Master of Science Thesis in Biomedical Engineering
Department of Biomedical Engineering, Linköping University, 2019

Synthesis of Thoracic Computer Tomography Images using Generative Adversarial Networks

Julia Hagvall Hörnstedt

Master of Science Thesis in Biomedical Engineering

Synthesis of Thoracic Computer Tomography Images using Generative Adversarial Networks:

Julia Hagvall Hörnstedt

LIU-IMT-TFK-A–19/566–SE

Supervisor: **Anette Karlsson**
IMT, Linköping University
Fredrik Noring
Combitech AB
Tim Fornell
Combitech AB

Examiner: **Magnus Borga**
IMT, Linköping University

Department of Biomedical Engineering
Linköping University
SE-581 83 Linköping, Sweden

Abstract

The use of machine learning algorithms to enhance and facilitate medical diagnosis and analysis is a promising and an important area, which could improve the workload of clinicians' substantially. In order for machine learning algorithms to learn a certain task, large amount of data needs to be available. Data sets for medical image analysis are rarely public due to restrictions concerning the sharing of patient data. The production of synthetic images could act as an anonymization tool to enable the distribution of medical images and facilitate the training of machine learning algorithms, which could be used in practice.

This thesis investigates the use of Generative Adversarial Networks (GAN) for synthesis of new thoracic computer tomography (CT) images, with no connection to real patients. It also examines the usefulness of the images by comparing the quantitative performance of a segmentation network trained with the synthetic images with the quantitative performance of the same segmentation network trained with real thoracic CT images. The synthetic thoracic CT images were generated using CycleGAN for image-to-image translation between label map ground truth images and thoracic CT images. The synthetic images were evaluated using different set-ups of synthetic and real images for training the segmentation network. All set-ups were evaluated according to sensitivity, accuracy, Dice and F2-score and compared to the same parameters evaluated from a segmentation network trained with 344 real images.

The thesis shows that it was possible to generate synthetic thoracic CT images using GAN. However, it was not possible to achieve an equal quantitative performance of a segmentation network trained with synthetic data compared to a segmentation network trained with the same amount of real images in the scope of this thesis. It was possible to achieve equal quantitative performance of a segmentation network, as a segmentation network trained on real images, by training it with a combination of real and synthetic images, where a majority of the images were synthetic images and a minority were real images. By using a combination of 59 real images and 590 synthetic images, equal performance as a segmentation network trained with 344 real images was achieved regarding sensitivity, Dice and F2-score.

Equal quantitative performance of a segmentation network could thus be achieved by using fewer real images together with an abundance of synthetic images, created at close to no cost, indicating a usefulness of synthetically generated images.

Acknowledgments

I would like to direct a big thank you to my supervisors Anette Karlsson, Fredrik Noring and Tim Fornell for their guidance, help and support throughout this thesis work. I would also like to thank David Abramian for the help and the interesting and educative discussions regarding GAN. A special thank you to Johnny Larsson for giving me the opportunity to perform the thesis work at Combitech.

I would also like to thank all my friends and family for supporting me through all ups and downs during this thesis work. Your encouraging words have meant a lot!

Linköping, May 2019
Julia Hagvall Hörnstedt

Contents

Notation

Dictionary

Word	Meaning
Ground truth image	Binary segmentation mask of lung nodule
Thorax	Latin word for the rib cage
U-net	Segmentation network
Voxel	Three dimensional equivalence to pixel

Abbreviations

Abbreviations	Meaning
CAD	Computer-Aided Diagnosis software
CNN	Convolutional Neural Networks
CT	Computed Tomography
GAN	Generative Adversarial Networks
GT	Ground Truth
HU	Hounsfield units
IDRI	Image Database Resource Initiative
LIDC	Lung Image Database Consortium
MAE	Mean Absolute Error
MRI	Magnetic Resonance Imaging
MSE	Mean Square Error

1 Introduction

Computer-aided diagnosis (CAD) software using machine learning algorithms have the potential to improve and optimize clinicians' workload if they are efficient and perform reliably [1]. In order to construct reliable CAD software it is important to have a sufficient data volume for training and validation of the algorithms [2]. To be able to perform image segmentation, the data sets also need to be annotated which is usually performed manually by one or several radiologists [2]. Sufficient annotated medical data sets are in general hard to find publicly due to restrictions concerning the sharing of patient data [2]. By synthetically being able to produce realistic images, with no connection to any patients, the public distribution of sufficient medical data sets as well as the sizes of the data sets could increase [2].

This master thesis was performed at Combitech AB and examined at the Department of Biomedical Engineering, IMT, at Linköping University with the purpose of synthetically generate thoracic Computer Tomography (CT) images using Generative Adversarial Networks (GAN). The following chapter presents the studied problem, the motivation to why the problem is of interest together with the problem formulation the thesis intend to answer.

1.1 Background

Machine learning is an approach with the goal of teaching a computer to be able to detect patterns in digital data and use these patterns to predict a certain output [3]. It is an application of artificial intelligence useful to process big amount of data and to help perform decision making based on that data [3]. The interest in machine learning approaches has grown in recent years due to advances in the field of machine learning, in form of deep learning. This is due to both increasing

amount of available data and due to the development of better and more powerful computers [4].

The use of machine learning algorithms to enhance and facilitate medical diagnosis and analysis is a promising and an important area, which could improve the workload of clinicians' substantially [5]. In order for machine learning algorithms to learn a certain task, large amount of data called training data, needs to be available for the computer to learn the underlying patterns. Data sets for medical image analysis are rarely public due to restrictions concerning the sharing of patient data. It often requires consent from the patient in order to be able to share the information, as well as anonymization of the data [6]. By being able to share medical data and distribute the data publically, it allows researcher to build on the work of others. I also allows the possibility to improve existing machine learning algorithms due to increase in available data.

Recently it has been proposed to use GAN as a method for synthesis of new medical images with no connection to real patients [2]. Synthetically produced medical images would be completely new images and therefore have no connection to any real patients, acting as an anonymization tool that enables distribution of medical images. It would also be a tool to create new training data and serve as a way to create sufficient variability within data sets [2]. GAN is a machine learning application that has gained a lot of interest since its development by Goodfellow *et al.* in 2014 [7]. GANs contains two types of neural networks: one generative network and one discriminative network [7]. The generative network is trained to be able to generate data as similar as possible to the target data and the discriminative network is trained to be able to distinguish between generated and real data [7]. The adversarial network therefore learns to estimate the distribution of targeted data and can produce completely new data with that same distribution [7]. This method has become popular to use for image-to-image translation where the adversarial network learns the mapping from an input image to an output image and can be used to map segmented label maps into images or black and white images into color images, along with a variety of other applications with good results [8].

It has also been found that a deep learning segmentation network (U-net) trained with synthetically produced medical images of retinas can perform almost equally good as the same U-net trained with real images [9]. This implies that the synthesized images are almost equally good as the real images, used for the particular network, and can be used instead of real images without loosing to much performance of the network.

Most work done on synthesizing of medical images is done in two dimensions whereas CT images often are three dimensional. It would therefore be interesting to evaluate the possibility to generate 3D medical images with Generative Adversarial Networks as well as its usability.

1.2 Aim and purpose

This thesis aims to investigate the possibility of synthesis of thoracic CT images using GAN and evaluate the usefulness of the images. The main evaluation of the usefulness of the images is done by using the images to train a segmentation network and evaluate the performance of the network.

The adversarial network is trained to map segmented label maps into thoracic CT images. The segmented label maps are generated from the data set and randomly combined with lung nodules in order to be able to create a completely synthetic data set of images with a large variation.

1.2.1 Problem statements

The following problem statements are answered in this thesis:

- Is there any difference in quantitative performance of a segmentation network trained with synthetic CT images compared to real CT images?
- If a segmentation network is trained solely on synthetic data, is it possible to outperform a network trained only on real data?
- Is it possible to increase the quantitative performance of a segmentation network by training it with a combination of synthetic and real images, where a majority of the images are synthetic images and a minority is real images, compared to a network trained with only real images?

1.3 Limitations

The master thesis project was conducted during 20 weeks with the following limitations:

- Only lung nodules larger than 3 mm are considered from the data set
- Only volumes of 128 x 128 x 128 voxels are generated by the network
- The hyperparameters of the U-net used for segmentation will not be optimized

2

Theory

The following chapter describes the relevant theory for this thesis. The basis of machine learning and deep learning is first presented to give the reader a better understanding of the function and use of GANs. The theory behind the evaluation method of deep learning networks is also described. Lastly, a brief introduction to computed tomography is also given to give a better understanding of the data used in this thesis.

2.1 Machine learning

Machine learning is a type of artificial intelligence useful for detecting patterns in large amount of data. By using the detected patterns, the machine learning algorithm can predict future data or perform decision making under uncertainty [3]. Machine learning is essentially a form of applied statistics where computer algorithms learn to statistically estimate complicated functions where some kind of uncertainty is involved [4]. Different kind of algorithms are used extensively today such as spam filters, handwriting recognition and face recognition among others [3].

Machine learning algorithms are often divided into two categories: supervised learning and unsupervised learning, where supervised learning is the form most widely used in practice [3]. In supervised learning, the goal is to learn a mapping from inputs $\mathbf{x}$ to outputs $\mathbf{z}$ given a set of labeled input-output pairs [3]. Each pair consists of an input x_i and an output label called z_i, where the label can represent, for example, a class the input belongs to. The inputs consists of a set of features, which represents real features from the data that is being processed [3]. The most common type of supervised learning task is classification where the algorithm's goal is to take a set of inputs $\mathbf{x}$ and return the correct class labels $\mathbf{z}$,

for each input [3]. The correct label is assigned by generalizing each input from a function learnt through training with input-output pairs [4].

