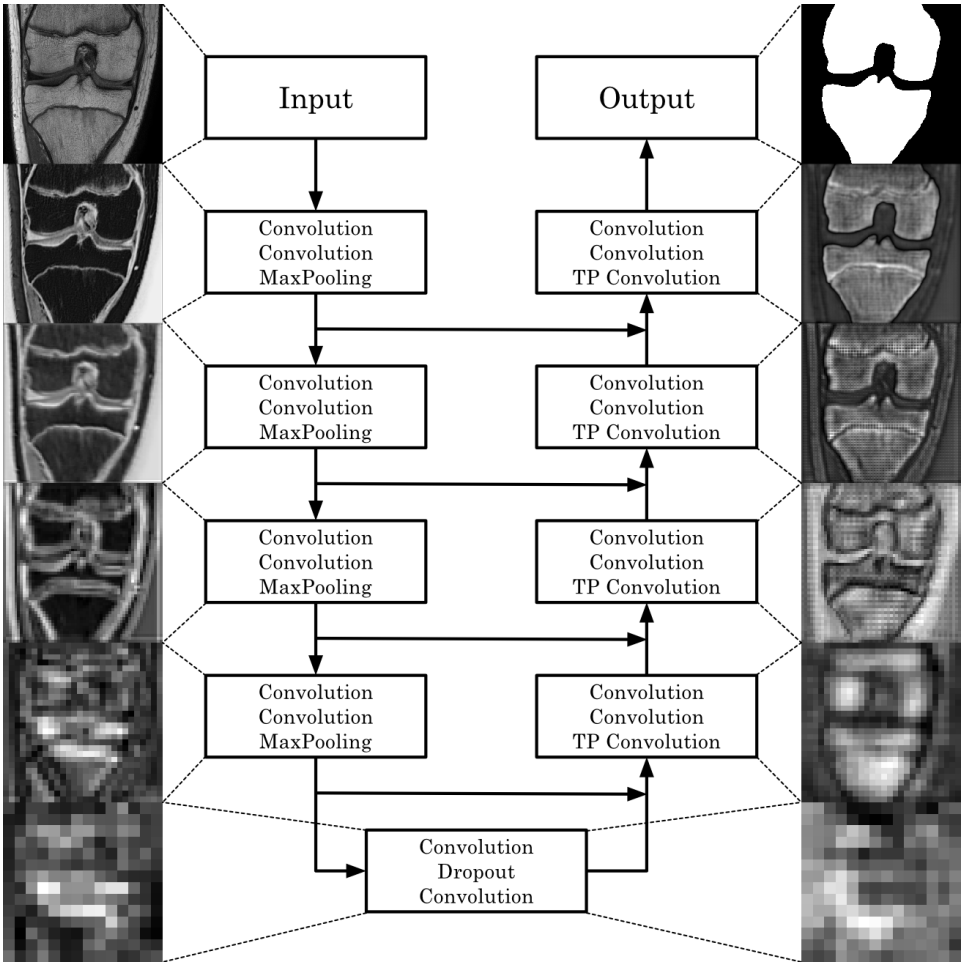

Figure 20: Intermediate sum of feature maps throughout the network

Each image shows the sum of all intermediate channels at a particular layer. The first image is the raw input, whereas the last image is the final segmentation map. One can clearly see the reduction of resolution towards the middle, which is then brought back up. The first convolutional block inverses the input and removes most of the bone structure except for the growth plates. The second to last output looks similar to the input, but the skin on the sides is almost entirely removed, and the dark growth plates have been filled. Looking closer at this image one can also notice a crisp black line that separates the bone from the rest. A simple threshold at this point would segment the bone fairly accurate.

Even at this level, it is difficult to judge what exactly the network is detecting. For another visualization, the intermediate channels are not added like before but analyzed separately.

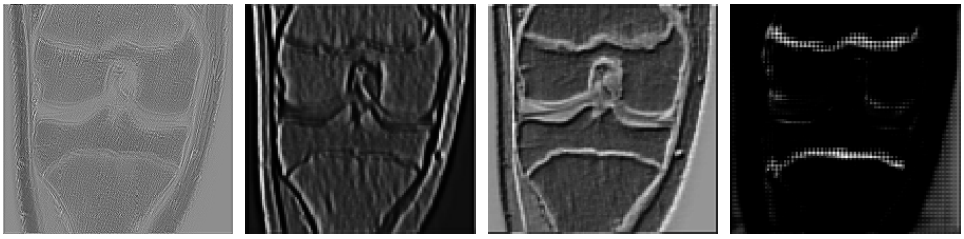

Figure 21: Examples of intermediate channels throughout the network

These are a few notable examples in the model. The first filter detects high frequencies in the bone and skin tissue. Filter 2 finds vertical edges while filter 3 shows horizontal ones. Finally, filter 4 appears to be a growth plate detector. It does not just detect horizontal edges, but only those inside the bone. This seems reasonable because the network needs to learn that the dark growth plates should not be interpreted as edges.

