Deep Learning to Predict Hip Fracture Risk from Clinical DXA-images

Meral Husein and Karin Yip

Lund, November 2018

LUND UNIVERSITY

Master's Thesis in Biomedical Engineering

Faculty of Engineering, LTH
Department of Biomedical Engineering

Supervisors: Hanna Isaksson, Lorenzo Grassi

Abstract

Osteoporosis is a bone disease that is defined as low bone mineral density (BMD) and results in an increased risk of bone fracture. It is a serious public health problem that causes excess mortality and major economical and social impact. Today there are more than 8.9 million fractures connected to low bone mineral density worldwide. Hip fractures are of particular concern since they are associated with excess mortality as high as 18-33%. The bone mineral density can be assessed with a number of different techniques. One of those is dual energy X-ray absorptiometry (DXA) which also is the most widely used clinical tool.

An arising field with many possible applications is the field within artificial intelligence (AI). Within medicine, smarter tools to ease diagnostics are developed with the help of AI. In this thesis project, a subfield of AI – deep learning and artificial neural networks – is explored to investigate whether an artificial neural network would be able to predict the hip fracture risk. The predictions are based on finding features of DXA-images that might indicate an increased risk for fractures. Two approaches of implementing artificial neural networks are carried out: developing a custom network and using transfer learning on a pre-trained ResNet network.

Different network architectures were investigated and developed. Experimentation involved both adding different network layers in multiple combinations and tuning of hyperparameters of the networks. The results from the networks were compared to the area under curve (AUC) obtained from the receiver operating characteristic (ROC) based on the BMD from the Malmö cohort. The AUC was 0.7497. The best AUC-value for the custom network was 0.7821, which is better than the AUC based on the BMD. The best AUC for the tuned ResNet-model was 0.6277, which is worse. The results in this thesis indicate that further exploration of using artificial neural networks as part of a diagnostic tool should be done.

Acknowledgement

We would like to thank our supervisors Hanna Isaksson and Lorenzo Grassi – our light sources in the darkness; our advisor in deep learning and neural networks Mattias Ohlsson – for all your insights and brilliant advice; and our consultant Sami Väänänen – the one who has been a helping hand in the beginning of the project, the one that got us all started. Also, a special thanks to Hannicka Sahlstedt who provided the script for and taught us how to read the images we were working with.

And of course, thank you all in the Biomechanics group that made us feel like a part of a bigger family during our time at your, our, place. We are very glad that you followed us on our journey from confused Master Students to confused Engineers in Mathematics. And all the other people we have met along the way, all the supportive voices and the interesting coffee breaks that you at BMC D13 have offered us – thank you!

Till we meet again...

List of Abbreviations

ANN	Artificial Neural Network
AUC	Area Under Curve
BMD	Bone Mineral Density
CNN	Convolutional Neural Network
DXA	Dual energy X-ray Absorptiometry
GPU	Graphics Processing Unit
MrOS	Mister Osteoporosis, a longitudinal cohort study in elderly men
PCA	Principal Component Analysis
ROC	Receiver Operating Characteristic
ReLU	Rectified Linear Unit
ROI	Region of Interest
ResNet	Residual Network
SNE	Stochastic Neighbour Embedding
t-SNE	(Student's) t-distributed Stochastic Neighbour Embedding
TF	TensorFlow

Contents

1. Introduction and aim

In this chapter, the background of this thesis project is presented. The objective and research questions are also stated.

1.1 Background

Bone is constantly being remodelled. The remodelling is carried out by two types of bone cells: osteoblasts which are responsible for the bone formation and osteoclasts which are responsible for the bone resorption. When the resorption rate is higher than the formation rate the bone mineral density (BMD) starts to decrease [1; 2]. This makes the bone less stiff, more brittle and increases the fracture risk. There is a number of different techniques to assess the BMD, but the most widely used clinical tool is dual energy X-ray absorptiometry (DXA). Prediction of the fracture risk can be obtained by using DXA-images (see Figure 1.1 for an example) from the femoral neck. A BMD of 2.5 standard deviation or more below a young female adult's mean is defined as osteoporosis, according to the WHO criteria. A BMD-value that is 1-2.5 standard deviation below that mean is defined as osteopenia [3]. The standard deviation from that mean is called T-score in clinical terms.

Osteoporosis is a disease where there is a reduction in BMD, which leads to an increased fracture risk due to more fragile and porous bone. It is considered a silent disease because often there are no symptoms before the first fracture. One aspect of the disease is the major economic and social impacts it causes [3; 4]. Worldwide there is a fracture associated with low BMD, a so called osteoporotic fracture, every three seconds which results in more than 8.9 million fractures each year. One in three women and one in five men over age 50 will experience osteoporotic fractures [5]. Osteoporotic fractures are associated with an increased risk for mortality which makes it a serious public health problem. Hip fractures, fractures in the proximal end of the femur, near the hip joint, are of particular concern since they are associated with excess mortality as high as 18-33% in the first year and can persist for five

years. They also cause morbidity with functional recovery limited to less than 50% [6]. About 30-50% of the actual osteoporotic fractures can be predicted based on the BMD-value only [7]. More accurate tools to identify and diagnose patients with an increased fracture risk is hence of interest. An illustration of the proximal part of the femur can be found in Figure 1.2.

Figure 1.1: Example of a DXA-image of the proximal femur that will be fed to the artificial neural network.

Figure 1.2: The proximal part of femur. A fracture in this area, close to the hip joint, is called a hip fracture.

During recent years, deep learning has broken new grounds within the field of artificial intelligence. The technique of deep learning has performed especially well in image segmentation and image classification [8]. Deep learning in short means that an artificial neural network (ANN) is developed. The network will be trained, i.e. exposed to thousands of images within the classes of interest for classification. All the images have a ground truth, a key, with the correct answer of to which class they belong. After training, the ANN will learn to find features of the images presented to it. Using these learnt features, the network will be able to perform more precise classification on images that it has not been exposed to [9].

A continual challenge when developing an ANN for classification tasks is the need of training data. For image classification, tens of thousands of images are required in order to train a classifier that will be able to perform a reasonable classification on new data. Even if the problem might be simple, the more images that can be provided, the more advantageous is it for the ANN. For example, a classic problem for a beginner in ANN is to classify handwritten digits. This is considered as a simple task, but still, there are

around 60 000 images in the training set. This large number of images tries to make sure that most of the variation of the handwritten digits is covered [10]. Nonetheless, often the required amount of images is not met.

With limited amount of images, developing a custom network is not always recommended; the preferred approach is instead to use transfer learning. In transfer learning, a well-performing image classifier is tuned to suit the problem of interest. If, for example, a task is to develop a classifier to distinguish between a bee and a chair, then these two new classes can be added to the classes of a pre-trained ANN. The ANN might already be able to differ between a bird and a sofa, and will therefore learn to differ a bee from a chair easier.

When training a network to learn two different classes, it is also important to have the same amount of images from both classes. Imbalance of data might be a disadvantage for the class with the smaller amount of images. This is because the ANN might not consider the smaller class as important and therefore not learn any special features of that class.

If there are any features in the DXA-images that indicate increased fracture risk and the ANN succeeds to find them, physical and pharmacological treatment can be applied. The risk of suffering from a fracture might then be reduced.

1.2 Related work

Deep learning is a fairly new addition to the field of orthopaedics. The idea explored in this thesis came from an article by Tiulpin et al. [11]. They used an ANN approach to implement a computer-aided diagnosis method for automatically diagnosing osteoarthritis in the knee. The authors used a siamese neural network, a network that combines two different input data, to classify plain radiographs into different stages of osteoarthritis, according to the Kellgren-Lawrence grading scale.

There have also been successful attempts to identify the risk factors for osteoporotic fractures with an ANN approach. For example, Tseng et al. [6] found that ANN outperforms conditional logistic regression at finding risk factors to hip fractures, both when it comes to discrimination and calibration of the factors. Ho-Le et al. [12] have also come to the conclusion that an ANN is good at assessing the contribution of different combinations of risk factors. This is also verified by Liu et al. [13], who concluded that the combined effect of different risk factors is easier detected and assessed by an ANN, in comparison to statistical models. Of the 74 risk factors that Liu et

al. analysed in their study, the ANN was able to find the top ten contributing factors amongst them.

1.3 Objective and research questions

The main purpose of this project is to investigate if the hip fracture risk can be predicted by analysing DXA-images of the proximal femur, using an ANN approach. The specific research questions we aim to answer are:

- Can an artificial neural network be used to predict fracture risk in the proximal femur from DXA-images?
- Is it better to use a pre-trained network or to develop a custom network?
- Is the amount of images we have enough?
- Is the number of fractures in the database enough?

1.4 Authors' contribution

The DXA-images and patient databases used in this project were provided by the supervisors Hanna Isaksson and Lorenzo Grassi in collaboration with Magnus Karlsson. Processing of the patient database and extracting information from it was done by Karin Yip, with help of a script based on code provided by Sami Väänänen. Hannicka Sahlstedt provided the code to translate the raw pixel values in the DXA-images to BMD. The pre-processing of the images was carried out by Meral Husein.

There have been two neural networks developed under the supervision of Mattias Ohlsson. Meral Husein has been focusing on a network for transfer learning; Karin Yip has worked on developing a custom network from scratch. Mattias Ohlsson has also helped the authors with interpreting the outputs of the networks. The work of this thesis and the writing of this report have been evenly split among the two authors. The report has also been revised after suggestions by the project supervisors.

2. Theory

This chapter describes the theory behind the methods and implementation for this thesis. First, the architecture of an ANN is described along with some of the challenges and strategies to handle them. Then, a background for transfer learning is given. Last, some theory for data visualisation is presented.

2.1 Deep learning and artificial neural networks

An ANN consists of several layers: an input layer, an output layer and sometimes hidden layers. The hidden layers' input and output are not visible for the programmer, and hence the name. See Figure 2.1 for an example of an ANN and its main components. Depending on the problem the network is to solve, the number of hidden layers may differ. Each layer consists of a number of neurons. There are several variants of neural networks. The most common ones are convolutional neural networks (CNN) and recurrent neural networks. The characteristics of a CNN is described later in this section. A recurrent neural network is a feedback network, i.e. the output of a layer may be fed into the same layer again for further processing [9]. A CNN or a recurrent neural network can also be referred to as a deep neural network; a deep neural network is a network that has two or more hidden layers [14].

2.1.1 Network architecture

The different layers in a neural network all have different properties. The input layer refers to the image itself. The output layer is the layer that states to which class the image belongs. All hidden layers are used to find the different features of a certain image and to e.g. classify or identify objects in it, but the following theory section will assume that ANNs only classify images. A CNN is also a kind of feed-forward neural network, i.e. the image is processed layer by layer until it reaches the output layer.