To classify whether a flower belongs to a certain species, the input to the machine learning algorithm could be petal length, petal width or other features from the flower and the label would represent the specific species. The algorithm would be trained on data containing certain features and the respective species they belong to. It then learns a function for mapping the features to the correct classes. After the training, the algorithm takes input features, and with the learnt function from training, predicts a class for each input.

In order for the machine learning algorithm to learn, it needs to be able to evaluate its own performance. The method used to evaluate the performance often depend on the specific task that the algorithm is set to perform [4]. For classification, the accuracy of the model is usually used and it is calculated as the amount of inputs the algorithm classifies correctly out of all inputs [4]. The performance of the algorithm can also be measured as an error rate, the proportion of inputs that the algorithm outputs incorrectly out of all inputs, which is the inverse of the accuracy [4]. The performance measurement is often called loss function. This loss function is used to optimize the algorithm to perform as good as possible, which is performed by an optimizing algorithm [4].

2.1.1 Neural networks

Neural networks are machine learning systems that are designed to model the way that the human brain performs tasks or functions [10]. A set of synapses acts as connecting links and are represented by a set of weights [10]. The synapses, the weights, receives information or input signals. These signals, $\mathbf{x}$, are multiplied by their synaptic signal, $\mathbf{w}$, and transferred to the cell body, the adder [10]. All signals from the different synapses are added together, limited to a certain range by an activation function and sent forward. A schematic sketch of the propagation in a neural network can be seen in Figure 2.1.

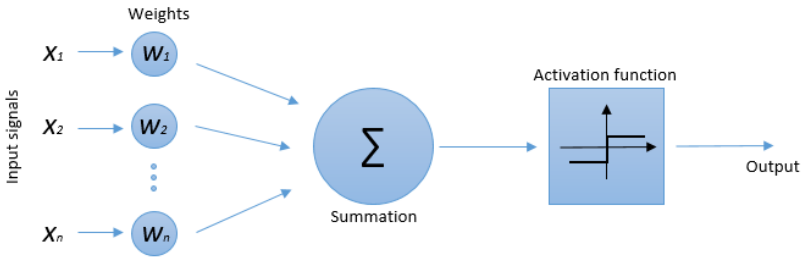


Figure 2.1: *One layer neural network architecture.*

A basic architecture of a neural network can be seen in Figure 2.2. The neural network has one input layer, two hidden layers and one output layer where all layers are connected with weights. All nodes consists of an adder and an activation function, as seen in Figure 2.1. The network presented is called a fully connected network, since all nodes in a layer are connected to all nodes in the following layer.

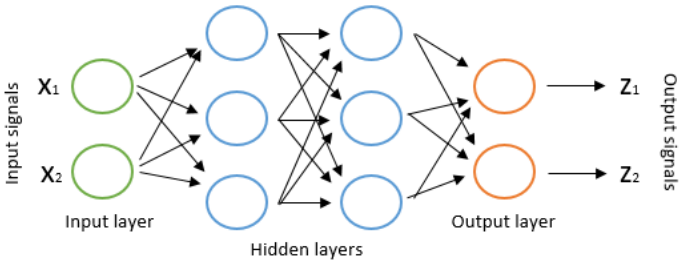


Figure 2.2: *Schematic representation of the architecture of a neural network, where the arrows represents the weights and the circles represents the nodes.*

Each node in the layers works as in Figure 2.1 where all inputs are multiplied with the weights of the node, $\mathbf{w}$, and added with the bias b. They are then summed together to produce the output of the node according to

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

where n is the number of inputs, y is the layer output and b is a bias. A bias is added to the input of each layer to increase or lower the net input of the activation function allowing it to shift [10]. These networks are usually called feedforward networks due to that information flows forward through the network without any feedback connections [4].

Activation function

To be able to compute the values in the hidden layers in the neural network, an activation function is required [4]. The activation function defines the output of the neurons in the layer based on its inputs [10], and it is the last step of each node as seen in figure 2.1. The activation function also restricts the output from the neuron in a predetermined interval. A commonly used activation function is the Rectified Linear Unit, usually called ReLU, defined as

$$f(z) = \begin{cases} z & \text{if } z \geq 0 \\ 0 & \text{if } z < 0. \end{cases} \tag{2.2}$$

ReLU is usually used in deep networks as it prevents the problem of vanishing gradients in neural networks [11]. This thesis uses ReLU as an activation function together with leaky ReLU defined as

$$f(z) = \begin{cases} z & \text{if } z \geq 0 \\ 0.01z & \text{otherwise.} \end{cases} \tag{2.3}$$

The slope of the regular ReLU is always zero in the negative part. If a neuron is stuck in the negative side of the activation function, it will always output zero and the neuron is then considered dead. LeakyReLU solves this problem as it does not have any zero slope parts.

The hyperbolic tangent, also know as tanh, is another activation function used in this thesis. Tanh has a range between -1 and 1, a difference compared to the previously described ReLU activation function. The hyperbolic tangent is defined as

$$\tanh(z) = \frac{2}{1 - e^{-2z}} - 1. \tag{2.4}$$

Due to the range between -1 and 1, only values near zero will be mapped to zero when using tanh as an activation function.

The Fermi function is also a classic activation function commonly used. The Fermi function is defined as

$$F(z) = \frac{1}{1 + e^{-x}}, \tag{2.5}$$

where the function outputs a continuous range between 0 and 1. The Fermi function is usually used in classification problems. In this thesis, the Fermi function is used as an activation function in the output layer of the GAN discriminator.

Loss function

The loss function is used to calculate the error of the models output. This loss function is used to train the neural network [4] to achieve optimal performance. The error can be calculated in several different ways, depending on the task. For calculating the error between two images, Mean Absolute Error, shortened as MAE, can be used as a loss function. The MAE is calculated as

$$\epsilon = \frac{1}{n} \sum_{i=1}^{n} |z_i - \hat{z}_i|, \tag{2.6}$$

where z is the correct output, $\hat{z}$ is the predicted output by the model and n is the number of outputs. It represents the mean of all absolute errors made by the model. Comparing two images, this means the mean of the absolute value of all pixel errors summed together.

Another usual loss function is the Mean Square Error, shortened as MSE, calculated as

$$\epsilon = \frac{1}{n}\sum_{i=1}^{n}(z_i - \hat{z}_i)^2, \tag{2.7}$$

where z is the correct output, $\hat{z}$ is the predicted output by the model and n is the number of outputs. It represents the mean of the square of all pixel errors summed together. Since the MSE loss function squares the difference between the output and the correct value, possible outliers have a larger impact on the error.

Optimization

The most used optimization algorithm for machine learning applications is Stochastic Gradient Descent (SGD) [4]. SGD is a form of gradient descent that optimizes the machine learning algorithm based on maximizing or minimizing some function $f(x)$ by alternating x [4]. The optimization algorithm is used to minimize the error calculated by the loss function, by changing the weights in the network. The weights are updated using back-propagation according to the gradient of the loss function with the goal to converge on a global minimum, indicating the most optimal weights. The weights are updated according to

$$w_{k+1} = w_k - \eta\frac{\partial\epsilon}{\partial w_k}, \tag{2.8}$$

where w is the weight, η is the learning rate and $\frac{\partial\epsilon}{\partial w_k}$ is the derivative, the gradient, of the loss for a certain weight w for iteration k. The learning rate is a hyperparameter that is not optimized when using regular gradient descent [4]. When updating the weights in the network, the weights of the biases are also updated. When using gradient descent, all samples in the training set are used to update the parameters, while in stochastic gradient descent only some samples are chosen randomly to update the parameters.

The adaptive moment estimation optimizer, shortened as Adam, is used in this thesis. Adam was first introduced by Kingma and Ba and is a form of stochastic gradient descent where both the first and the second moments of the gradient are used to compute individual learning rates for different parameters [12]. Adam compares favourably to other stochastic methods in both memory requirements as well as computational efficiency [12].

Regularization

Regularization is a technique used to prevent overfitting, when the model performs much better on training data compared to test data [4]. There are several different ways of performing regularization in a network, for example; *data augmentation*, *drop out*, *batch normalization* and *instance normalization* [4].

Data augmentation is an efficient way of improving the performance of a network. By adding more data to the training of the network, it learns to generalize better [4]. This can be done by creating synthetic data and adding it to the training data. When using images, cropping and translation of the images are useful techniques for augmentation [4].

Drop out is a method of randomly selecting some nodes, at each iteration of the network, and remove their input and output [4]. The canceled nodes are sampled independently from each other and the probability is chosen by a hyperparameter [4]. By doing this, it prevents the nodes to co-adapt too much. When training the whole network at once, some nodes may change due to errors caused by other nodes, which causes co-adaptations between the nodes. By continuously selecting different nodes when propagating through the network, it forces the parameters of the specific node to only correct their own mistakes and thus preventing overfitting [13].