4.4 Noise Exploitation

Ground truth data is often subject to noise, because mistakes happen during their creation or because it may be debatable what the actual truth is. To test how well the network performs on noisy data, another training environment was set up where synthetic noise was added to the ground truth segmentations. By using erosion and dilation with a kernel size of 7 both the training and validation sets were modified.

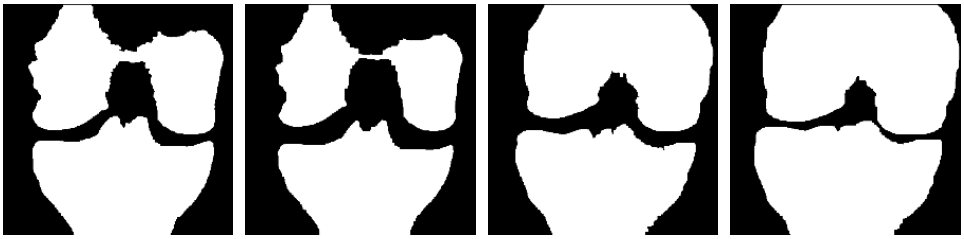

Figure 22: Examples of synthetic noise using erosion and dilation

The first and third images show the original segmentations, while two and four were changed using erosion and dilation respectively. The noise on the training data accounted for an error of 7.1% DSC. The predictions show an accuracy of 95.2% DSC on the unmodified test set, meaning that the noise hurts the performance of the model. More importantly, the DSC error is lower than the error that was introduced by the noise. When applying the same noise on the test set and using it as a prediction, the score is 92.8% DSC. The predictions of the network contain 33% less noise than the data it was trained on and show the robustness against inaccurate labels.

4.5 Transfer Application

Neural networks are known to be unreliable when used on data that exceeds the range of variation in the training set. They cannot learn what they were not taught. On the other hand, convolutions are translation invariant, allowing them to recognize patterns anywhere in the frame [17]. Another merged model was trained that used the same architecture but added further image augmentation of horizontal and vertical flips. This way it only reached a DSC score of 97.3% but was more flexible to structural differences in the image.

The proposed architecture can be described as "fully convolutional" because it does not include any dense layers. This allows the network to accept any resolution and still being able to process it. An experiment was set up that used uncropped images from the Epi data which were resized to 448 x 448 pixels. This made them four times larger than the images the network was trained with.

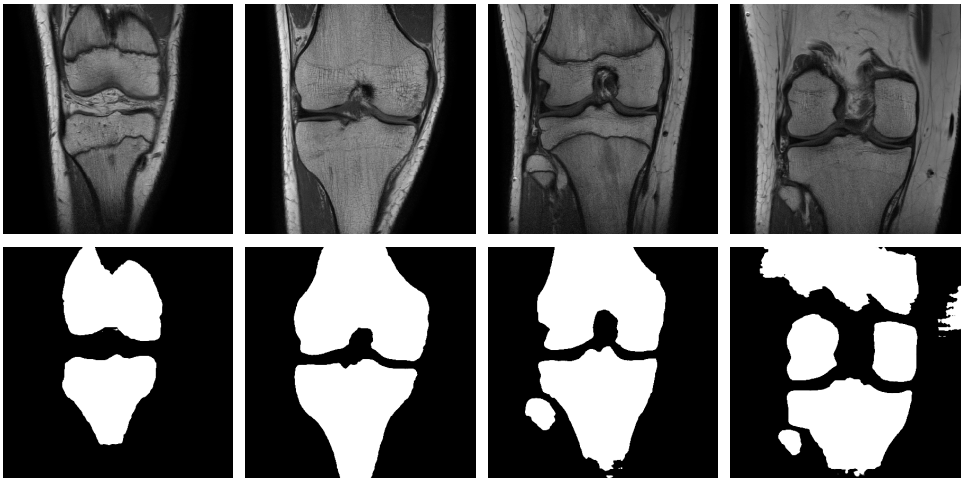

Figure 23: Examples of uncropped and 448 x 448 pixel predictions

The predictions show good results even following the shaft of the bone which wasn't visible in the original cropped images. Even large intensity gaps through growth plates are recognized as Femur or Tibia. The recall performance is very accurate. Problems are visible in tissue that isn't bone but was recognized as such as the 4th example shows.