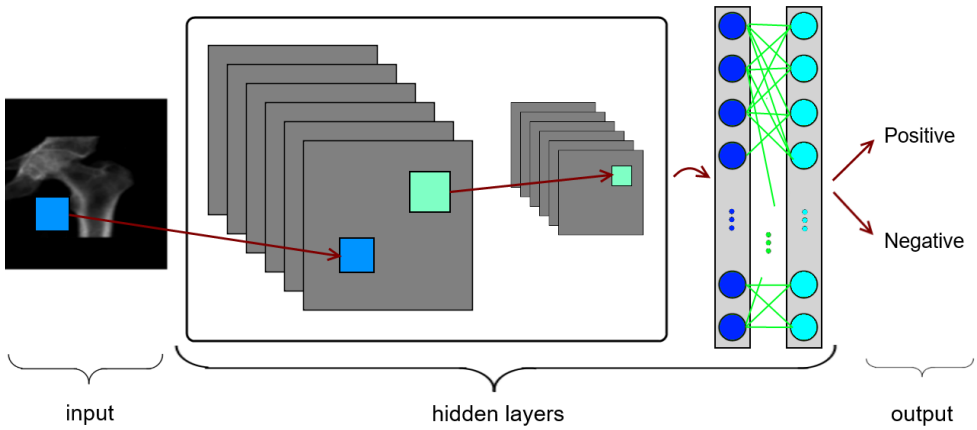


Figure 2.1: Flowchart of a CNN. An image is fed to the network. The hidden layers perform computations. Before the classification of the given image is done, a layer connects all the outputs from the hidden layers in order to compute the probability of the classes.

The most characteristic layer in a CNN is the convolutional layer. This layer performs convolution between a filter and the preceding layer. The filter slides through the input in given step sizes and multiplies the pixel values of the input with the filter's receptive field. The filters are usually small. Most often, the filters are created at random, and the initial receptive field of the filter is thus randomised. In contrary to traditional image analysis, where filters are hard coded to detect features as lines or corners, the convolutional filters detect features determined by the training [9]. An example of the procedure of the convolutional layer can be found in Figure 2.2. The output of this layer is trained filters that detect certain features of the layer's input.

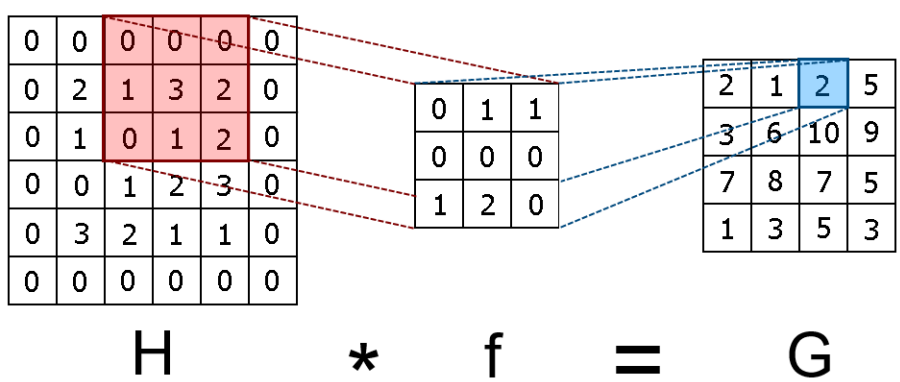


Figure 2.2: Example of the procedure of the convolutional layer. Part (the red box) of input H to the left is convoluted with the filter f in the middle and results in one number (the blue box) in the output G. The input has zero padding in order to preserve the input size for the output.

In order to not lose information along the image border, zero padding can be added during implementation. This technique will ensure that the size of the layer's input is preserved [14]. Zero padding means that there will be a pixel border of zeros added to the input, see Figure 2.2. This will prevent information from the layer's input to be lost while not adding any false information to it in the mean time.

An essential term in deep learning is the activation function. The function gives a layer the opportunity to perform non-linear computations. Linear neurons are limited in the sense of not being able to perform non-linear computations, and it can be considered as having a network with no hidden layers [9]. A common activation function is rectified linear unit (ReLU). This function outputs 0 if the input is negative, and outputs the input value if it is greater than or equal to 0. The ReLU-function is the blue dashed graph viewed in Figure 2.3. The activation function can e.g. be applied to the convolutional layer.

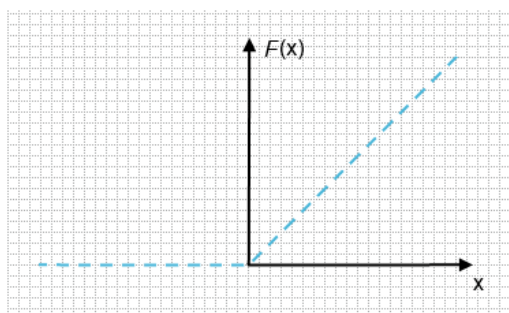


Figure 2.3: The ReLU-function.

Another useful layer in a CNN is the pooling layer. This layer often downsizes the input and outputs a smaller image. The pixel values of the output are either generated by choosing the maximum value of the pixels to pool or by computing the average of the pixels to pool [9]. When implemented, the pooling layer has a variable called "stride" that needs to be defined. This variable defines how many pixels away the filter should be placed again, with or without overlap. If a layer has size 2×2 and stride 2, then the output from that layer would be a similar image, with a pixel size equal a quarter of the original [8]. An example of an image that has been put through a max-pooling layer of size 2×2 with stride 2 can be viewed in Figure 2.4.

A common practice is to add a fully connected layer (densely connected in some literature) to the network before classification, see e.g. the rightmost part of the hidden layers in Figure 2.1 for an illustration. In a fully connected layer, all the neurons in the current layer are connected to all the neurons in

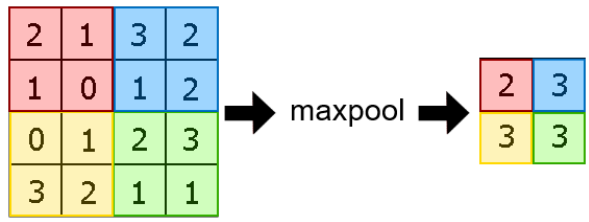


Figure 2.4: Example of the pooling layer. To the left is the input and to the right the output. In this example, max-pooling has been performed. The size of the filter is 2×2 and the stride is 2. The output pixel size is a quarter of the input pixel size.

the subsequent layer. The advantage of placing such a layer before classification is that all the findings from the network are summarised to help the classification [8].

Before classification of an image, all the filters generated by the convolutional layers along with the image itself are flattened. The classification is then handled by a classification layer that computes the probability of an image belonging to a certain class. Among the classification layers, the most common ones are the sigmoid layer and the softmax layer. The sigmoid function computes the probability according to Equation 2.1; here $f(x)$ denotes the probability and x the output from the precedent layer.

$$f(x) = \frac{1}{1 + e^{-x}} \tag{2.1}$$

The softmax function computes the probability according to Equation 2.2; here $S(y_i)$ denotes the probability for class i, where y_i denotes the output from the precedent layer. For a binary classification problem, where there are only two classes to decide between, a sigmoid layer is used. In a multiclass classification problem, a softmax layer is used [9].

$$S(y_i) = \frac{e^{y_i}}{\sum_j e^{y_j}} \tag{2.2}$$

The performance of a classifier is evaluated according to the accuracy of the training data. In order to get good results, there is a chance of the classifier fitting most of its parameters to features that are only present in the training data. This will then lead the classifier to overfit to the training data and will therefore perform worse on new data since the learnt features are not generalised enough. Overfitting to the training data is a common issue in deep learning [9]. There are several methods to prevent this; one method is

to add a dropout layer. The approach is to deactivate some neurons at a random rate each time a new input layer is presented to the network. In this fashion, other neurons will need to learn more general features to compensate for the neurons that are deactivated.

Batch normalisation is a method to accelerate the training process. The method normalises the image inputs to every layer in the neural network. This is done by limiting how much the weights in a hidden layer can shift. If the shifts in the bottom layers are too large, the neurons in the top layers not only have to learn how to do good predictions, but also compensate for the weight shifts from preceding layers. This will slow down the training process and the weights of the network will not converge as fast [9].

An error function is implemented to measure how well the network performs in terms of making correct predictions. The error function, also called cost function or loss function, can be expressed as the difference between the prediction and the ground truth. The aim is to find the parameters, weights and structure that minimise the error function. When dealing with a binary classification problem it is common to use a binary cross entropy function.

Another method to combat overfitting is to add regularisation to the ANN. Regularisation adds additional terms to the cost/loss functions and penalises large weights. The effect of the regularisation term is determined by a variable called "scale". If the scale equals zero, then the regularisation does not affect the outcome of the error function. The most common regularisation terms are L2-regularisation and L1-regularisation.

2.1.2 The math behind an ANN

Behind every layer of an ANN there are many computations. As already mentioned, to assure the network is learning something, the error between the network's output and the actual key is calculated. Through the chain rule, the error derivatives can be backpropagated from the last layer to the first one. This will later give the information about how changes in the connections between the input and the output will affect the error [15].

Different optimisers can be used for the task of finding a global minimum. The computations of the new values often involve gradient descent of the current value. One of many optimisers that are already implemented in the TensorFlow (TF) and Keras libraries is the Adam Optimiser. In short, the optimiser implements stochastic gradient descent of the error [16]. But in contrary to classical stochastic gradient descent, which maintains the same learning rate throughout the training, Adam calculates individual adaptive

learning rates during training. Different parameters have different learning rates, and the input value for the implemented optimiser is the maximum value of the learning rate.

2.1.3 Residual Networks

Neural networks that are very deep, i.e. have many hidden layers, are harder to train due to the risk of vanishing gradients. When the error to the ground truth is calculated, the error is based on outputs from earlier layers. If those outputs are close to zero, the gradient will vanish and equal zero as well. This is known as the vanishing gradient problem [9]. Even though this issue has been known for long, training deeper networks is still of interest. The depth of the network is of crucial importance for the success of a CNN, since deeper networks will be able to learn and solve more complex problems. Increasing the depth of the network will also lead to another well-known problem: degradation – the accuracy gets saturated without any improvement, no matter for how long the network is trained [17].

The challenge of training a deep network is what He et al. [17] were aiming to combat by adding identity mappings of the original input. These identity mappings of the original image have skip-layer connections. Skip-layer connections take shortcuts and the input skips some of the hidden layers of the network, see Figure 2.5. This technique is mostly used to ease the implementation of deeper networks [18].

He et al. state that the degradation problem can be battled by approximating a residual function, $\mathcal{F}(x) := \mathcal{H}(x) - x$, to an underlying mapping $\mathcal{H}(x)$ of the network. Here x denotes the input, and $\mathcal{H}(x)$ a mapping that is going to be fitted by a few stacked layers. The authors assume that if someone thinks that multiple non-linear layers can asymptotically approximate a complicated function, then this person, in other words, thinks that these layers can asymptotically approximate the residual function of the given function. Approximation of $\mathcal{F}(x)$ is easier to implement and will therefore ease the learning process of the network. By experiments the authors confirmed that identity mappings solved the degradation problem [17]. In Figure 2.5, the skip-layer connection is adding the identity map x to $\mathcal{F}(x)$ (that is the output from the convolutional layers). The type of network developed by He et al. is called a residual network (ResNet) [17].