Batch Normalization improves the stability of performance of the network by normalizing each set of input training data, called batches. The network normalizes the output of a previous activation layer by subtracting the batch mean and dividing by the batch standard deviation according to

$$H' = \frac{H - \mu}{\sigma}, \tag{2.9}$$

where H is a minibatch of activations of a previous layer to normalize, μ is a vector containing the mean of each unit and σ is a vector containing the standard deviation of each unit [4].

Instance Normalization improves the stability of performance of the network by normalizing each channel in each training sample, instead of each batch as batch normalization [14]. This is done by subtracting the mean and divide by the standard deviation over each channel.

2.2 Deep learning

Deep Learning refers to neural networks with several hidden layers, where the amount of hidden layers determines the depth of the model [4]. The idea with ordinary neural network and back-propagation originates from the 1980's and has not changed much since then [4]. The spark in deep learning in recent years is due to bigger data sets and data availability as well as the ability to use deeper networks as a result of better computers [4]. Deep learning algorithms are expected to be applied to more tasks in the future and their performances are expected to improve by advances in optimization algorithms and model design [4].

2.2.1 Convolutional Neural Networks

Convolutional Neural Networks (CNN) is a special kind of neural networks containing at least one convolutional layer in its architecture [4]. The convolutional layer applies convolution as a mathematical operator between layers instead of matrix multiplication, used by the original neural networks. This has shown to be good for processing data with grid like topography, as images [4]. The weights of the convolutional layer are arranged as a scalar in a one dimensional kernel and convolved with the input to generate the output of the layer according to

$$s(t) = (x * w)(t) = \sum_{a=-\infty}^{\infty} x(a)w(t-a), \tag{2.10}$$

where x is the input and w is the weight [4]. Usually, neural network implementations do not implement the flip of the kernel, relative the input, as seen in equation 2.10. Instead they only use the cross-correlation, the same definition without the flip [4]. For simplicity, the mathematical operator is still called convolution. The use of convolutional layers is motivated by sparse interactions and parameter sharing. In a conventional neural network, all neurons in one layer is connected to all neurons in the next layer. Convolutional layers have sparse interactions, meaning that not all neurons are connected between two layers due to the use of kernels smaller than the input [4]. This results in fewer parameters in the model which reduces memory requirements and improves the statistical efficiency [4]. Parameter sharing also reduces the total amount of parameters in the network. In a conventional neural network, each element of the weight matrix is used exactly once when computing the output layer meanwhile in convolutional layer each part of the kernel is used at every position of the input [4].

One convolutional layer in a convolutional network usually performs several convolutions in parallel. This is done to produce a set of linear activations [4]. Each activation is then run through an activation function before the output is modified using a pooling function [4]. The pooling function is used to downsample the outputs of the convolutions and is done by replacing the outputs at certain points with a summary of statistic based on the nearby points [4].

2.3 Generative Adversarial Networks

The idea of generating images by using adversarial networks was first proposed by Goodfellow *et al.* in 2014 [7]. The framework proposed contains two convolutional neural networks trained simultaneously via an adversarial process; a generator (G) and a discriminator (D). The generator is trained to generate data with the same distribution as the training data meanwhile the discriminator is trained as an adversary to the generator, to be able to distinguish between real data and generated data by G [7]. By training both networks simultaneously, the generator improves until the discriminator no longer can differ between the real and fake data [7]. The generator can be seen as a team of counterfeiters trying to

create fake money and use it, while the discriminator is the police trying to detect the fake money. The competition between them forces both groups to improve until the fake money is as real looking as the real money and the police can no longer distinguish between them [7].

In the first proposed GAN model, the generator's distribution p_g over some data $\boldsymbol{x}$ is learnt by synthesis of input noise $p(z)$. The noise is then mapped to $G(z, \theta_g)$ where G is the function represented by the network parameters θ [7]. The synthesized data are then passed to the discriminator, $D(x, \theta_d)$, which outputs a scalar representing the probability of $\boldsymbol{x}$ being from the real data compared to the distribution p_g. The discriminator is trained to maximize the probability of assigning the correct label to both the real data as well as the data generated by G, meanwhile the generator is trained to fool the discriminator to assign a high probability of the generator samples to belong to the real data. This is represented by minimizing $log(1 - D(G(z)))$, which can be interpreted as maximizing the probability of the discriminator to assign the wrong label to the output of the generator [7]. The resulting minmax game between the generator and the discriminator can be described as

$$\min_G \max_D V(D, G) = \mathbb{E}_{x \sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{x \sim p_z(z)}[\log(1 - D(G(z)))], \qquad (2.11)$$

where p_{data} is the real data distribution and p_z is the noise distribution. As the training proceeds it is harder and harder for the discriminator to classify the generated data correctly as the generator develops. Eventually this results in a probability of 0.5 to classify the input correctly where the discriminator classifies each input only by chance [7].

Since the first proposed GAN model by Goodfellow *et al.* in 2014, a large amount of new adversarial models has been proposed for various applications. According to a review article of GANs used in medical imaging published in 2018, the most popular method for image-to-image translation is Pix2Pix [15], which is described further in section 2.3.3.

2.3.1 Training Generative Adversarial Networks

The goal of the adversarial model is to find what is called the Nash equilibrium to the two player game between the generator and the discriminator. This happens when neither one of the networks has anything to gain from updating their weights with respect to the loss. The discriminator is trained by feeding it generated and real data together with their respective labels. During this phase, the weights of the discriminator is updated using back-propagation. To train the generator, the generator is stacked together with the discriminator. The generated data from the generator is directly fed to the discriminator for validation. During this phase, the weights of the generator are updated using back-propagation based on the result from the stacked discriminator and the weights of the discriminator are frozen to optimize the generator. Most adversarial networks are

trained using gradient descent, designed to find a local or global minimum of the loss function rather than to find the Nash equilibrium [16]. Since the generator and discriminator are trained sequentially, this can cause failure in converging and the gradient descent enters a stable orbit instead [16].

2.3.2 Conditional Generative Adversarial Networks

Mirza and Osindero suggested in 2014 a conditional GAN model to be able to control the data generated by the network [17]. By conditioning both the generator and the discriminator with some auxiliary information, as for example a class label, the conditional GAN model can generate data with the same label [17]. To perform the conditioning, the conditional information is fed to both networks as an additional input layer [17]. The resulting minmax game between the generator and the discriminator for the conditional GAN can be described as

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}[\log D(x|y)] + \mathbb{E}_{x\sim p_z(z)}[\log(1-D(G(z|y)))], \quad (2.12)$$

where both x and z are conditioned with condition y. The model was bench marked by Mirza and Osindero using the MNIST data set [1], where images representing certain numbers was generated with their respective label as a condition [17]. The conditional GAN model is a key contribution for applications in domain translation of images using adversarial networks.

2.3.3 Image-to-image translation

In a paper by Isola *et al.* from 2017, the use of conditional GANs was investigated as a general-purpose solution for image-to-image translation [8]. The generator is presented with an image from domain X with the purpose of translating the images into domain Y where the results should be indistinguishable from real images from domain Y. To encourage less blurring of the generated images, Isola *et al.* added an L1 loss term, the absolute error, to the original adversarial loss used in GANs. The total loss of the proposed model can be described as

$$G^* = \arg\min_G \max_D \mathbb{E}_y[\log D(y)] + \mathbb{E}_{x,z}[\log(1-D(G(x,z)))] + \mathbb{E}_{x,y,z}[\|y - G(x,z)\|_1]. \quad (2.13)$$

The authors successfully showed that the proposed model, called Pix2Pix, could perform a good image-to-image translation with real looking generated images on a variety of data sets including translating semantic labels into photos, maps into aerial photos and black and white images into colored images [8].

2.3.4 Cycle-consistent image-to-image translation

Zhu *et al.* introduced in 2017 a new GAN model, called CycleGAN, based on the model presented by Isola *et al.*, for image-to-image translation of unpaired

[1] Available at: http://yann.lecun.com/exdb/mnist/

images. The approach attempts to learn the translation from a source image domain X to a target image domain Y by introducing a cycle-consistency loss to the adversarial training [18]. The CycleGAN model attempts to learn the two mapping functions $G : X \rightarrow Y$ and $F : Y \rightarrow X$ by using two different generator and discriminator pairs. To guarantee the correct mapping of input x_i to output y_i, a cycle-consistency loss is introduced. The authors use the analogy that by translating an English sentence to French, the sentence should be the same as the original sentence if translated back to English again.

For each image x from image domain X it should be possible to translate the image back to its original domain according to $x \rightarrow G(x) \rightarrow F(G(x)) \approx x$. It should as well be possible to translate image y from image domain Y back to its original domain according to $y \rightarrow F(y) \rightarrow G(F(y)) \approx y$. This is called forward cycling consistency and backwards cycling consistency [18]. This can be seen in Figure 2.3.

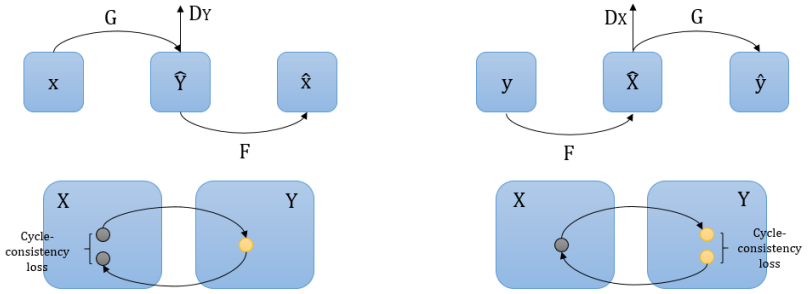


Figure 2.3: *Visualization of cycle-consistency loss in CycleGAN.*

By using and minimizing the cycle-consistency loss, it is possible to learn the mapping functions G and F without paired images [18].