In 3.2 a third data source was mentioned that featured five sagittal recordings of knees. These were not used for the training because of their structural differences and because no ground truth segmentations were available. The following samples show what happens when the network is applied to images from a different perspective.

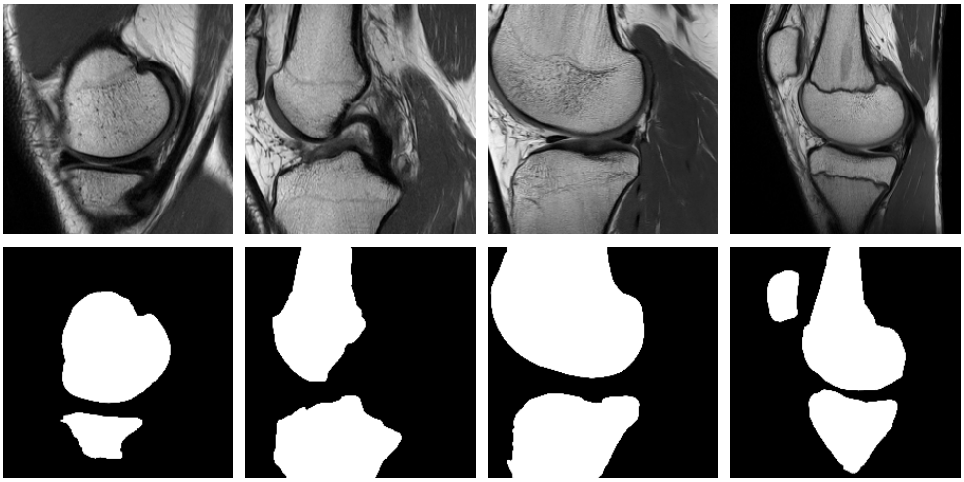

Figure 24: Examples of sagittal class 3 data segmentations

These results look very accurate. In example 4 it even recognizes the Patella, a bone it was never trained on. This shows that with enough image augmentation the network will learn to segment any bone.

4.6 Proof of Concept: Age Assessment

The initial cause for the segmentation experiment was to reduce the amount of information in a knee MRI. The resulting images could then be used to make age assessments that focussed on the bone and the growth plate. This section will briefly cover such an age prediction pipeline.

The age of the candidates ranged from 14 to 21 years with a mean at 17.5 years. Predicting this age for every person meant that it was never off more than 3.5 years. Since the data was normally distributed, a static prediction of the mean led to a mean difference of 1.2 years.

Even by using many of the techniques mentioned in previous chapters, it was not possible to train a stable model that could beat this baseline on the raw input data. After the segmentation maps had been applied to all 145 3D samples, many of the slices were completely empty because they did not contain any of the three bones. It was decided to only use the middle 18 slices from each of the two sources. Even with these changes, the network would not converge.

Only after using the contracting side of the proposed model with the parameters it learned on the segmentation task was it possible to train a network in a stable manner. This approach did not make the model converge every time, but when it did its predictions were stable through multiple epochs. A global

average pooling layer and a dense layer were added after the convolutional block in the middle. The output was a single continuous value representing the age of the individual.

The results on the validation data showed a mean difference of 0.64 $\pm$ 0.48 years for all of the slices. Since every 3D sample now had 18 different predictions, another random forest regressor was built that would take a vector of 18 values and predict a single age. This resulted in a mean difference of 0.56 $\pm$ 0.40 years on the validation data, and a final run on the test set showed 0.48 $\pm$ 0.32 years. The most accurate assessment of the 14 test samples was off by 0.07 years, and the worst estimate showed a difference of 1.37 years.

These results show higher accuracy than the work of Stern et al. [12] using carpal MRIs which led to a mean difference of 0.85 $\pm$ 0.58 years. Their approach extracted 11 growth plates in the left hand of candidates and then mapped these features to an age estimate. It was not based on convolutional neural networks but a random forest regressor for both stages. Their data set was equally distributed resulting in a higher standard deviation for the age of candidates. However, their worst age estimate was off by 2.35 years in comparison to the 1.37 years mentioned above. This suggests that the normal distribution of this study is not the only cause for better results.