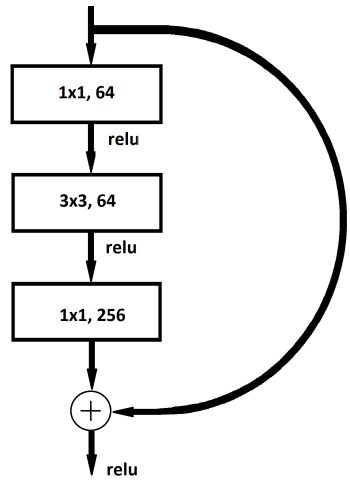


Figure 2.5: Illustration of a skip-layer connection. The first number in each box denotes the size of each convolutional filter, and the second number is the number of filters in each layer.

2.1.4 Training a classifier

The data sets used for training a classifier are generally divided into the following three sets: training, validation and test. The training set usually contains most of the data and is used to train the classifier. The validation set is used after every round of training to validate what the classifier has learnt. Validating the training will help prevent the classifier from overfitting its parameters to the training data. When all the images in the training set has been presented to the network, it is said that the network has finished one epoch of training. A training session consists of several epochs. After the training session the network is fed the test set for prediction. This is done to see how well the classifier performs on data it has not been exposed to earlier. The data that is not used for training is usually split evenly between the validation set and the test set [9].

Implementation of cross-validation is a way to measure the validation error. Here, the data set is only divided into a training set and a test set. The training data is then divided into K partitions, and one of these partitions is chosen for validation, whilst the rest are used for training. After several epochs, a new partition is chosen for validation. This procedure is done K times, until all the partitions have been used for validation. Using the technique of K partitions is often referred to as K-fold cross-validation.

2.2 Transfer learning

Training a neural network from scratch requires a lot of data and takes a lot of time. This is where transfer learning can be useful. The idea behind transfer learning is to use information gained from a neural network that has been trained on a similar image classification problem. Instead of starting from scratch the patterns from this network can be used as a starting point. Then the performance can be improved and progress can be achieved more quickly when performing on a new task, when compared to training from scratch [19; 20].

Transfer learning is mostly used when a huge amount of computational power is needed, such as in computer vision and natural language processing tasks [19]. It is also commonly used in deep learning due to the huge amount of resources or the large data sets needed for training those models [20]. When using transfer learning it is important to remember that in order for it to work in deep learning, the model's learnt features from the first task need to be general [19].

When facing a prediction task with image input data, it is common to use a deep learning model that is pre-trained for a large and challenging image classification task [20]. One of these is ImageNet which is a project that aims to provide a large image database. ImageNet contains more than 14 million colour RGB-images which belong to more than 20 000 classes. These classes have a big variety, some examples of them are dogs, chairs, persons and stones [21].

2.3 Visualising data and results

To visualise high dimensional data, one can use different multidimensional scaling techniques. These techniques will help categorise data and tell if there are initial differences within the data that will be used in different experiments.

2.3.1 Student's t-distributed stochastic neighbour embedding

Stochastic neighbour embedding (SNE) starts by converting the Euclidean distances between data points into conditional probabilities. Let x denote the high-dimensional data points, while y denotes the low-dimensional data points. If the conditional probabilities between x and y are equal, then there

is no need to compute the conditional probability for x. SNE aims to minimise the eventual mismatches between the different conditional probabilities between x and y. The conditional probabilities denote similarities between two points. More similar points are placed closer to each other while less similar are placed further apart. In this fashion, it is easier to distinguish – both visually and mathematically – between data belonging to different classes [22].

In comparison to SNE, Student's t-distribution stochastic neighbour embedding (t-SNE) uses the Student's t-distribution instead of the Gaussian distribution. Another cost function that is easier to optimise is also implemented. And due to the nature of the t-distribution of having heavier tails, the chance of high-dimensional data points ending up in a cluster on a plane can also be avoided [22].

2.3.2 Principal components analysis

Principal components analysis (PCA) aims to minimise the data loss when reducing the dimensionality of the data. This is done by deriving a new set of uncorrelated variables, called principal components, from the original set of variables. All the new variables are placed in a new orthogonal coordinate system. All the axes in the new system are pointing towards the direction of maximum variance, in decreasing order of their contribution to the total variance. The reduction of dimensionality is then done by recreating the original set of variables from the first principal components, since most of the information lies in the first few components [23]. After dimension reduction, the data can be plotted in either a two dimensions plot or a three dimensions plot to more clearly demonstrate the similarities of the data. More similar data are clustered together.

2.3.3 Interpreting results from an ANN

The results of an ANN can be viewed through different curves: the loss curve, the accuracy curve and the receiver operating characteristic (ROC) curve. The loss curve tells if the network is improving during training and validation. The smaller the loss is, the better is the network learning. A sign of whether the network is overfitting or not is an increasing loss for validation but decreasing loss for training. The accuracy curve demonstrates how well the network performs. High accuracy is aimed for on the training, validation and test set.

When a data set for binary classification is imbalanced, the accuracy is not as helpful for describing the performance of the ANN. This is because high accuracy can be achieved by the network by disregarding one of the classes and only classify the images in the class that most of the data represent. For such a data set, it is rather recommended to view the ROC-curve. The curve shows the rate of true positives against the rate of false positives. As the names state, the rates tell how many of the positives are classified correctly (true positives) and how many negatives that have been classified as positives (false positives). The data points of the curve are derived from the probability of an image being correctly classified. To fully interpret the ROC-curve, one can calculate the area under curve (AUC). The greater value of the AUC, the better is the classifier. A classifier that makes random guesses will achieve an AUC equal to a half; the maximum value of the AUC is one.

3. Material and methods

In this chapter, the procedure of the thesis project is described. First, exploration and a comparison of the two different cohorts were made. Then, implementation of an own neural network and transfer learning to an existing neural network were carried out.

3.1 Computer resources

The libraries used in this project are TF [24] and Keras [25]. The simulations were run on the graphical processing unit (GPU); the GPU used was a NVIDIA GeForce GTX 1060 AERO ITX with 6 GB memory. Running simulations on the GPU, in comparison to the central processing unit, provided faster results. This is because the computations, that mostly involve matrix multiplication and convolution, can be done faster on the GPU.

3.2 Material

The DXA-images used in this project are from the Swedish part of MrOS (Mister Osteoporosis, a longitudinal cohort study in elderly men), a study of osteoporotic fractures in elderly men [26]. All the images are therefore images that have been taken for the study, and no new images were taken especially for this thesis project. Data augmentation of existing images has been performed.

MrOS is a study in which men from the US, Hong Kong and Sweden participated. The main objective was to study elderly men in different part of the world and their risk of suffering an osteoporotic fracture. For the Swedish part, three cities were involved – Gothenburg, Uppsala and Malmö [27; 28]. Due to different brands of the screening machines that were used in different

hospitals, only images from the Uppsala and Malmö cohorts have been used in this project.

The study from Uppsala followed 1003 men from 2001 to 2010. Of these men, 989 have taken images of the hip region. The researchers in Malmö studied 1005 men from 2001 to 2009, with 997 patients that have taken images of the hip. The mean age of the participants in these two cities was 73.5 years at the start of the study. The initial DXA-screenings were performed in 2001 and most of the men were followed up for a second screening at a mean of 2.5 years from the first screening. The third screening was performed at a mean of 1.5 years from the second screening. In Uppsala, 858 participants returned for the second screening, and 637 for the third. In Malmö, 831 patients returned for the second screening, and 635 for the third.

Fractures that have occurred from the beginning of the study until the last of December 2013 have all been recorded. There is also a patient database in which information of the patients and their connections to the images are documented. A total of 111 fractures were recorded: 60 in the Uppsala cohort and 51 in the Malmö cohort.

3.3 Processing the patient database

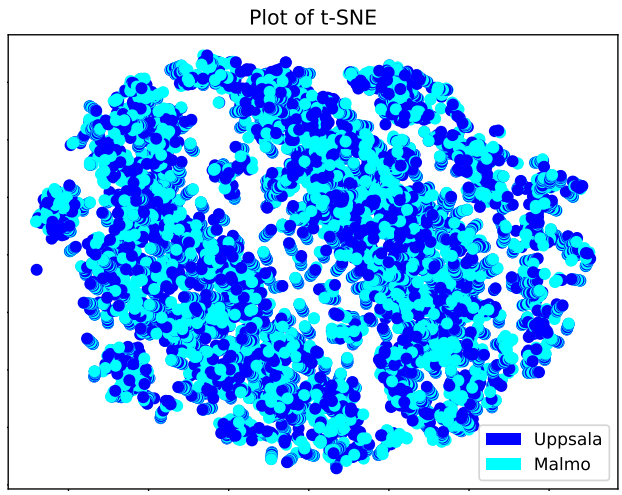
The patient database was processed in order to connect the different images to the different patients. The outcome was used to create the keys used for the ANNs. Images from both left and right femur were either both treated as negatives (no fracture recorded for that patient) or positives (fracture recorded for that patient), since no information of whether the fractures occurred on the right or left femur was found.

There were some cases in which a patient suffered a fracture not long after the beginning of the study, and there were also cases where the patients did not have any follow-ups and did not suffer a fracture until the very end of the study. This motivated a threshold for what should be considered a positive and what should be considered a negative. A threshold of five years was chosen. If a patient had taken an image more than five years prior to the fracture, the images of that patient would be disregarded. The number of five years was calculated by subtracting the acquisition time of the image from the time the fracture happened. If this number was greater than the number of days in five years, the image was considered as a negative.

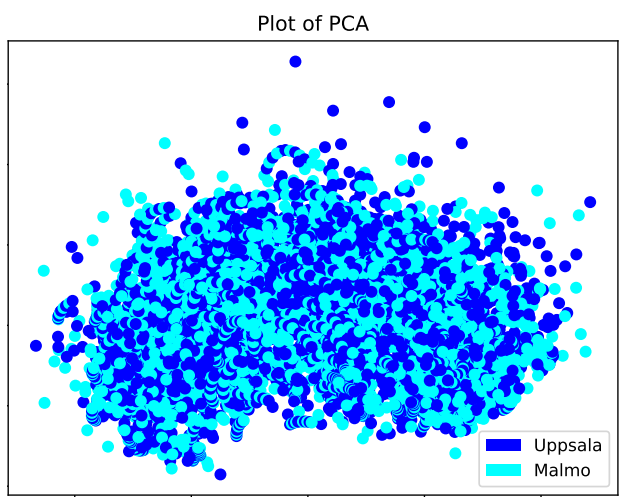
The threshold of five years was chosen since the authors thought it might be harder to predict something that is in a distant future. Choosing five years did not result in losing too many images due to the time threshold. Also, if

a patient has kept on participating in the study after fracture, all the images related to the patient that were taken after the fracture were disregarded. Regarding all the decisions made, it resulted in 178 images being classified as fractures: 96 images from the Uppsala cohort and 82 from the Malmö cohort.

3.4 Comparing data sets using t-SNE and PCA



(a) Data generated using t-SNE-algorithm.



(b) Data generated using PCA-algorithm.