2.4 Evaluation

To evaluate adversarial networks, two kind of evaluations can be performed. Firstly, the images can be inspected visually to see how they correlate to real images. Secondly, the performance of a neural network trained with the generated images compared to the same network trained with real images will give an estimation of the quality and usefulness of the images. A significance test can be used to evaluate the comparison between a neural network trained with synthetic images and a neural network trained with real images.

2.4.1 Neural segmentation network

Several parameters of a neural network can be evaluated in order to estimate its performance. This is done after the training phase, when all hyperparameters of the network are set. The parameters used to evaluate 3D medical image segmentations can differ between different methods, but sensitivity, Dice and F2-score belongs to the commonly ones used [19]. All evaluation parameters are based on the instances in the confusion matrix shown in Figure 2.4.

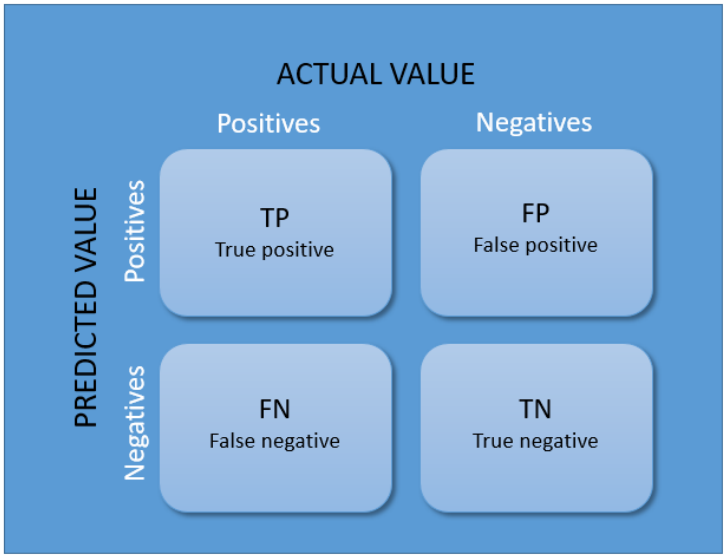


Figure 2.4: *Confusion matrix showing the instances the evaluation parameters are based on.*

When having a network that aims to segment nodule voxels from healthy voxels, the true positives (TP) are voxels that are segmented into the nodule class that correctly belongs to the nodule class. The false positives (FP) are voxels that belongs to healthy tissue but are segmented to belong to the nodule class by the network. The false negatives (FN) are nodule voxels that are not segmented by the network and are therefore regarded as healthy tissue and true negatives (TN) are healthy tissue voxels that are regarded as healthy tissue voxels by the network.

Sensitivity

The sensitivity of a segmentation network can be calculated as

$$\text{Sensitivity} = \frac{TP}{TP + FN}, \tag{2.14}$$

and is a measure of how well a network identifies positive cases, the true positive rate. This corresponds to the percentage of voxels that is correctly classified as cancer voxels out of all cancer voxels present in the image.

Accuracy

The voxel wise accuracy of the network is calculated as

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}, \tag{2.15}$$

and is a measure on how well the network performs in general. The accuracy measurement gives a percentage of correctly classified voxels from all voxels in the image.

Dice

The dice score is calculated as

$$\text{Dice} = \frac{2 \cdot TP}{2 \cdot TP + FP + FN}, \tag{2.16}$$

and is a measure which combines precision and sensitivity. The dice score rewards true positives and is a good measure on how well the network detects positive cases, i.e cancer voxels.

F2-score

The F2-score is calculated as

$$\text{F2} = \frac{5 \cdot TP}{5 \cdot TP + FP + 4 \cdot FN}, \tag{2.17}$$

where false negatives are more weighted than false positives. This is important when classifying and segmenting nodules since missed cancer voxels effects the usefulness of the model distinctly.

2.4.2 Significance test

A t-test can be used to determine if the mean of two sets of data, or two parameters, are significantly different from each other [20]. The t-test returns a probability of the null hypothesis to be correct, where the null hypothesis is that there is no difference between the two sets. If there is a significant difference between the mean of the two sets of data, the null hypothesis will fail to be accepted [20].

2.5 Computed tomography

Computed tomography (CT) is a medical imaging modality that produces volumes represented by 2D image slices by utilizing x-rays [21]. Cross-sectional

digital images are formed from reconstruction of a set of projections that are attained by shooting and collection x-rays from different angles around a subject [22]. Most CT scanners use both a rotational source and a rotational detector to obtain profiles from all angles of the patient [21]. 3D volumes are created by stacking together all obtained 2D image slices.

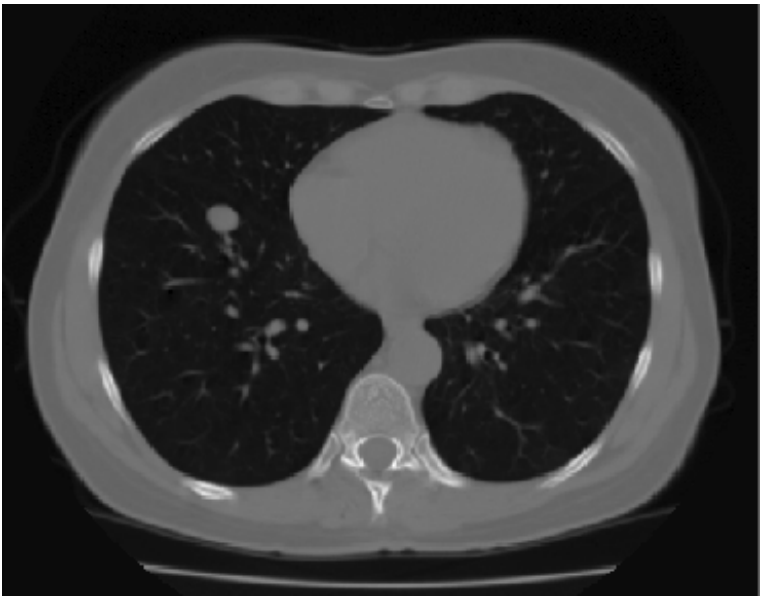


Figure 2.5: *Example of a CT image with high attenuation in the lungs and lower attenuation in bone and soft tissue.*

2.5.1 Attenuation

As the x-rays passes through tissue, the radiation interacts with the tissue and some of the beam energy is attenuated [22]. The amount of attenuation depends on the type of tissue it passes. The attenuation of an x-ray beam passing through tissue is measured and calculated as

$$I_x = I_0 e^{-\mu x} \tag{2.18}$$

where I_x is the x-ray beam intensity at a distance x from the source, I_0 is incident intensity of the x-ray beam and μ is the linear attenuation coefficient of the tissue [23]. As the tissue is not homogeneous, the attenuation can not be described using only one attenuation coefficient. The total attenuation depends on the local attenuation coefficient for each path of the x-ray during acquisition. The total attenuation can be seen as the sum of the attenuation for all present tissues as

$$I = I_0 e^{\sum -\mu_i x_i}. \tag{2.19}$$

The difference in attenuation between different tissue give rise to the appearance of the CT image. If the tissue attenuates all the energy, the image appears black and if the tissue does not attenuate any energy the image appears white.

2.5.2 Image formation

One rotation of the x-ray source and detector results in projections from 360 degrees of the patient. These projections are constructed into a 2D cross-sectional slice by reconstruction algorithms resulting in a digital image. The algorithm assigns a value to each pixel in the image depending on the average attenuation from all projections [21].

Since the attenuation is dependent on the energy of the x-ray beam, as seen in equation 2.18, it is hard to compare images obtained with different scanners. Therefore, all values in CT images are translated to their respective Hounsfield unit (HU).

Hounsfield units

HU is a quantitative value to describe the radio density, the attenuation, in CT images. The unit has an arbitrary scale normalized with regard to water, meaning that water has the value 0 HU and air has value -1000 HU [23]. The Hounsfield unit is calculated as

$$\text{HU} = 1000 \cdot \frac{\mu_0 - \mu_{H_2O}}{\mu_{H_2O}} \tag{2.20}$$

where μ_0 is the attenuation and μ_{H_2O} is the attenuation coefficient of water [23]. The reconstructed images can contain HU values varying from -1000 to +3000 meanwhile a display screen is usually only able to display 256 gray scale values. Thus, some type of windowing is used to represent the complete gray scale when displaying a CT image.

2.5.3 Lung nodules

Lung nodules, also known as pulmonary nodules, are abnormalities found in the lung tissue. Lung nodules are common radiographic findings and can be both cancerous and benign [24]. Due to the possibility of being cancerous, detection of lung nodules are of high importance and a diagnostic challenge in chest radiography [25]. Nodules are often detected and followed up using CT imaging, but can also be followed up using Positron Emission Tomography (PET) [24].

2.6 Data format

There are several ways to store image data in the medical imaging field. The image file formats provide a standardized way of storing information in a computer file. Usually, a medical data set contains one or several images that represents a projection of anatomical data onto an image plane or represented as slices in

a volume [26]. Two common medical file formats [26] are used in this thesis, DICOM and NIfTI, described in more detail below.