5 Discussion

Convolutional neural networks proved to be a very effective measure for the segmentation of bones in MRIs. They show better performance than the ray casting technique and multi-atlas registration which were used in previous studies. However, the underlying data sets were only similar and not identical, which could have also led to differences in the results.

The hypothesis that the reduction of data could enable the age assessment through neural networks showed to be true based on a brief test in the previous chapter. More importantly, it supports the work of Säring et al. [7] that the status of growth plates is an indicator of the age to the border of adulthood.

The greatest limitation of this study was time. Three weeks were spent on the programming aspect and trying out different sets of parameters. Each new idea came with several hours of training to verify the performance. As such, only popular techniques in the deep learning field were applied. Another three weeks were invested in writing this thesis to complete everything by the deadline. Spending more time on data analysis, fine tuning parameters and trying entirely new architectures might open up new possibilities. All of the tests were based on encoder-decoder architectures, which means there might be new advancements to be made outside this scope.

Lately, a new wave of convolutional layers has been getting much attention. Depthwise separable convolutions and dilated convolutions are used more and more frequently in CNNs. Both can increase the efficiency of parameters and are especially popular on mobile devices and real time applications. They were briefly applied in this study as a replacement for some convolutional layers without making a difference. However, more time could be spent on restructuring the entire architecture to benefit the needs of these types of convolutions.

Adding noise to the data hurts the performance of the model. However, the resulting error of the predictions was smaller than the error that was introduced through the noise itself. Creating multiple ground truth segmentations by hand and averaging the results, could present a way of improving the findings in this thesis. This would further reduce the noise in the data and allow the model to move beyond the 98% mark. Since the smallest architecture achieved the highest score, the challenge of improving the performance is likely not limited by the capacity of the network.

It was interesting to see that the architecture using 3D convolutions performed worse than the 2D version. This is also something Prasoon et al. [42] found when applying CNNs to the segmentations of cartilages in the knee. They were able to achieve even better results using a triplanar CNN that worked with 2D slices from each of the three perspectives. 2D CNNs also have the benefit of being faster to train and not needing volumetric images to perform their work.

Although processing speed was no priority, the architecture consists of only 195,000 parameters making it 150 times smaller than U-Net. On a sub-$1000 workstation, it can be trained from scratch in 1 to 3 hours depending if the bones need to be separated. Using the pre-trained weights, one can likely apply transfer learning with minimal effort to solve similar problems.

It is to be investigated more thoroughly how the segmentation maps can be used on age related estimates to improve the results further. Additional information about the patients like height and weight may help to improve accuracy in the future.

6 Conclusion

This thesis presented the development of a fully automated workflow based on convolutional neural networks, which segments bones in three dimensional MRI data of human knees. It shows an excellent performance of 98% DSC while distinguishing between Femur, Tibia, and Fibula. The network even shows accurate results on sagittal images although it was entirely trained using coronal data. It is robust against noise and adaptable to changes in resolution due to its fully convolutional structure. This also helps to visualize all of the layers in the network. Furthermore, it was possible to combine the segmentations with part of the architecture as a pre-trained model to assess the age of candidates with a mean difference of 0.48 years $\pm$ 0.32. All of this is possible using noninvasive magnetic resonance imaging.

References

[1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. feb 2015.

[2] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. sep 2016.

[3] Mariusz Bojarski, Philip Yeres, Anna Choromanska, Krzysztof Choromanski, Bernhard Firner, Lawrence Jackel, and Urs Muller. Explaining How a Deep Neural Network Trained with End-to-End Learning Steers a Car. apr 2017.

[4] Joshua Saxe and Konstantin Berlin. Deep Neural Network Based Malware Detection Using Two Dimensional Binary Program Features. aug 2015.

[5] European Asylum Support Office. Age assessment practice in Europe. 2013.

[6] World Health Organization. Ionizing radiation, health effects and protective measures, 2016.

[7] Dennis Säring, Markus Mauer, and Eilin Jopp. Klassifikation des Verschlussgrades der Epiphyse der proximalen Tibia zur Altersbestimmung. pages 60–65. Springer, Berlin, Heidelberg, 2014.

[8] Eilin. Jopp. *Methoden zur Alters- und Geschlechtsbestimmung auf dem Pruefstand - eine rechtsmedizinische empirische Studie*. Kovac, 2007.

[9] R Setiono and H Liu. Neural-network feature selector. *IEEE Transactions on Neural Networks*, 8(3):654–662, may 1997.