Figure 3.1: Plots of how the cohorts from Malmö and Uppsala differ. Both plots indicate that the cohorts are similar.

Since the two cohorts originate from two different cities and the images were taken with calibrated but different DXA-instrument, both t-SNE and PCA were performed on the data sets in order to learn if there were any initial differences amongst them. If there were no differences of greater magnitude, then one of the data sets could be used for training and validation, while the other data set could be used as a test set for prediction. The results of the comparison can be viewed in Figure 3.1a (result produced by the t-SNE algorithm) and Figure 3.1b (result produced by the PCA-algorithm). Since the data points are overlapping in the plots, similarity of the two data sets can be confirmed and the suggested data set split could be applied.

3.5 Pre-processing the images

The images from the cohorts contain the pixel intensities. The first step, as we also can see in Figure 3.2, was to translate those raw DXA-intensities to BMD-values. The data sets did not only consist of hip images but also full body scans and images of the spine. The next step was to identify the hip images in the cohorts which was done automatically based on the width of the images. The range used for the width was 230 to 290 pixels. In this step, images with strange artefacts were sorted out based on the pixel intensities. A pixel intensity greater than the value of 8.5 normally indicated a strange white region in the image. Images with the white regions on the pelvis or femur were removed manually. If the white region was to the left of the femur, the pixels were made black automatically and kept in the data set. Other images with artefacts in the form of black regions on the femur were removed manually. An example of an image used and an image removed can be found in Figure 1.1 and 3.3 respectively. The default input size of a ResNet-50 in Keras is 224×224 and the hip images were made this size by first adding two black boxes of equal size over and under the image (as can be viewed in Figure 3.2); then the images were cropped to the right width at the right side. The images were also normalised by dividing all pixel values by the maximum value in each image.

As we can see in Table 3.1 the data sets contain a low amount of positives. This amount was increased by performing some augmentation in form of rotations. All the positive images, and the same amount of negatives, were rotated in steps of one degree from $-5°$ to $5°$. This resulted in 1920 rotated images for the Uppsala cohort and 1640 for the Malmö cohort. After the rotations, every second non-rotated negative image was removed until the amount of removed images reached the same amount as the rotations, or until half of all the negative images were removed. This was done to increase

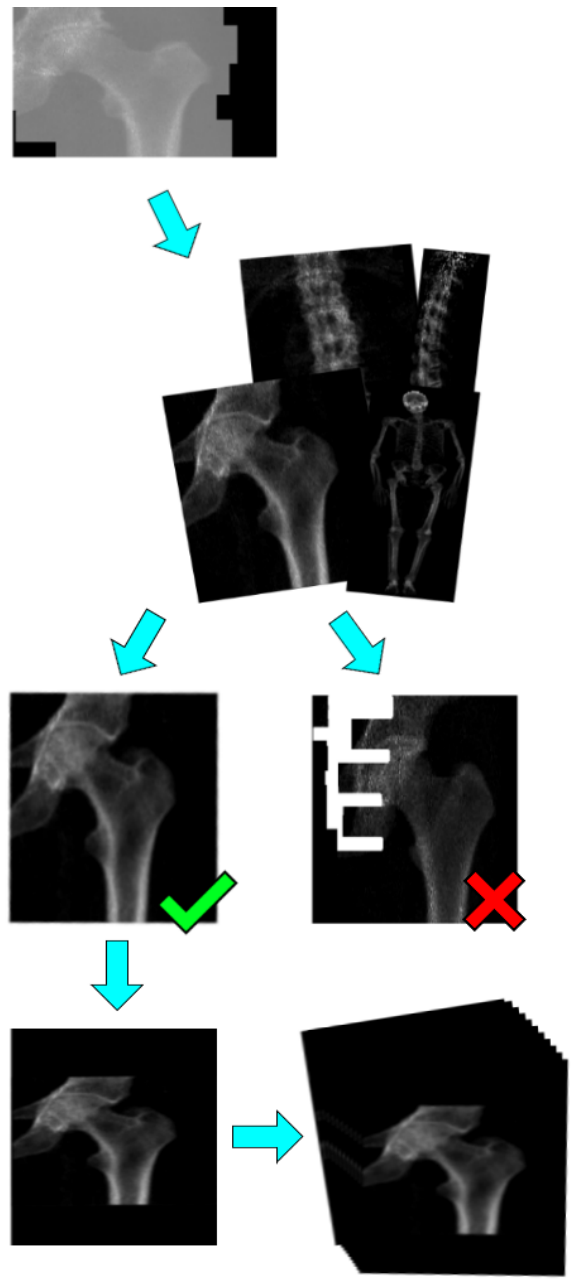


Figure 3.2: Flowchart of the image pre-processing. First, raw DXA-images were converted to BMD-images. Then the images of hips were sorted out. Also, the hip images with artefacts were discarded. All the images were then resized and normalised. Data augmentation in the form of rotating the images was then performed. Note that in the raw DXA-images and the resized images, the pixels are quadratic while they are rectangular in the other images.

the rate of positives further and 1920 images were removed from the Uppsala cohort and 1319 from the Malmö cohort. For patients that have participated in the follow-ups before suffering a fracture, only the image taken on the date closest to the fracture date was kept. The reason for removal of precedent images was to make the key more accurate. All images were converted into vectors by being stacked column wise. Then they were saved in a matrix, with one image per row. A key was also made corresponding to the matrix.

Figure 3.3: Example of an image removed due to artefacts.

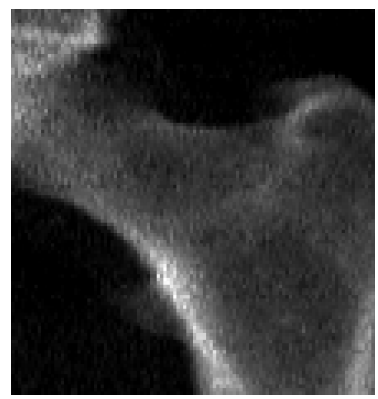


Figure 3.4: Example of an image from the ROI-data set.

To complement this an additional data set consisting of images of the femoral neck and trochanter, which is the region of interest (ROI), was created. These images were obtained by performing a centre crop around the femoral neck. Then the images were reshaped into the input size, 224× 224 pixels. An example of the ROI-images can be found in Figure 3.4.

Table 3.1: Table of some specific values for the cohorts before and after image augmentation.

Before augmentation	**Uppsala**	**Malmö**
Number of fractures	96	82
Total number of images	4665	2885
Percentage positives	2.06%	2.84%

After augmentation	**Uppsala**	**Malmö**
Number of fractures	1056	902
Total number of images	4665	3206
Percentage positives	22.64%	28.13%

3.6 Own developed model

There have been two major network architectures developed from scratch. What the networks have in common are the convolutional baseline, a few maxpool layers and a sigmoid classification layer with two neurons due to the problem being binary. Calculations of the probabilities, according to the sigmoid function presented in Equation 2.1, were done with the logit values as input from the network, generated by the last dense layer. The activation function for all the convolutional layers is ReLU. No activation function has been added to the dense layer at the end of the network. What differ the two major network architectures the most are the amount of layers in each network and the amount of filters in each convolutional layer. As a start, only dropout was implemented in the ANNs to prevent overfitting.

In early stages, deeper networks were of major interest with the sole purpose of reaching greater accuracy and performing better on the test set. One of the networks of that structure is presented in Table 3.2. There have been some numerical instabilities for those networks that were later solved. Because of the depth of the network, the simulations took longer time to run. In the later stages, the focus shifted from reviewing the accuracy to reviewing the ROC-curve and interpreting the AUC. Smaller CNNs were then developed to investigate the different regularisation methods and how they could prevent overfitting. If a smaller network with addition of regularisation and K-fold cross validation would be able to produce the same results as a deeper network, then there would not be any need of a deeper network that needs much more computational power and has the risk of overfitting. One of the smaller CNNs implemented has the network architecture presented in Table 3.3. The networks will from here on be referred to as the "deeper network" (the architecture in Table 3.2) and the "shallow network" (the architecture in Table 3.3).

All the CNNs have been developed in TF. No explicit calculations have been done manually, the layers that have been used in the networks are pre-defined in the library. A class called `Estimator` has been used to handle training, validation and prediction for the network.

Each simulation had random weight initialisation, which make the results hard to reproduce. All the parameters were kept the same for several runs in order to assure whether the adjustment has had effect.

The training of the own developed model used smaller subsets – minibatches – of the images. This is since the number of images affected the number of parameters to be computed. The size of a minibatch was limited by the memory of the GPU (6 GB).

Table 3.2: Table over the architecture of the deeper CNN. "Kernel size" is the size of the filter; "Filters" denotes the amount of filters; "Stride" how much the image is downsized; "Activation" the activation function; "Padding" if there has been any zero-padding added to the image and "Regularizer" if any regularisation has been added to a layer. The pooling technique is maxpooling.

Layer	Kernel size	Filters	Stride	Activation	Padding	Regularizer
`input`	-	-	-	-	-	-
`conv1`	5	16	-	ReLU	yes	-
`pool1`	2	-	2	-	-	-
`conv2`	5	32	-	ReLU	yes	-
`conv3`	5	64	-	ReLU	yes	-
`pool2`	2	-	2	-	-	-
`conv4`	3	64	-	ReLU	yes	-
`conv5`	3	128	-	ReLU	yes	-
`pool3`	2	-	2	-	-	-
`dense`	-	-	-	-	-	-
`dropout`	-	-	-	-	-	-
`logits/dense`	-	-	-	-	-	-
output	-	-	-	-	-	-

Table 3.3: Table over the architecture of the shallow CNN. "Kernel size" is the size of the filter; "Filters" denotes the amount of filters; "Stride" how much the image is downsized; "Activation" the activation function; "Padding" if there has been any zero-padding added to the image and "Regularizer" if any regularisation has been added to a layer. The pooling technique is maxpooling.

Layer	Kernel size	Filters	Stride	Activation	Padding	Regularizer
input	-	-	-	-	-	-
`conv1`	5	8	-	ReLU	yes	L2
`pool1`	2	-	2	-	-	-
`conv2`	5	8	-	ReLU	yes	L2
`pool2`	2	-	2	-	-	-
`conv3`	5	8	-	ReLU	yes	L2
`dense`	-	-	-	-	-	-
`dropout`	-	-	-	-	-	-
`logits/dense`	-	-	-	-	-	-
output	-	-	-	-	-	-

Different hyperparameters have been tuned in order to produce better results. The hyperparameters that could be tuned were:

- learning rate,
- batch size,
- dropout rate,
- `pos_weight`, parameter to adjust the network to favour either true or false positives,
- amount of steps taken in each training epoch,
- the number of epochs in training,
- the number of epochs in validation,
- the number of training sessions and
- the scale parameter for the regularisation

For the deeper network, the whole Uppsala cohort has been used for training. Malmö cohort has been split to be used for validation and test. Of the data set, 60% was used for validation and 40% for test. For the shallow network, K-fold cross validation was implemented. For that reason, the whole Malmö data set could be used for testing of the shallow network. The Uppsala cohort was used for training in this case as well. The data sets used for training and validation contained augmentated data while the data set used for prediction did not have any augmentated data.