2.6.1 DICOM

DICOM, Digital Imaging and Communcations in Medicine, is a commonly used data format for medical images [27]. The format was invented to create a vendor-independent standard in radiology to enable communication of images, diagnostic information and any associated data [27]. All DICOM data come with an associated header called Information Object Definitions (IODs). The IOD contains attributes describing for example the modality used, the characteristics of the examination and technical details about the image acquisition [27].

2.6.2 NIfTI

The NIfTI file format was first created by a commitee at National Institutes of Health as a format for neuroimaging [26]. NIfTI files also provides a header with metadata together with the image, just as DICOM files. This header is often provided in the same file as the pixel data [26] with the possibility to extend the heather information [28]. NIfTI is still the preferable data format used in neuroimaging research meanwhile DICOM is most widely used in general [26].

3

Related work

The following chapter briefly presents the work, previously published, related to the work in this thesis. It presents work done in medical image synthesis using GAN both in two dimensions as well as in three dimensions. The chapter also presents the previously done pre-processing of the data from the data set used in this thesis as well as a description of the segmentation network, U-net, used to evaluate the work presented in the thesis.

3.1 Medical image synthesis using Generative Adversarial Networks

Synthetisation of medical images is one of the most important areas where GANs can be used [15], mostly due to the privacy regulations connected to medical data. GANs provide a more generic solution to the lack of data, by learning the image distribution, than regular translation of images and has been used in numerous works with promising results [15].

3.1.1 Two dimensional data

There are several interesting works done on generating x-ray images in two dimensions involving the thorax [29] [30]. Salehinejad *et al.* showed the use of a Deep Convolutional GAN to synthesize artificial chest x-rays to balance, create equal variability between classes, and expand a training data set used to train a deep convolutional neural network for classification of chest pathologies [29]. The generator uses a series of six strided convolutions to convert the projected and reshaped noise vector into a 256x256 pixel chest x-ray [29].

Chuquicusma *et al.* generated two dimensional slices of malignant and benign

lung nodules from the LIDC-IDRI data set using a similar Deep Convolutional GAN as Salehinejad *et al.* [30]. The network generator reshapes a noise vector into a 56x56 pixel nodule slice [30].

The Deep Convolutional GAN, DCGAN, structure was first proposed by Radford *et al.* [31] to scale up GANs using CNNs. The DCGAN architecture uses strided convolutions to replace pooling functions allowing the network to learn its own spatial downsampling in both generator and discriminator [31]. No fully connected layers are used in the network and batch normalization are applied to all layers except for the output layer in the generator and the input layer in the discriminator. This is done to help the gradient flow in the deeper model [31]. ReLU is used as an activation function in all layers of the generator, except for the output layer which uses tanh. For the discriminator, Leaky ReLU is used as an activation function in all layers [31].

3.1.2 Three dimensional data

Even though most work in medical image synthesis is done i two dimensions, there is some previous work published using GAN in three dimensions [2] [9] [32] [33] [34] [35].

Shin *et al.* generated MRI images of abnormal brains containing tumors from segmentation masks of brain anatomy and tumor using a modified Pix2Pix GAN [2]. The authors used two GANs to perform both synthetic image generation and image segmentation to be able to generate MRI images containing tumors from input label maps. By alternating the label maps into the adversarial networks a high level of variations in images can be obtained [2].

Guibas *et al.* proved the use of two GANs to generate label maps of retinas and to translate the label maps into corresponding photorealistic images of retinas [9]. The first GAN uses a 3D modified DCGAN architecture with the purpose to generate new segmentation label maps of the vessels in the retina. The second GAN uses 3D modified conditional GAN model to generate the corresponding real images of retinas from the segmented label maps [9]. The proposed model is able to generate medical images for a segmentation task end-to-end, using a pair of generative adversarial networks [9].

Nie *et al.* proposed a supervised deep convolutional GAN model for estimating a target image from a source image both between 3T MRI and CT [34] as well as between 3T MRI and 7T MRI [32]. As a generator architecture, the authors uses a Fully Convolutional Network to be able to preserve spatial information in a local neighborhood of the image space and for the discriminator the authors uses a regular CNN [32] [34].

Costa *et al.* proposed a method for end-to-end generation of retinal images by using a combined autoenconder and GAN [33]. Vessel tree images are first gen-

erated by using an adversarial autoenconder network that learns to copy the distribution of its training images. The vessel network images are then translated into retinal color images using a GAN with the Pix2Pix architecture. Both the autoencoder and the GAN are trained together to perform synthesis of new retinal images [33].

Yang *et al.* demonstrated MRI cross-modality translation of images using a conditional GAN architecture [35]. The conditional GAN architecture is similar to Pix2Pix, with some differences in the layer structure of the generator [35].

Pix2Pix is widely used in medical image synthesis for image-to-image translation where paired data is available [15]. The model performs well on a variety of different three dimensional medical image tasks, as described above.

3.1.3 CycleGAN

The CycleGAN method has mostly been used in medical imaging to map MRI images to CT image and vice versa for cadiac images as well as images of the brain [36] [37] [38]. CycleGAN has also been used to map images between different MRI weightings [39] [40]. Most work using CycleGAN has been done in two dimensions [15].

The advantage of CycleGAN, and the reason for the creation of the model, is the possibility to map images between two different domains without having paired data. The model learns to map between both image distributions due to the cycle-consistency loss. The disadvantage of using CycleGAN is the complex network architecture that demands high computational power. There is also very little documentation of the use of CycleGAN for three dimensional medical images synthesis.

3.2 Pre-processing of data

The data used in this thesis was previously, to this work, pre-processed as a part of another master thesis work by Bardolet Pettersson [41]. This was done to normalize the data and to ensure the quality of the data. A more detailed description about the data set used is given in section 4.1 below.

The following pre-processing steps were performed:

- The axial slices of the CT images were chosen.
- All images were converted into Int16 as data type.
- All voxels outside the scan field was set to a gray value that after conversion to Hounsfield units would represent air.

- All voxel values were normalized and converted into Hounsfield units according to

$$H = IV \cdot S + I \tag{3.1}$$

where H is the Hounsfield unit value, IV represents the attenuation, gray value, of each voxel, I is the rescale intercept and S is the rescale slope. Both I and S were obtained from the metadata of each image.

- Removal of artifacts by setting all voxel values over 1900 HU to the Hounsfield value of soft tissue.

- Normalization of voxel dimensions by resampling all images to the same voxel dimension, 0.48828125 x 0.48828125 x 1 mm^3.

- Conversion of file format from DICOM to NIfTI.

- Creation of ground truth data for each image from the corresponding XML-file.

- All images with missing metadata or other errors were discarded.

3.3 U-net

The segmentation network used in this thesis, to evaluate the generated images, is based on an architecture called 3D U-net [42]. The architecture of the 3D U-net is originally based on a 2D version of the network first presented by Ronneberg *et al.* in 2015 [43]. The network utilizes a U-structure for segmentation with a contraction encoder part, seen to the left in Figure 3.1, to analyze the whole volume and an expanding decoder part, seen to the right in Figure 3.1, to produce a full segmentation [42]. The network also uses skip connections between the encoder part and the decoder part, called concatenations.

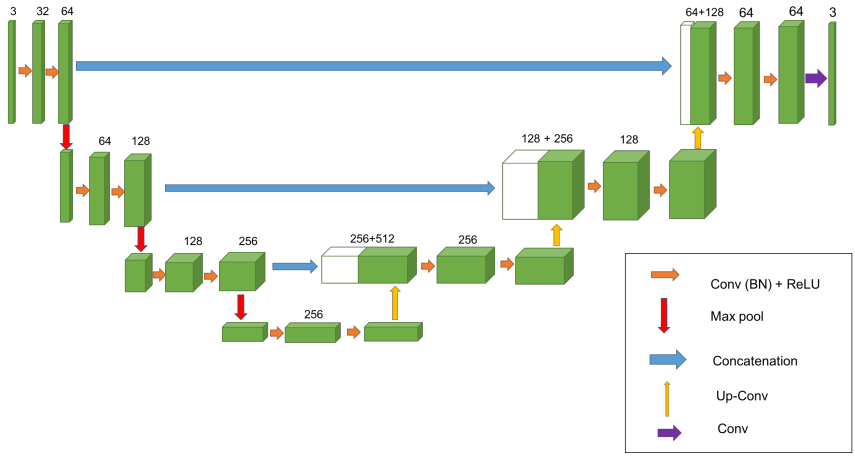


Figure 3.1: *Figure of the architecture of the 3D U-net segmentation network, inspired by Çiçek et al.. Figure by Bardolet Pettersson [41]*

The network used in this thesis was implemented previously by Bardolet Pettersson [41] who collected the network from NiftyNet, an open source platform form medical image analysis [44]. The U-net architecture was chosen since it has been demonstrated to have good performance on a variety of biomedical segmentation tasks [43] [41]. The network is trained with volumes of size 96x96x96 voxels, due to hardware constraints, where eight segments are sampled randomly from each input volume with the same probability of each class being sampled. Two volumes are then chosen from the eight sampled volumes to be used in a batch for training. The U-net network demands all data to have NIfTI file format for training and inference.

The U-net architecture is also used as a generator architecture in the image-to-image translation model Pix2Pix [8] as well as an inspiration to the network architecture in CycleGAN [18].

4

Method

The following chapter describes the data set used in this thesis along with a detailed description of the implemented adversarial networks. The evaluation process of the images produced by the adversarial networks is also described.