[10] Pierre Dodin, Johanne Martel-Pelletier, Jean-Pierre Pelletier, and Francois Abram. A fully automated human knee 3D MRI bone segmentation using the ray casting technique. pages 1413–1424, 2011.

[11] Erik B Dam, Martin Lillholm, Joselene Marques, and Mads Nielsen. Automatic segmentation of high-and low-field knee MRIs using knee image quantification with data from the osteoarthritis initiative.

[12] Darko Stern, Thomas Ebner, Horst Bischof, Sabine Grassegger, Thomas Ehammer, and Martin Urschler. Fully automatic bone age estimation from left hand MR images. *Medical image computing and computer-assisted intervention : MICCAI ... International Conference on Medical Image Computing and Computer-Assisted Intervention*, 17(Pt 2):220–227, 2014.

[13] Catherine Westbrook. *MRI at A Glance*. Wiley Blackwell, 3 edition, 2016.

[14] Gerhard Aumüller, Gabriela Aust, Andreas Doll, Jürgen Engele, Joachim Kirsch, Siegfried Mense, and Laurenz Wurzinger. *Anatomie*. Number 2. 2010.

[15] David E. Attarian. Your Bones: What are growth plates?, 2013.

[16] Markus Auf der Mauer. *Automated Quantification of the Growth Plate of the Proximal Tibia for the Age Assessment in 3D MR Images Using a Fuzzy-Logic Classification Approach*. PhD thesis, 2015.

[17] Francois Chollet. *Deep Learning With Python*, volume 1. 2017.

[18] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. The MIT Press, 2016.

[19] Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. sep 2014.

[20] Y. Le Cun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, and W. Hubbard. Handwritten Digit Recognition with a Back-Propagation Network. *Advances in Neural Information Processing Systems*, pages 396–404, 1990.

[21] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet Classification with Deep Convolutional Neural Networks.

[22] NVIDIA. GPU vs CPU? What is GPU Computing?

[23] Nicholas J. Tustison, Brian B. Avants, Philip A. Cook, Yuanjie Zheng, Alexander Egan, Paul A. Yushkevich, and James C. Gee. N4ITK: Improved N3 bias correction. *IEEE Transactions on Medical Imaging*, 29(6):1310–1320, 2010.

[24] Özgün Çiçek, Ahmed Abdulkadir, Soeren S. Lienkamp, Thomas Brox, and Olaf Ronneberger. 3D U-net: Learning dense volumetric segmentation from sparse annotation. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 9901 LNCS, pages 424–432, jun 2016.

[25] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. dec 2015.

[26] Forrest N. Iandola, Song Han, Matthew W. Moskewicz, Khalid Ashraf, William J. Dally, and Kurt Keutzer. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and. feb 2016.

[27] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. *Miccai*, pages 234–241, may 2015.

[28] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking Atrous Convolution for Semantic Image Segmentation. jun 2017.

[29] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid Scene Parsing Network. dec 2016.

[30] Guosheng Lin, Anton Milan, Chunhua Shen, and Ian Reid. RefineNet: Multi-Path Refinement Networks for High-Resolution Semantic Segmentation. nov 2016.

[31] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. jun 2016.

[32] Vladimir Nekrasov, Janghoon Ju, and Jaesik Choi. Global Deconvolutional Networks for Semantic Segmentation. feb 2016.

[33] Fausto Milletari, Nassir Navab, and Seyed Ahmad Ahmadi. V-Net: Fully convolutional neural networks for volumetric medical image segmentation. In *Proceedings - 2016 4th International Conference on 3D Vision, 3DV 2016*, pages 565–571, jun 2016.

[34] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation. nov 2015.

[35] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15:1929–1958, 2014.

[36] Vinod Nair and Geoffrey E Hinton. Rectified Linear Units Improve Restricted Boltzmann Machines.

[37] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs). nov 2015.

[38] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. dec 2014.

[39] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On Large-Batch Training for Deep Learning: Generalization Gap and Sharp Minima. sep 2016.

[40] Elad Hoffer, Itay Hubara, and Daniel Soudry. Train longer, generalize better: closing the generalization gap in large batch training of neural networks. may 2017.

[41] Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour. jun 2017.

[42] Adhish Prasoon, Kersten Petersen, Christian Igel, François Francois Lauze, Erik Dam, and Mads Nielsen. Deep Feature Learning for Knee Cartilage Segmentation Using a Triplanar Convolutional Neural Network. 2013.

Statutory Declaration

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 marked all material which has been quoted either literally or by content from the used sources.

This thesis was not previously presented to another examination board and has not been published.

.....

City, Date Signature