3.7 Transfer learning

For transfer learning a ResNet-50 in Keras was used as a base model. A ResNet-50 model consists of 50 layers. The first layer is input layer which is followed by 16 residual blocks of three layers each. These blocks are of the same structure as the block in Figure 2.3, but with different convolutional filter sizes. The last layer is the output layer. The ResNet-model was trained on ImageNet and classified images into 1 000 different classes. ResNet has a pre-defined input size of $224 \times 224 \times 3$ since the network is trained on colour images and colour images are represented in the 3-channel RGB model. Greyscaled images on the other hand are only represented with 1-channel. The greyscaled images in the database therefore needed to be adapted to the 3-channels system. This was done by using identical copies of the images for the second and the third channels.

In order to adapt the ResNet for binary prediction, the last layer was replaced by a dense layer that gives one output instead of 1 000. This architecture, with the adapted ResNet-model, was trained with different learning rates. The training consisted of both transfer learning and fine-tuning. The transfer learning was performed by freezing all weights in the ResNet-architecture and training the adapted network for ten epochs with an Adam optimiser and a binary cross entropy cost function. Afterwards the network was fine-tuned by making two of the last layers in the ResNet-architecture trainable. Then the network was trained with the same optimiser and cost function for an additional 30 epochs.

Later a dropout layer was added after the dense layer in order to prevent overfitting. An overview of this architecture can be found in Table 3.4. This adapted network was trained with different learning rates. In an attempt to improve the performance of the model further, two blocks consisting of a convolutional layer followed by a batch normalisation layer were added. This architecture was also trained with different learning rates. Due to overfitting, L2-regularisation was added to the dense and convolutional layers. This architecture, see Table 3.5, was also trained with different learning rates. Architectures with the same structure as in Table 3.4 and 3.5 were trained with and without L2-regularisation on the dense and convolutional layers in different combinations, e.g. with regularisation only on the dense layer.

The whole Uppsala cohort after data augmentation has been used for training. The Malmö cohort, without data augmentation, has been used to test the network. Different hyperparameters have been tuned in order to produce better results. The hyperparameters that could be tuned were:

- learning rate, both for transfer learning and fine-tuning,
- batch size,
- step size,
- dropout rate,
- the number of epochs in transfer learning,
- the number of epochs in fine-tuning,
- the scale parameter for the regularisation

Table 3.4: Table of the base architecture used for transfer learning. "Kernel size" is the size of the filter; "Filters" denotes the amount of filters; "Activation" the activation function; "Padding" if there has been any zero-padding added to the image and "Regularizer" if any regularisation has been added to a layer.

Layer	Kernel size	Filters	Activation	Padding	Regularizer
input	-	-	-	-	-
`Resnet50`	-	-	-	-	-
`flatten`	-	-	-	-	-
`dense`	-	-	-	-	-
`dropout`	-	-	-	-	-

Table 3.5: Table of the architecture used for transfer learning. "Kernel size" is the size of the filter; "Filters" denotes the amount of filters; "Activation" the activation function; "Padding" if there has been any zero-padding added to the image and "Regularizer" if any regularisation has been added to a layer.

Layer	Kernel size	Filters	Activation	Padding	Regularizer
input	-	-	-	-	-
`Resnet50`	-	-	-	-	-
`conv1`	3	5	sigmoid	same	L2
`batch normalization`	-	-	-	-	-
`conv2`	3	5	sigmoid	same	L2
`batch normalization`	-	-	-		-
`flatten`	-	-	-	-	-
`dense`	-	-	-	-	L2
`dropout`	-	-	-	-	-

4. Results

In this chapter, the results of the thesis project are presented. The following different comparisons are made to evaluate the outcome of this thesis project:

- How well the different networks of the same approach perform
- How well customised networks perform in comparison to fine-tuned networks
- The results depending on whether the full size image data set or smaller ROI data set was fed to the networks

To facilitate comparisons, the results from the different models will be presented in similar figures. To each network there are one ROC-curve and two loss curves – one for the training loss and one for the validation loss. The AUC of each ROC-curve is also documented in the figure caption. All networks are tested on images from the Malmö cohort without any augmentation.

4.1 Results from own developed model

Results from the deeper network presented in Table 3.2 can be viewed in Figure 4.1. The full size images were fed the network to generate predictors in Figure 4.1a and the ROI-images were fed the network to generate the predictors in Figure 4.1b. The corresponding loss curves to the ROC-curves are presented in Figure 4.1c and 4.1d, respectively. The values of the parameters used by the network to generate the results can be found in Table 4.1.

From Figure 4.1a and 4.1b we can read that the AUC is greater when the network is fed the ROI-images, the graphs have an AUC of 0.6102 and 0.6362 respectively. The loss curves in Figure 4.1c and 4.1d indicate that the network is learning since the losses for training are decreasing. However, the validation losses are increasing. The accuracy of the network is 0.9581 for the full size images and 0.9529 for the ROI-images.

Results from the shallow network presented in Table 3.3 can be viewed in Figure 4.2. The full size images were fed the network to generate predictors in Figure 4.2a and the ROI-images were fed the network to generate the predictor in Figure 4.2b. The corresponding loss curves to the ROC-curves are presented in Figure 4.2c and 4.2d, respectively. The values of the parameters used by the network to generate the results can be found in Table 4.2.

From Figure 4.2a and 4.2b we can read that the AUC is greater when the network is fed the full size images, the graphs have an AUC of 0.7821 and 0.6664 respectively. The loss curves in Figure 4.1c and 4.1d indicate that the network is learning since the losses for training are decreasing. The validation loss in Figure 4.2c has a value close to zero, which means that the network has solved the problem presented for it. The validation loss in Figure 4.2d still has not stabilised at the end of the simulation, but there is a slight trend of it increasing. The accuracy of the network is 0.9700 for the full size images and 0.9390 for the ROI-images.

Table 4.1: Table over the parameters used to generate the results for the deeper network model. The parameter `pos_weight` is the parameter that could be tuned to adjust the network to favour either true or false positives.

Parameter	Full size images	ROI-images
`learning rate`	0.0004	0.0007
`batch size`	106	106
`dropout rate`	0.6	0.3
`pos_weight`	0.8	0.8
`steps, training`	30	20
`epochs, training`	None	None
`epochs, validation`	1	1
training sessions	40	50

Table 4.2: Table over the parameters used to generate the results for the shallow network model. The parameter `pos_weight` is the parameter that could be tuned to adjust the network to favour either true or false positives.

Parameter	Full size images	ROI-images
`learning rate`	0.002	0.0002
`batch size`	110	110
`dropout rate`	0.2	0.4
`pos_weight`	0.8	0.8
`steps, training`	30	30
`epochs, training`	None	None
`epochs, validation`	1	1
`L2-scale`	0.02	0.02
training sessions	20	10
K-folds	4	4

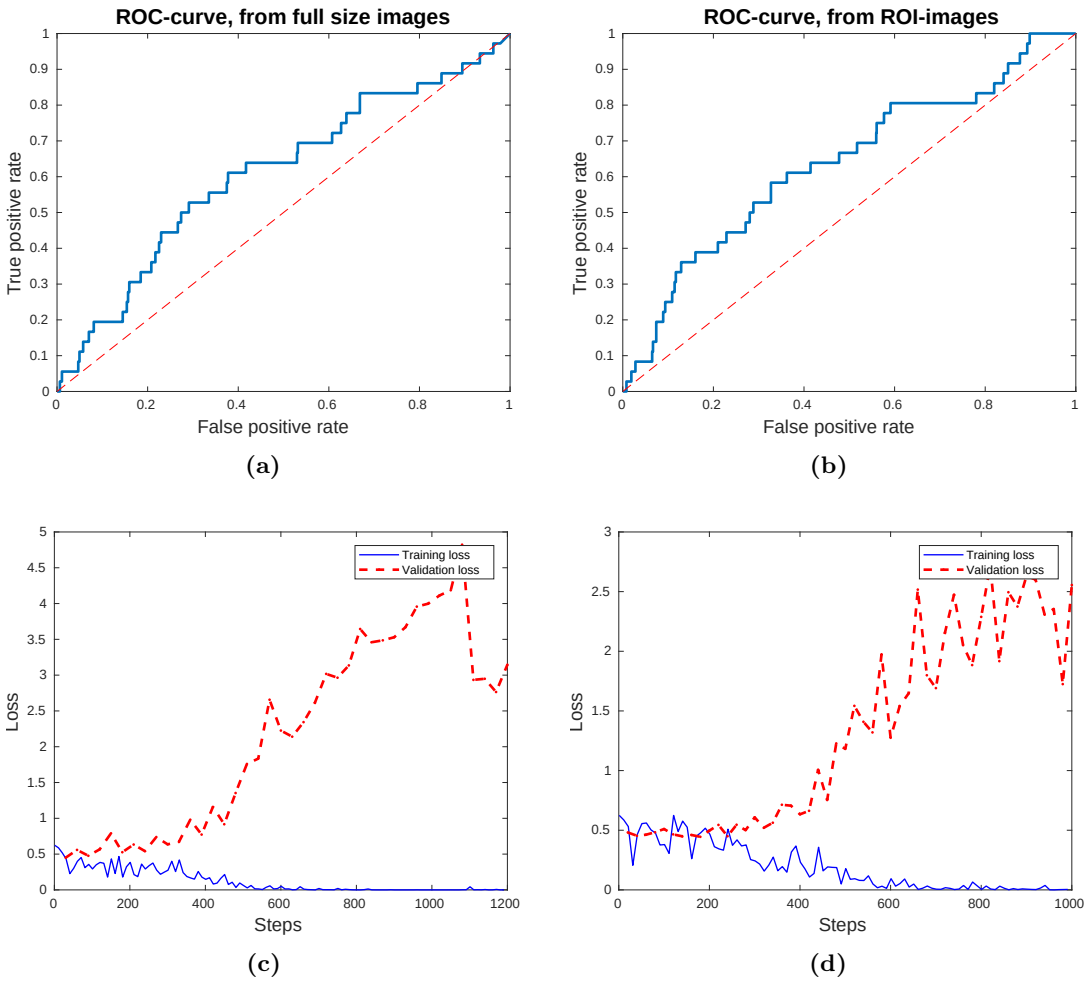


Figure 4.1: The deeper custom network when (a) the full size images, or (b) the ROI-images, were used for training. For the full size images, the AUC is 0.6102 and the accuracy is 0.9581. For the ROI-images, the AUC is 0.6362 and the accuracy is 0.9529. The corresponding loss curves are found in (c) and (d) respectively. The network architecture can be viewed in Table 3.2. In (a) and (b), the blue graph is the ROC-curve. The red dashed graph represents the ROC-curve of a classifier that makes random guesses. In (c) and (d), the blue graph is the training loss and the red dashed graph the validation loss.