4.1 Data set

The data set used in this master thesis is an open source data set from The Lung Image Database Consortium (LIDC) and Image Database Resource Initiative (IDRI) [1] consisting of 1018 cases from clinical thoracic CT scans [25]. When creating the database all scans were evaluated by four radiologists with the task of detecting and outlining lung nodules in the scans [25]. The nodules were outlined so that the pixels of the outer border did not overlap with the nodule pixels [25]. The nodules were classified as

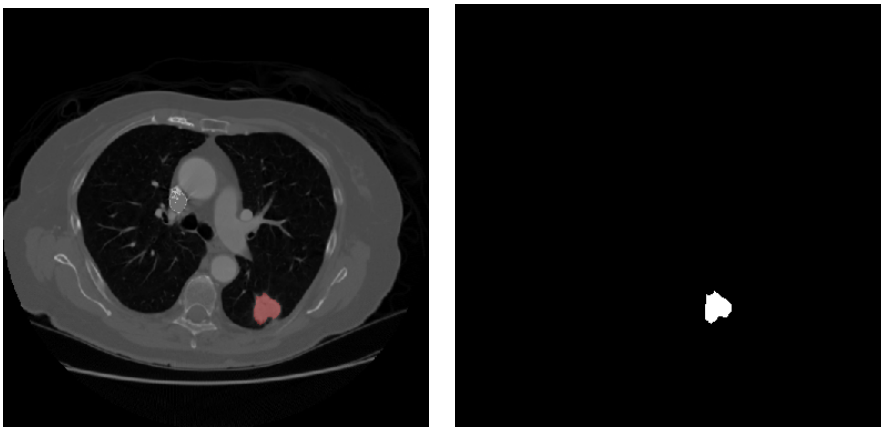
- *nodule* $\geq$ *3 mm*, with an in-plane dimension ranging from 3 mm-30 mm
- *nodule* $<$ *3 mm*, with an in-plane dimension not greater than 3 mm and not clearly benign
- *non-nodule* $\geq$ *3 mm*, any lesion that does not possess any features consistent with a nodule

by pre-decided criteria [25]. The evaluation of the scans was done independently by all four radiologists in two phases [25]. During the first phase, each radiologist categorized and outlined any nodules in all of the scans [25]. During the second phase, each radiologist evaluated all of the scans again with the other radiologists' markings displayed, with the chance to re-evaluate their own marking [25]. All

[1] Available from: https://wiki.cancerimagingarchive.net/display/Public/LIDC-IDRI

markings and categorizations of the nodules were available from the database in the form of XML-files for each scan [25]. Nodules with a size larger than 3 mm has a higher probability of being malignant and therefore, only these nodules are outlined in the data set [25].

The database contains in total 2669 lesions that was marked as *nodule ≥ 3 mm* by at least one radiologist and 928 lesions was marked as *nodule ≥ 3 mm* by all four radiologists [25].



(a) *Example of a CT image with a nodule ≥ 3 mm.*

(b) *Example of a created ground truth image, from an outlined nodule, extracted from an XML-file.*

Figure 4.1: *CT image from the data set with the respective outlined nodule.*

For this thesis, outlined nodules with an agreement of 75 % were used. This means that the nodules were marked by three out of four radiologists. All nodule markings were extracted from the respective XML-file, as mentioned in section 3.2, creating binary ground truth images where white voxels represents nodules and black voxels represent healthy tissue. An example of a CT image and the corresponding ground truth image can be seen to the left and right respectively in Figure 4.1.

4.1.1 Data set split

All pre-processed data were split into three subsets, with one training set, one test set and one validation set. The training set contained 460 volumes, the test set contained 59 volumes and the validation set contained 56 volumes. The training set volumes were used to train the adversarial network and to create new data for the segmentation network. The test set volumes were used to train the segmentation network with a combination of real and synthetic volumes. The validation set was used to evaluate the segmentation network.

All training volumes, of size 128x128x128 voxels, used for the adversarial networks were centered around the nodule in each image to achieve a set volume size. All volumes without any nodules were discarded, resulting in a total of 444 training volumes. To enlarge the two dimensional training set, all slices of the training volumes containing nodules were split into separate two dimensional images resulting in 8156 images used for training the two dimensional adversarial networks. Both test and validation volumes were kept as full volumes, meaning that they were not split up into slices, during use.

4.2 Implementation overview

The following section describes the full chain of steps performed to create the synthetic CT images. A flowchart of all steps can be seen in Figure 4.2.

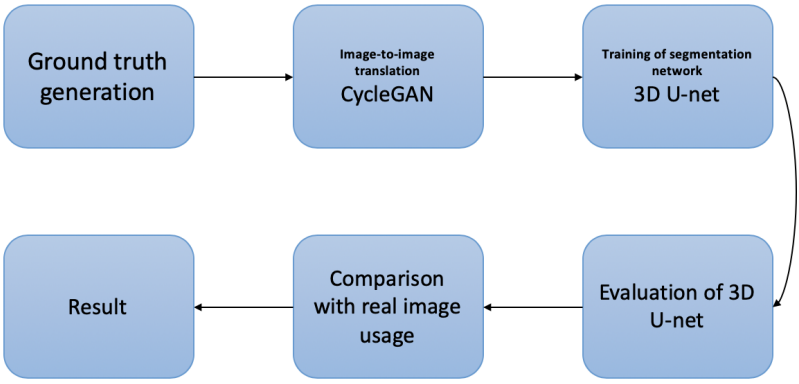


Figure 4.2: *Flowchart of the steps performed to create new synthetic data.*

New ground truth data were first created to ensure that all data created were separated from the training data as well as to ensure that the data were not connected to any real patients. The new ground truth data were then fed to the adversarial network to perform image-to-image translation and generate new synthetic CT images based on the ground truth data. The images were then used to train a 3D segmentation network to evaluate the usability of the generated images. As a final step, the performance of the 3D segmentation network was compared to the performance of the same network trained with real images to produce the result.

4.2.1 Implementation details

For implementation of the networks used in this thesis, the open source software library TensorFlow together with the open source neural network library Keras

were used. Keras was run on top of TensorFlow in Python. The hardware used had the following specifications:

- CPU: Intel Core i7-6700K, 4 cores @ 4.00 Ghz
- GPU: GeForce GTX 1070, 8 GB
- RAM: 32 GB

4.3 Image-to-image translation

Image-to-image translation was performed to map the ground truth images (GT) to their respective CT image. The data set images were first pre-processed to achieve correct normalization as well as to extract more information before feeding them to the GAN. Two different GAN models was explored to perform the image-to-image translation, Pix2Pix and CycleGAN.

Pix2Pix was tested and evaluated first due to the wide use of the model in medical imaging together with good results, as presented in section 3.1.2. Due to lack of good mapping results between the regular GT and CT images with Pix2Pix, the CycleGAN model was also evaluated to explore the impact of the cycle-consistency loss on the image mapping between GT images and CT images. The implementation details of both models are described below. Two different methods for improving the image-to-image translation, by adding more information to the GT images, were tested for both models.

Both the Pix2Pix model and the CycleGAN model showed improved results with label maps as GT images, described further in section 4.3.2. Due to saturation problems in the generator of the Pix2Pix model, the CycleGAN model was eventually chosen for the image-to-image translation.

4.3.1 Adding of noise

As a first try to induce more information to the GT images, noise with a low standard deviation was added. Each new voxel value was calculated according to

$$o[k] = i[k] + \phi \cdot \alpha \tag{4.1}$$

where k is a certain voxel, $i[k]$ is the voxel value of the label map input image, ϕ is a random variable uniformly distributed between 0 and 1 and α is the standard deviation, chosen to 0.05. Unfortunately, this did not improve the performance of the adversarial networks.

4.3.2 Label map creation

As a second try in order to provide the adversarial network with more information, to be able to perform the image-to-image translation, label maps were created using the original CT images. By adding more information to the mapping

between the image domains, the common GAN problem of mode collapse could be prevented and a better mapping was achieved. Mode collapse occur when the generator responsible for the mapping between image domain X and Y only learns one single example of domain Y and then converts every single instance of domains X into that example of domain Y.

The label maps were used to train the network to perform image-to-image translation instead of using the original binary GT images from Figure 4.1. Each label map was created to represent more classes than just nodule tissue and other tissue and hence providing the network with more information to generate correct CT images.

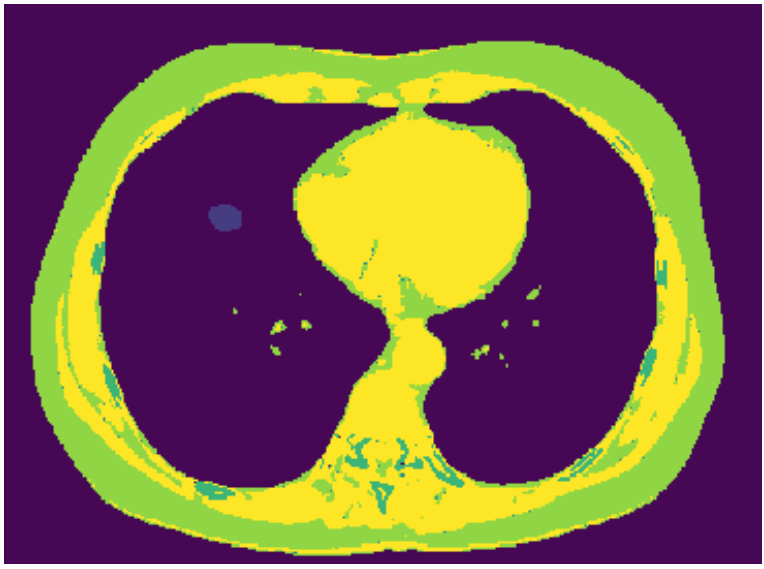


Figure 4.3: *Example of a created label map where the different colors represent different anatomical classes.*

The different classes were created by dividing the Hounsfield values, of each voxels in the CT images, in spans correlating to the tissue the Hounsfield value equates. The different classes represented nodule tissue, water, bone, fat, muscle and soft tissue and other tissues resulting in six different anatomical classes.
The following Hounsfield unit spans were used:

Class	HU span
Water	0
Bone	> 500
Fat	-200 to 0
Muscle tissue and soft tissue	0 to 500
Other tissue	All other

The nodule class was extracted from the ground truth images created from the XML-files in the data set.