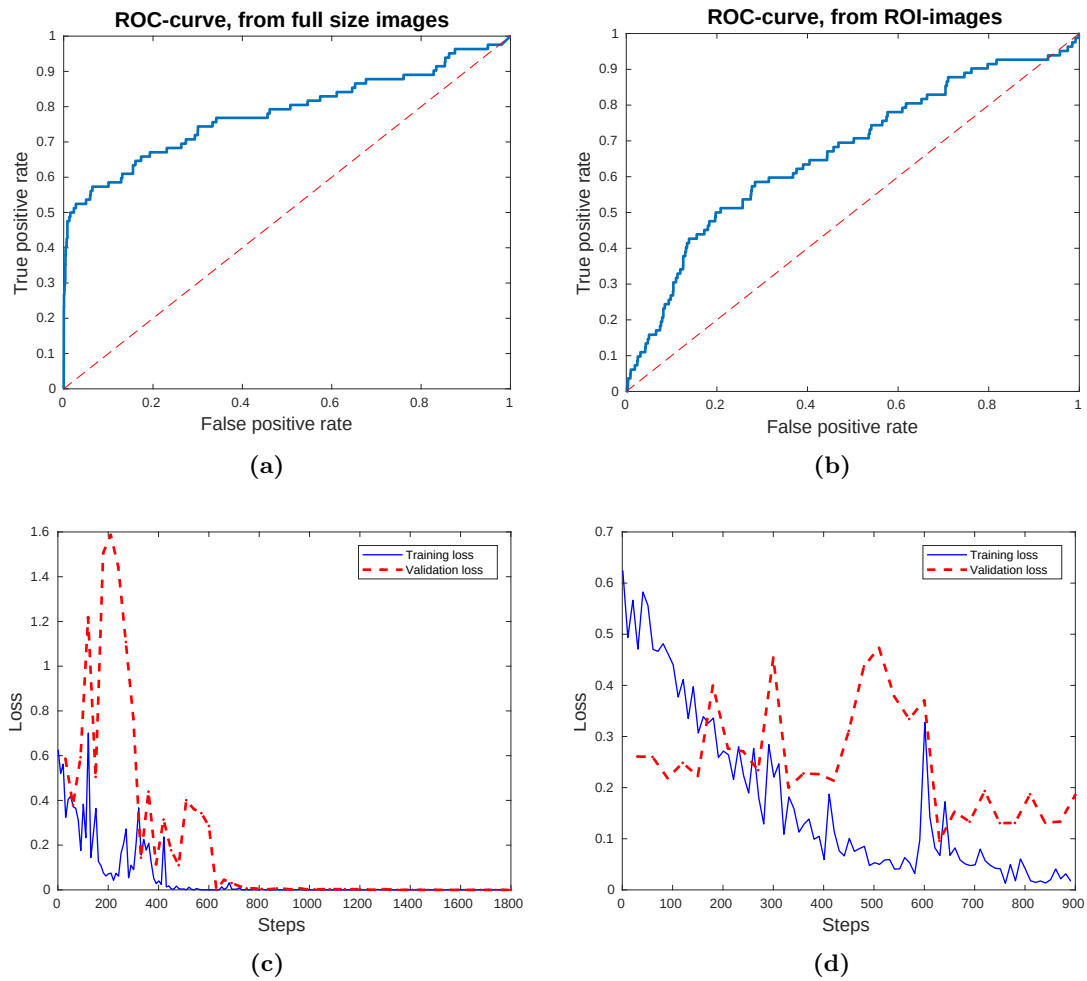


Figure 4.2: The results of the shallow custom network when (a) the full size images, or (b) the ROI-images, were used for training. For the full size images, the AUC is 0.7821 and the accuracy is 0.9700. For the ROI-images, the AUC is 0.6664 and the accuracy is 0.9390. The corresponding loss curves are found in (c) and (d) respectively. The network architecture can be viewed in Table 3.3. In (a) and (b), the blue graph is the ROC-curve. The red dashed graph represents the ROC-curve of a classifier that makes random guesses. In (c) and (d), the blue graph is the training loss and the red dashed graph the validation loss.

4.2 Results from pre-trained model

Results from one of the earlier architectures, presented in Table 4.3, can be found in Figure 4.3a and 4.3b. To generate the predictors in Figure 4.3a the full sized images were fed to the network and for the predictors in Figure 4.3b the ROI-images were fed to the network. The corresponding loss curves can be found in Figure 4.3c and 4.3d. The hyperparameters used can be found in Table 3.4.

From Figure 4.3a and 4.3b we can see that the AUC-value, 0.6193, is greater when the network is fed the full sized images compared to 0.5207 when it is fed the ROI-images. The validation loss in Figure 4.3c is increasing while the training loss is decreasing. The validation loss in Figure 4.3d is oscillating but still decreasing and the training loss is also decreasing. This indicates that the network is learning.

In the later part of the project the architecture in Table 3.5 was used. The results from this network can be found in Figure 4.4a, where the full sized images were used, and in Figure 4.4b, where the ROI-images were used. The corresponding loss curves can be found in Figure 4.4c and 4.4d. The hyperparameters used can be found in Table 4.4.

From Figure 4.4a and 4.4b we can see that the AUC-value, 0.6277, is greater when the network is fed the full sized images compared to 0.5262 when it is fed the ROI-images. We can see that the validation loss in Figure 4.4c is starting to oscillate which indicates some instability. The training loss is also oscillating but still decreasing. In Figure 4.4d both loss curves are decreasing, but the validation loss is slightly oscillating.

Table 4.3: Table over the parameters used to generate the results for the fine-tuned network model in Table 3.4.

Parameter	Full size images	ROI-images
learning rate, transfer learning	0.0001	0.001
learning rate, fine-tuning	0.0001	0.001
batch size	56	56

Table 4.4: Table over the parameters used to generate the results for the fine-tuned network model in Table 3.5.

Parameter	Full size images	ROI-images
learning rate, transfer learning	0.0001	0.001
learning rate, fine-tuning	0.0001	0.00001
batch size	56	56
dropout rate	0.3	0.3
conv1, regularisation scale	0.00005	0.00005
conv2, regularisation scale	0.00005	0.00005
dense, regularisation scale	0.0005	0.0005

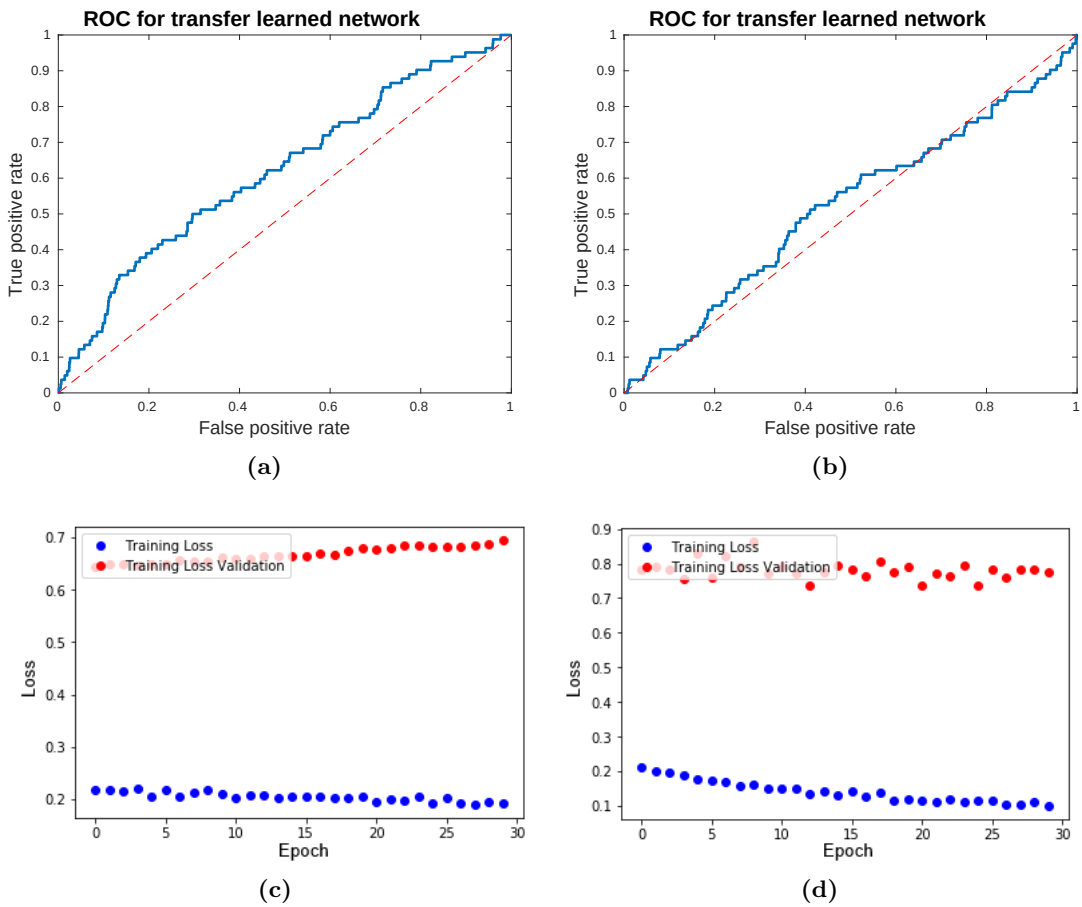


Figure 4.3: The results of the early pre-trained network when (a) the full size images, or (b) the ROI-images, were used for training. For the full size images, the AUC is 0.6193, and the accuracy is 0.9711. For the ROI-images, the AUC is 0.5207 and the accuracy is 0.9711. The network architecture can be viewed in Table 3.4. In (a) and (b), the blue graph is the ROC-curve. The red dashed graph represents the ROC-curve of a classifier that makes random guesses. In (c) and (d), the blue points are the training loss and the red points the validation loss.

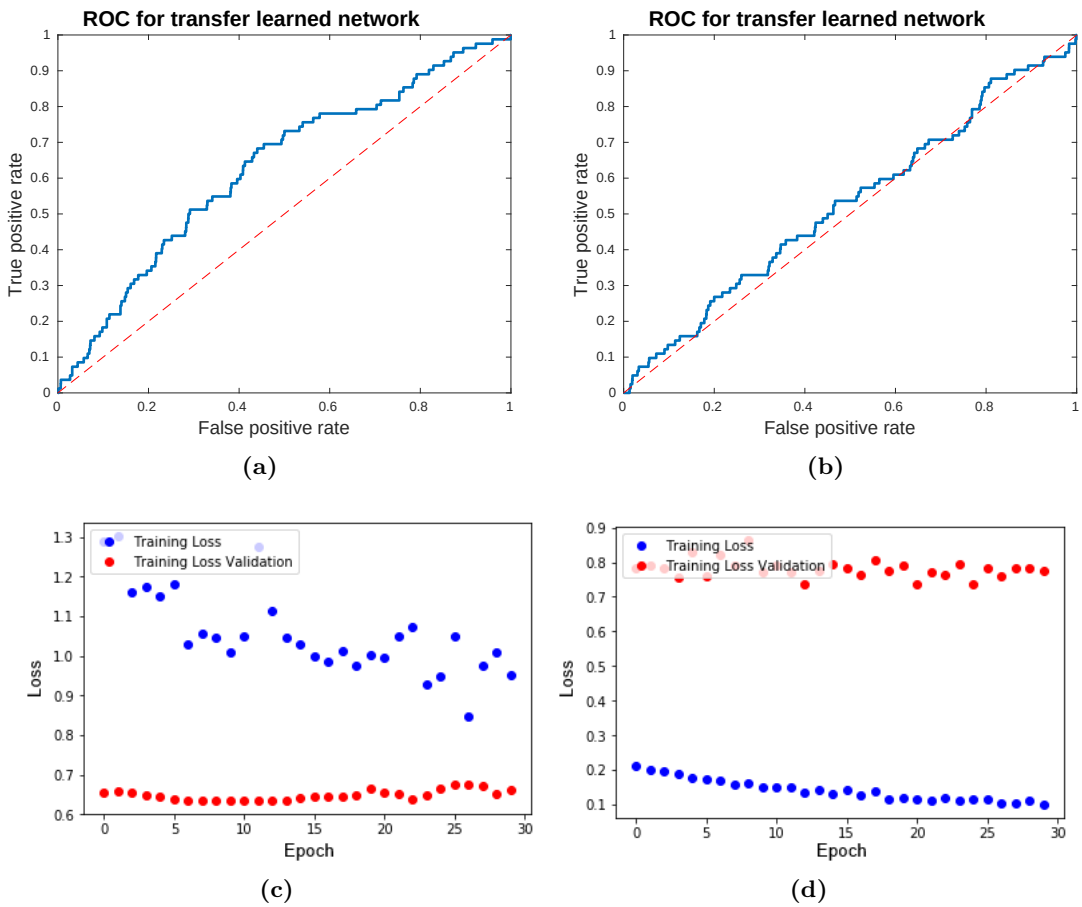


Figure 4.4: The results of the early pre-trained network when (a) the full size images, or (b) the ROI-images, were used for training. For the full size images, the AUC is 0.6277, and the accuracy is 0.9711. For the ROI-images, the AUC is 0.5262 and the accuracy is 0.9711. The network architecture can be viewed in Table 3.5. In (a) and (b), the blue graph is the ROC-curve. The red dashed graph represents the ROC-curve of a classifier that makes random guesses. In (c) and (d), the blue points are the training loss and the red points the validation loss.

5. Discussion

This thesis project has confirmed that it is possible to use an ANN to predict fracture risk in the proximal femur from DXA-images. The approach of using an ANN should be investigated further in order to obtain more promising results. Factors to consider in future work are other custom network architectures and other pre-trained networks that can be tuned that will be better suited for the presented problem. Also, from the reference list it can be read that most of the work using an ANN approach in orthopaedics are from the recent years. Therefore, the results from this thesis, although perhaps not as good as desired, should not be overlooked.

There is potential for deep learning in the field of biomechanics, but since the risk for fracture is dependent on so many other factors (e.g. previous falls, low body weight and cigarette smoking [12]) than osteoporosis and BMD themselves, it is hard even for a human to tell the risk for fracture within the five years to come. This makes the key that has been constructed for the neural network, not that reliable. If the problem is not easy for a human to solve, then this kind of neural networks with labelled data might not be preferable as a method. Also, patients may have traits of osteoporotic bones without having suffered from a fracture yet, which could have affected the reliability of the key.

5.1 Own developed models

As can been seen in the loss curves in Figure 4.1c and 4.1d, the deeper network that was developed in the early stages overfitted quite heavily to the data. One explanation could be that the amount of images was not enough. The amount of images used for training may not have been enough to cover the variations of the positives and negatives respectively. Since the network overfitted so heavily, only implementing regularisers or K-fold would not solve the problem. Therefore, the decision of developing a more shallow network was made.

The shallow network performed the best on the full size images. Even the predictors for the ROI-images gave a greater AUC than both the results from the deeper network. It has been shown that a more shallow network with the right parameters and methods that prevent overfitting has the most potential of producing better results for a project like this.

Prediction of fracture risk is based on the DXA-images from the femoral neck. Since the BMD-value in the femoral neck is used to determine the T-score, that specific region might also have more information about the risk of fracture. The results show that feeding the networks images of the ROI to train on gave different outcome depending on the network architecture. Comparing Figure 4.1a and 4.1b, we can see that the deeper network made more accurate classifications on the ROI-images. Even though the AUC is greater for the ROI-images, no conclusions can be made since the difference is so small. The opposite is seen in the comparison between Figure 4.2a and 4.2b: the shallow network performed better on the full size images. Here, the difference in AUC and accuracy of the two figures are great enough to conclude that training the shallow network on the full size images gave better results.

There has been a problem with the implementation of K-fold. The idea of K-fold is that the same data is used for both training and validation. The network would later be evaluated on the average of the joint results from the different training sessions of the K-fold. However, as can be viewed in the loss curves in Figure 4.2c from the shallow network presented in Table 3.3, the loss has reached the value of zero for both training and validation. This means, in other words, that the network has solved the problem presented to it. But if we instead look at the AUC of the test set in Figure 4.2a, we can see that it is not as promising as the loss curves indicate. The reason why there is a contradiction might be the use of the `Estimator` class in TF. This class might have remembered all the earlier training sessions for the next training session, which means that the validation data has already been exposed to the network and the network only needs to remember the validation data, without learning the features. Shuffle of the data has been enabled during training. This did not seem to help the network learn the specific features. An approach that might help is to not use the `Estimator` class to handle training, validation and prediction.

Splitting up the data into minibatches might have been the wrong approach since the amount of fractures was not that high. There could have been minibatches that did not contain any positives that the network was fed. During training with these minibatches the network might have adjusted its weight parameters to not take the positive class into consideration. Enabling

shuffle of the training data might have helped preventing this issue, but this has not been confirmed since the training has not been visualised and evaluated step by step.

5.2 Transfer learning

The ResNet-model did not perform well on this classification task. This can also be seen from the AUC-values in Figures 4.3a, 4.3b, 4.4a and 4.4b. The AUC-values for the network trained on the ROI-images are only slightly better than a classifier that makes random guesses, which indicates that the network does not perform well on this task. Despite adding regularisation and extra convolutional layers the AUC-values did not improve for this data set. The networks performed better when full sized images were used, AUC-values improved to 0.6193 and 0.6277, but they are still not as good as the custom network. In Figure 4.3c we can see that the training loss is decreasing while the validation loss is increasing. This is an indicator that the network is overfitting to the training data.

One possible explanation for the poor performance could be that the ResNet-model is pre-trained on images from ImageNet which could be too different from medical images. Another thing to consider is that three layers of the same greyscale image might not be able to replace the RGB-layers in a real colour image. It is also possible that the ResNet-model is too deep for this thesis' objective. Possible reasons for this are: too few images in the data sets and too many parameters that needed to be tuned.

Despite a lot of tuning of hyperparameters, the ResNet-model did not seem to learn, which probably can explain why the accuracy for all the transfer learned networks was the same. The hypothesis of the network not learning was based on the predictors obtained from the network and not the loss curves. To test this hypothesis a data set consisting of ten images, five positives and five negatives, was created. The network with the architecture as in Table 3.4 was trained on this data set with a high learning rate until it got accuracy equal to one and a loss close to zero. This means that the network correctly classified all the images in the training set, but the predictions showed that the network still could not separate positives and negatives. This contradiction was further investigated by creating a data set with nine images from ImageNet: five dogs and four cats. The network was then trained until the accuracy was equal to one on this data set. The predictions showed that all the images were classified correctly. The same results were obtained if the images were made greyscale. In order to see if another pre-trained network could solve this problem, the tests were also done with

a VGG-16 model instead of the ResNet-50 model. The greyscale images of the cats and dogs were not used in this case. The result of this can be seen in chapter B in appendix.

5.3 Comparison of the different approaches

From the results of this thesis, a custom network that is developed from scratch is more suited for this kind of project than a pre-trained ResNet-50. When a network that has been transfer learned is compared to a custom network, it can be seen that the custom network performs better both on the full size images and the ROI-images, almost independent of the network architecture.

The results of this thesis can be compared to today's diagnostic tools. A ROC-curve based on the BMD of the images in the data set that was fed to the network can be seen in Figure C.1. The AUC of the curve is 0.7497, which is greater than the AUCs presented in most of the results from the ANNs explored in this project. One possible explanation might be that the ANNs need more data in order to learn how to predict fracture risk.

5.4 Data sets

The number of positives, as can be seen in Table 3.1, was only a few percent in the cohorts before augmentation of data. This number might not be enough for training a good classifier since the ANNs might not consider the positive class as important. To come to a conclusion, both the data set with augmented data and the original data set were fed to an ANN. The results are viewed in Figure 4.2a (result from network trained on the augmented data) and Figure A.1 in appendix (result from network trained on the original data). Both results were given by the network presented in Table 3.3. As can be seen, the network performed better on predicting fracture risk of new data if it has been trained on the data set with augmented data. The same data set was fed the both networks for prediction.

The results of this comparison show that, as assumed at the start of the project, the amount of fractures in the original data set was not enough. The data augmentation was thus needed in order to achieve a well-performing classifier after training the neural network.

As viewed in the results, the predictions from the networks when trained on the smaller ROI-images were less accurate and the AUC in general smaller.

One reason for this phenomenon might be that the smaller images were enlarged in order to fit the image size for ResNet, 224×224 pixels. Rather than scaling up the pixels, perhaps the parts of the images could have been made black instead. The re-scaling might have affected the networks' calculations. The region for assessing the BMD-value is only the femoral neck. Although this is known, the ROI-images contain not only the femoral neck, but rather both the femoral neck and the trochanter. This could have affected the outcome of the networks. There might have been information about fracture risk in the femoral shaft that has not been considered. The smaller ROI-images are only the original images viewing the region of the proximal femur that is most prone to fractures.

5.5 Limitations

From a clinical point of view it would be useful if the fracture risk in the coming ten years or more could be predicted, but that is a more difficult problem than predicting for the coming five years. Risk assessment tools such as FRAX evaluate the 10-years probability. A five year threshold was chosen as a starting point and if the networks were successful another threshold can be chosen.

The DXA-images that have been used in this project are all images from men. There is an anatomical difference between the hips of a man and the hips of a woman. The hip of a woman is usually wider than that of a man. In order to produce a more general result, hip images of women should also be used. Also, the T-score is based on the mean of a young female adult, so that might also have affected the result of this thesis due to the anatomical difference.

5.6 Ethics

The common practice today is to evaluate the fracture risk based on a combination of the BMD-values of the hip and documented risk factors. To assess the BMD-value, DXA-images of the hip must be taken. Without any BMD-values, the risk assessment would not be as accurate. However, even with the values, the risk assessment is still not a good predictor.

Using ANNs to predict the fracture risk also needs the DXA-images to be taken. The level of radiation a patient would be exposed to would stay the same. But if an ANN performs the prediction better and gives more

accurate risk assessment, the chance of prevention would be greater and it would more advantageous for the patient. All fractures that are prevented will save suffering for the patient and resources in society.