This method showed improved results for the image-to-image translation to CT images, resulting in the use of GT images consisting of label maps, seen in Figure 4.3 instead of the previously described regular GT images, seen to the right in Figure 4.1.

4.3.3 Normalization

During the training process of the adversarial network, the input is multiplied with the weights of the network and added together with the biases to cause activations that are then backpropagated with the gradients. To keep the gradients from going out of control, each voxel need to have the same range. All input images are therefore normalized first to have the voxels range between -1 and 1. This is done to scale the input according to the activation functions used in the network, where the range of the voxel values of the images must be the same as the range of the activation function. Since the generator network uses *tanh* as an activation function in the output layer, the images are scaled to range between -1 and 1.

The Hounsfield values of the CT images are naturally centered around 0, with both positive and negative values. All voxel values are also in the range between -1900 and 1900 after the pre-processing described in section 3.2. All CT images were divided by 2000, chosen to ensure that all values would be within the normalization span, to normalize the voxel values between -1 and 1. The label map GT image had 6 different anatomical classes and were therefore divided by 3 and subtracted with 1 to ensure a range between -1 and 1.

4.3.4 Pix2Pix

The Pix2Pix model was implemented following the article by Isola *et al.* [8] with the generator having a U-net architecture, with some modification to fit the size of the data. The generator of the model utilizes skip connections between the encoder and the decoder part of the generator to allow the network to use shortcuts when backpropagating to avoid the problem of vanishing gradients as well as to learn faster. The discriminator utilizes a PatchGAN structure, where the

discriminator evaluates several patches of the image instead of the whole image at once. This is done to produce sharper output images from the generator [8].

ADAM was used as an optimizer for both the generator and discriminator with learning rate $\eta = 0.0002$. The learning rate used was suggested by Isola *et al.* and was not investigated further in this thesis. The batch size was set to 32 during training.

The following layers were used to build the generator and the discriminator:

Layer	**Generator**
1	Convolution-(Filters-64, Kernel size-4, Strides-2), LeakyReLU
2	Convolution-(Filters-128, Kernel size-4, Strides-2, Batch-Normalization), LeakyReLU
3	Convolution-(Filters-256, Kernel size-4, Strides-2), Batch-Normalization), LeakyReLU
4-7	Convolution-(Filters-512, Kernel size-4, Strides-2), Batch-Normalization), LeakyReLU
8	Deconvolution(Filters-512, Kernel size-4, Strides-2), BatchNormalization, Dropout(Rate-0.5), ReLU
9-10	Deconvolution(Filters-1026, Kernel size-4, Strides-2), BatchNormalization, Dropout(Rate-0.5), ReLU
11	Deconvolution(Filters-1026, Kernel size-4, Strides-2), BatchNormalization, ReLU
12	Deconvolution(Filters-512, Kernel size-4, Strides-2), BatchNormalization, ReLU
13	Deconvolution(Filters-256, Kernel size-4, Strides-2), BatchNormalization, ReLU)
13	Deconvolution(Filters-1, Kernel size-4, Strides-1), Tanh

Layer	**Discriminator**
1	Convolution-(Filters-64, Kernel size-4, Strides-2), LeakyReLU
2	Convolution-(Filters-128, Kernel size-4, Strides-2), Batch-Normalization, LeakyReLU
3	Convolution-(Filters-256, Kernel size-4, Strides-2), Batch-Normalization, LeakyReLU
4	Convolution-(Filters-512, Kernel size-4, Strides-2), Batch-Normalization, LeakyReLU
5	Convolution-(Filters-1, Kernel size-4, Strides-1), sigmoid

To train the discriminator, MSE was used as a loss function. Meanwhile for training the generator MSE and MAE was used as loss functions with weight 1 and 100 respectively.

4.3.5 CycleGAN

A modified CycleGAN model was used, matching the overall architecture proposed by Zhu *et al.* [18] where the residual layers and filter sizes were decreased due to hardware constraints. The CycleGAN model also utilizes the PatchGAN structure for the discriminator, as described in section 4.3.4.

ADAM was used an optimizer for both generators and discriminators with the learning rate of $\eta = 0.0002$. The learning rate was kept the same over the first 100 epochs and linearly decaying to 0 during the last 100 epochs. *Beta_1 = 0.5* and *Beta_2 = 0.999* was also used for the Adam optimizer, as proposed by Zhu *et al.* [18]. The batch size was set to 32 during training.

The following layers were used to build the generator and the discriminator:

Layer	Generator
1	Convolution-(Filters-32, Kernel size-7, Strides-1), InstanceNormalization, ReLU
2	Convolution-(Filters-64, Kernel size-3, Strides-2), InstanceNormalization, ReLU
3	Convolution-(Filters-128, Kernel size-3, Strides-2), InstanceNormalization, ReLU
4-9	Residual block - Convolution-(Filters-256, Kernel size-3, Strides-1), InstanceNormalization, ReLU
10	Deconvolution-(Filters-64, Kernel size-3, Strides-2), InstanceNormalization, ReLU
11	Deconvolution-(Filters-32, Kernel size-3, Strides-2), InstanceNormalization, ReLU
12	Convolution-(Filters-1, Kernel size-7, Strides-1), tanh

where the residual block consists of two convolutional layers with the same number of filters in both layers [18]. Activation and normalization were both applied between the convolutional layers and normalization was also applied after the two layers. The original residual block input was merged together with the input passed through the residual block as a final step of the layer. Reflection padding of size 3 was applied before layer 1 and between layer 11 and 12 to reduce artifacts.

Layer	Discriminator
1	Convolution-(Filters-64, Kernel size-4, Strides-2), LeakyReLU
2	Convolution-(Filters-128, Kernel size-4, Strides-2), InstanceNormalization, LeakyReLU
3	Convolution-(Filters-256, Kernel size-4, Strides-2), InstanceNormalization, LeakyReLU
4	Convolution-(Filters-512, Kernel size-4, Strides-2), InstanceNormalization, LeakyReLU
5	Convolution-(Filters-1, Kernel size-4, Strides-1), Sigmoid

To train the discriminators, MSE was used as a loss function. Meanwhile for training the generators MSE was used together with MAE for the cycle-consistency loss. The loss weights 1 was used for the MSE and loss weight 10 was used for the cycle-consistency loss, as proposed by Zhu *et al.* [18].

Even though the training data consisted of paired images, the adversarial network was trained using unpaired data from the training data set. This was done to utilize the full potential of the network architecture as well as to try to prevent the network from memorizing the correct translation of each label map ground truth image.

4.3.6 Conversion to NIfTI

All generated images were converted back to NIfTI file format in order to be usable to train the 3D U-net segmentation network. This was done using the voxel dimension of 0.48828125 x 0.48828125 x 1 mm^3.

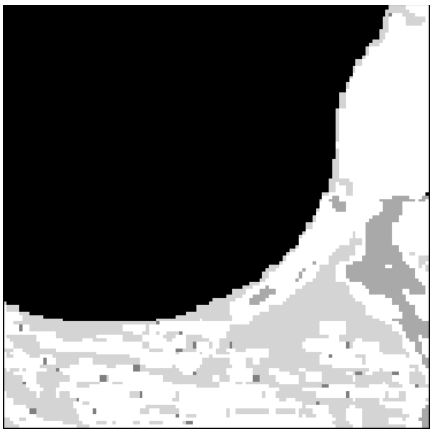
4.4 Volume generation

Due to hardware constraints, only two dimensional images were generated instead of full three dimensional volumes. To generate three dimensional CT volumes, three dimensional ground truth volumes were first split into stacks of two dimensional axial slices. The slices were then fed to the two dimensional adversarial generator one by one and stacked together to re-create the three dimensional volumes.