No personal information of the patients were fed to the networks. This could ensure anonymity for the participants in the study. Also, since no new images were taken especially for this thesis project, there was no need of anonymising any patient information further.

Introducing artificial intelligence in diagnostic tools also means that even in the medical field, computer-aid is gaining acclaim and is used more frequently. Computers are better at analysing data and is therefore superior when it comes to diagnostics that are based on statistics. The diagnoses that are obtained by the computer programs are still evaluated by a human in order to make the diagnosis more reliable. However, as the artificial intelligence becomes more advanced, the less mistakes will it make and the more seldom will the outcome be evaluated. The questions that need to be asked then are: what if the computers one day make a fatal mistake? Who is responsible and who will take the blame? Putting our trust in machines is tempting, but how much can we trust them?

6. Conclusions and future work

This thesis project has concluded that the first step towards using ANN to assess the hip fracture risk has been taken. The results indicate that there is more to be done within this path. Further investigation of using a pre-trained VGG-network should be carried out and other custom network architectures should be developed.

One of the major challenges with this project was the lack of positives and hip images in general. More images are therefore suggested to be retrieved and also other risk group, other than elderly men, should be analysed. The generalisation of the results from this thesis project can be questioned since the data has not been enough. Nonetheless, this first step has been important and from what have been accomplished, the future of this kind of projects is promising.

To develop neural networks that will perform even better than those presented in this thesis, more computational power is recommended. Even though a shallow network can do reasonably well, the variations of the hip images are still not fully covered. More images means more variation and a pre-trained network might then hold a better chance.

Bibliography

[1] M. J. Olszta, X. Cheng, S. S. Jee, R. Kumar, *et al.*, "Bone Structure and Formation: A New Perspective," 2007.

[2] L. Solomon, D. Warwick, and S. Nayagam, *Apley's Concise System of Orthopaedics and Fractures*. Oxford: Hodder Arnold, 3. ed. ed., 2005.

[3] World Health Organization, "WHO Scientific Group on the Assessment of Osteoporosis at Primary Health Care Level," *World Health Organization*, vol. May, no. May 2004, pp. 5–7, 2004.

[4] "International Osteoporosis Foundation - What is Osteoporosis?." `https://www.iofbonehealth.org/what-osteoporosis-0`. [Online; accessed 2018-09-06].

[5] "International Osteoporosis Foundation - Facts and Statistics." `https://www.iofbonehealth.org/facts-statistics`. [Online; accessed 2018-04-19].

[6] W. J. Tseng, Hung, L. Wei, J. S. Shieh, *et al.*, "Hip Fracture Risk Assessment: Artificial Neural Network Outperforms Conditional Logistic Regression in an Age- and Sex-Matched Case Control Study," *BMC Musculoskeletal Disorders*, vol. 14, no. 7, 2013.

[7] B. R. McCreadie and S. A. Goldstein, "Biomechanics of Fracture: Is Bone Mineral Density Sufficient to Assess Risk?," *Journal of Bone and Mineral Research*, vol. 15, no. 12, pp. 2305–2308, 2000.

[8] F. Chollet, *Deep Learning With Python.* Manning Publications, 2017.

[9] N. Buduma and N. Locascio, *Fundamentals of Deep Learning: Designing Next-Generation Machine Intelligence Algorithms*. Sebastopol, CA: O'Reilly Media, first ed., 2017.

[10] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-Based Learning Applied to Document Recognition," *Proceedings of the IEEE*, vol. 86, pp. 2278–2324, Nov 1998.

[11] A. Tiulpin, J. Thevenot, E. Rahtu, *et al.*, "Automatic Knee Osteoarthritis Diagnosis from Plain Radiographs: A Deep Learning-based Approach," *Scientific Reports*, vol. 8, no. 1, pp. 1–10, 2018.

[12] T. P. Ho-le and J. A. Eisman, "Prediction of Hip Fracture in Post-Menopausal Women using Artificial Neural Network Approach," *IEEE Engineering in Medicine and Biology Society*, pp. 4207–4210, 2017.

[13] Q. Liu, X. Cui, Y. C. Chou, *et al.*, "Ensemble Artificial Neural Networks Applied to Predict the Key Risk Factors of Hip Bone Fracture for Elders," *Biomedical Signal Processing and Control*, vol. 21, pp. 146–156, 2015.

[14] A. Géron, *Hands-on Machine Learning with Scikit-Learn and TensorFlow : Concepts, Tools, and Techniques for Building Intelligent Systems.* Beijing: O'Reilly, 2017.

[15] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning Representations by Back-Propagating Errors," *Nature*, vol. 323, pp. 533–536, Oct 1986.

[16] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," in *International Conference on Learning Representations 2015*, pp. 1–15, 2014.

[17] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, p. 770–778, 2016.

[18] B. D. Ripley, *Pattern Recognition and Neural Networks.* Cambridge: Cambridge Univ. Press, 1996.

[19] N. Donges, "Transfer Learning." `https://towardsdatascience.com/transfer-learning-946518f95666`. [Online; accessed 2018-10-01].

[20] J. Brownlee, "A Gentle Introduction to Transfer Learning for Deep Learning." `https://machinelearningmastery.com/transfer-learning-for-deep-learning/`. [Online; accessed 2018-10-01].

[21] "ImageNet." `http://www.image-net.org/index`. [Online; accessed 2018-10-18].

[22] L. van der Maaten and G. Hinton, "Visualizing High-Dimensional Data using t-SNE," *Journal of Machine Learning Research*, no. 9, pp. 2579–2605, 2008.

[23] H. Hotelling, *Analysis of a Complex of Statistical Variables into Principal Components*. Baltimore: Warwick & York, 1933.

[24] TensorFlow. `https://www.tensorflow.org/`. [Online; accessed 2018-03-28].

[25] Keras. `https://keras.io/`. [Online; accessed 2018-03-28].

[26] E. Orwoll, J. B. Blank, E. Barrett-Connor, J. Cauley, *et al.*, "Design and Baseline Characteristics of the Osteoporotic Fractures in Men (MrOS) Study - A Large Observational Study of the Determinants of Fracture in Older Men," *Contemporary Clinical Trials*, vol. 26, no. 5, pp. 569–585, 2005.

[27] D. Mellström, O. Johnell, Ö. Ljunggren, *et al.*, "Free Testosterone is an Independent Predictor of BMD and Prevalent Fractures in Elderly Men: MrOS Sweden," *Journal of Bone and Mineral Research*, vol. 21, no. 4, pp. 529–535, 2006.

[28] N. C. Harvey, A. Odén, E. Orwoll, *et al.*, "Falls Predict Fractures Independently of FRAX Probability: A Meta-Analysis of the Osteoporotic Fractures in Men (MrOS) Study," *Journal of Bone and Mineral Research*, vol. 33, no. 3, pp. 510–516, 2018.

A. ROC-curve based on original data set

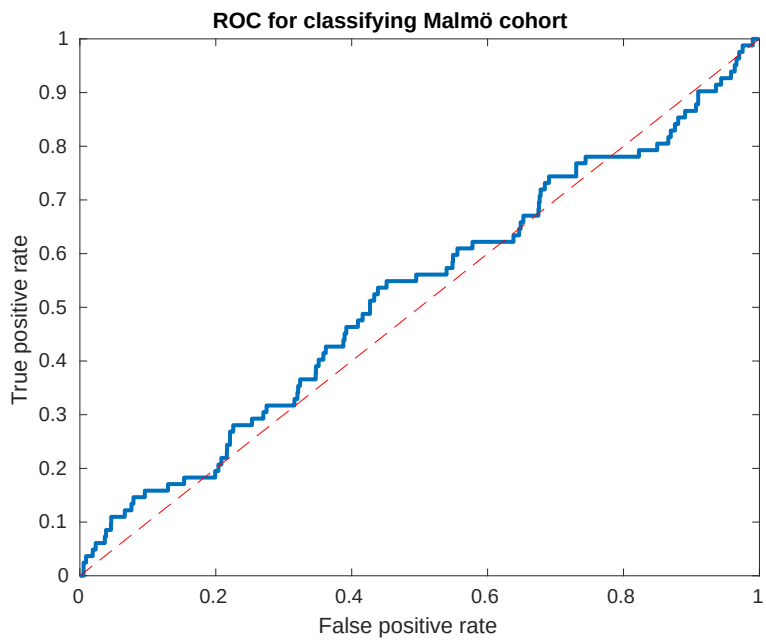


Figure A.1: Plot of the ROC-curve of the results given by the own shallow model. The AUC is 0.5222 and the accuracy is 0.9679. The model is trained on the original images, with no data augmentation. The architecture of the network can be found in Table 3.3.

B. Comparison of different pre-trained models

The small set of hip images, consisting of five positives and five negatives, was then trained with another pre-defined and pre-trained network, VGG-16. This network is also pre-trained on ImageNet and has the same default input size as ResNet. To clarify, the architecture was again as in Table 3.4 but with a VGG-model instead of a ResNet-model. This network was also trained until the accuracy was equal to one and the loss close to zero for both data sets, hips and the images from ImageNet, and was able to correctly classify all the images. Since the VGG-model seemed to be able to separate positives and negatives, architectures such as in Table 3.4 and 3.5 were tested with a VGG-16 instead of ResNet-50. These architectures were of limited success because they could not be tuned properly due to lack of time and GPU-memory (6 GB).

C. ROC-curve based on BMD

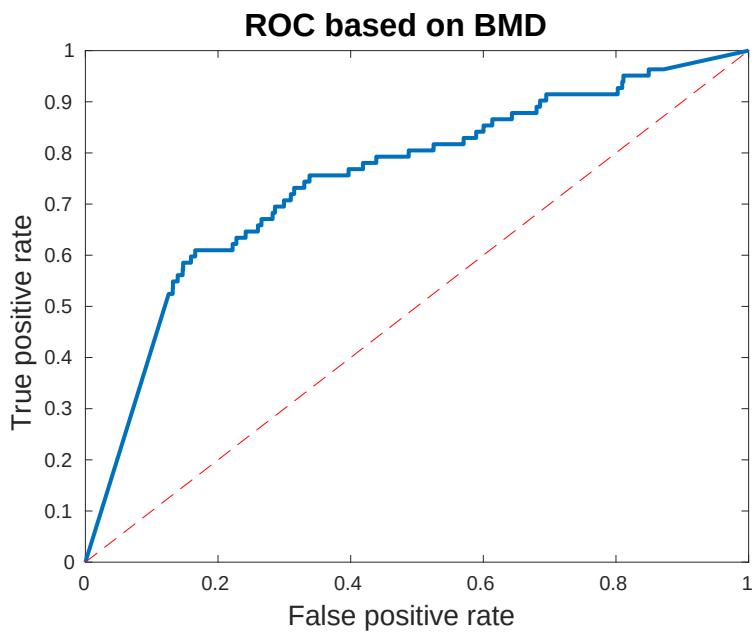


Figure C.1: Plot of the ROC-curve based on BMD from Malmö cohort. The AUC is 0.7497 and the accuracy is 0.8367.