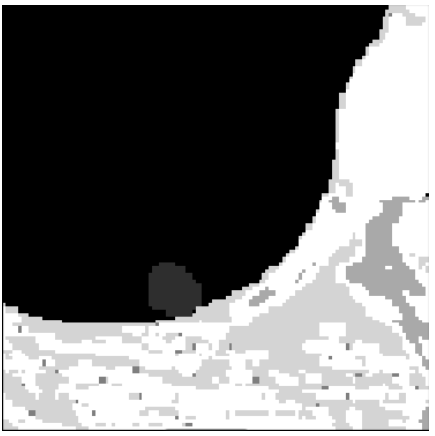
4.5 Creation of data

New label map GT data were created by inserting nodule tissue in random locations onto previously healthy tissue. Nodule templates were first extracted from 15 existing nodules in the training data set, using the binary GT images. As a second step, random cubes of size 128x128x128 were extracted from the training

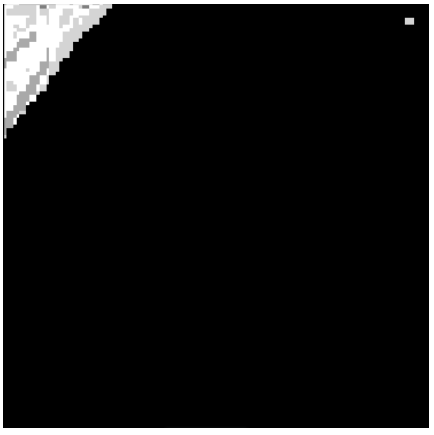
data by first randomly selecting a training volume and then randomly selecting a cube from the chosen training volume. The nodule tissue was then inserted into a random site in the chosen cube with the constraint that 80 % of the nodule had to be positioned on voxels representing *other tissue*, to ensure that the nodule should be positioned inside the lungs. The nodule template inserted was randomly chosen between the 15 nodule templates available.



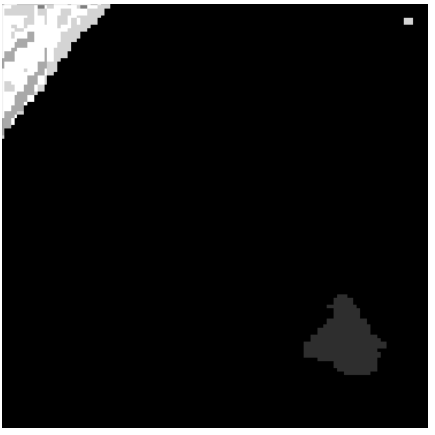
(a) *Example of an original label map GT image.*



(b) *An added nodule onto the original label map GT image.*



(c) *Example of an original label map GT image.*



(d) *An added nodule onto the original label map GT image.*

Figure 4.4: *Original label map GT images and the respective label map GT images with added nodules. All labels are shown in gray scale.*

To create new data sets, the described procedure was iterated a chosen num-

ber of times to create a data set with a set size. To mimic the reality of both volumes with and without nodules, the data sets were created with an 80 % chance of each volume containing a inserted nodule and a 20 % chance of each volume to not contain an inserted nodule.

4.6 Evaluation

The generated synthetic images were evaluated by training a 3D segmentation network, described in section 3.3, with different set ups of synthetic images and evaluating its performance according to sensitivity, accuracy, Dice and F2-score, described in section 2.4. The hyperparameters of the segmentation network were adapted from the thesis work by Bardolet Pettersson [41] and were not evaluated further. The performance result of the segmentation network was compared to the result of the same network trained with real images, performed by Bardolet Pettersson [41].

Three different models of the U-net segmentation network was trained to evaluate the problem statements aimed to be answered in this thesis. The first model was trained with 344 synthetic volumes, the second model was trained with 1400 synthetic volumes and the third model was trained with a combination of 59 real volumes and 590 synthetic volumes. All models were evaluated using 56 real volumes.

The size of the training data set of the first model was chosen to concur with the size of the training data set used by Bardolet Pettersson [41]. This was done to be able to evaluate the difference in performance of the segmentation network when trained with equal amount of synthetic images and real images. The size of the training data set of the second model was chosen to see if the performance of the segmentation network could be increased by a large increase in training data. The size of the training data set used in the third model was chosen to see if the performance of the segmentation network could be increased by using a small amount of real image and large amount of synthetic images in combination.

The data set used to train the third model contained the images used to train the first model and the data set used to train the second model contain both the synthetic images from the first model and the third model. To avoid any bias between the models, all initial weights of the segmentation network was randomly selected for all models.

T-tests were performed comparing the mean of the evaluation parameters of the three models and the model trained with real images. This was done to investigate the significance of the result of each parameter.

5
Results

The following chapter presents the results obtained from the previously described methods, to be able to answer the problem statements presented in the thesis. Since the result of the segmentation network greatly depends on the result of the GAN, visual results from the adversarial networks are shown. Both examples of good image mappings and poor image mappings between label map GT images and CT images are shown. The results from the evaluation of the three different U-net models are presented together with the evaluation of the U-net trained with real images, the comparison model, performed by Bardolet Pettersson [41].

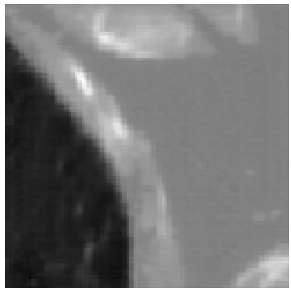
5.1 GAN

The following section presents the result of the image-to-image translation between label map GT images and CT images performed by the CycleGAN. Alongside the generated results, the original CT images are also presented. Both images without any inserted nodules and with inserted nodules are presented.

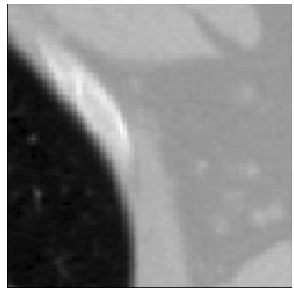
Figure 5.1, 5.2, 5.3, 5.4 and 5.5 shows good mapping results between the label map GT images and the generated CT image, compared to the real CT image.

Figure 5.1 and Figure 5.2 shows two examples of slices from generated volumes without any inserted nodules. Figure 5.3 and Figure 5.4 shows two examples of slices from generated volumes with inserted nodules in the label map GT images. Since the nodules are inserted, there are no nodules present in the real CT images. Figure 5.5 shows an example of a generated volume with an original nodule. This can be seen by the appearance of the nodule in the real CT image.

(a) Example of a label map GT image.

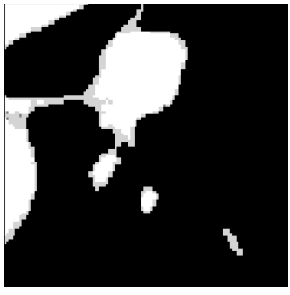


(b) Generated CT image from the adversarial network.



(c) The real CT image corresponding to the label map GT image.

Figure 5.1: *Example of a two dimensional axial slice of a generated volume without a nodule.*

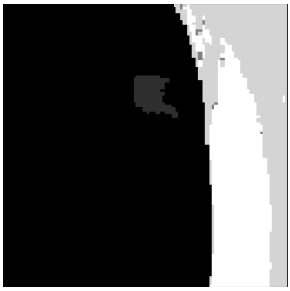


(a) Example of a label map GT image.

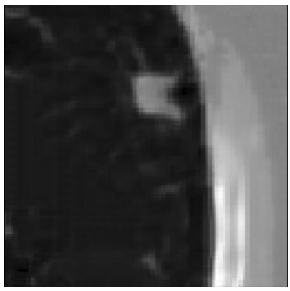
(b) Generated CT image from the adversarial network.

(c) The real CT image corresponding to the label map GT image.

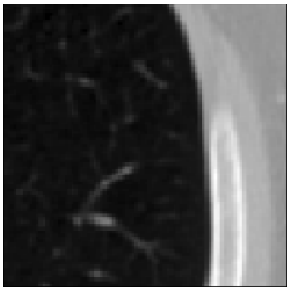
Figure 5.2: *Example of a two dimensional axial slice of a generated volume without a nodule.*



(a) Example of a label map GT image with an inserted nodule.

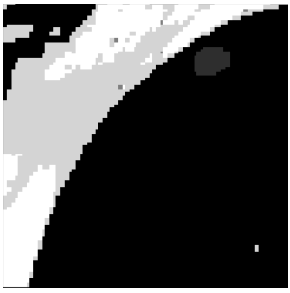


(b) Generated CT image from the adversarial network with an inserted nodule.

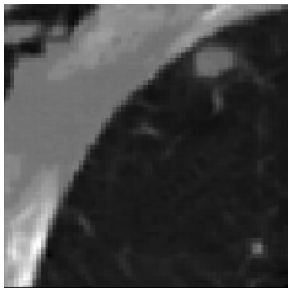


(c) The real CT image corresponding to the original label map GT image without a nodule.

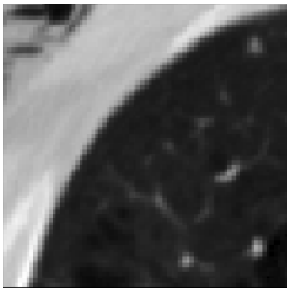
Figure 5.3: *Example of a two dimensional axial slice of a generated volume with an inserted nodule.*



(a) Example of a label map GT image with an inserted nodule.

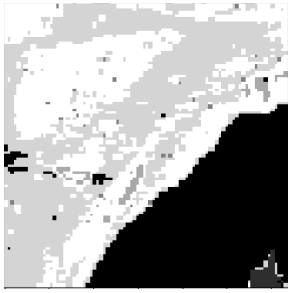


(b) Generated CT image from the adversarial network with an inserted nodule.

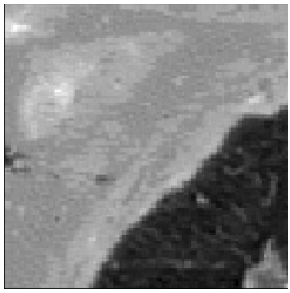


(c) The real CT image corresponding to the label map GT image without a nodule.

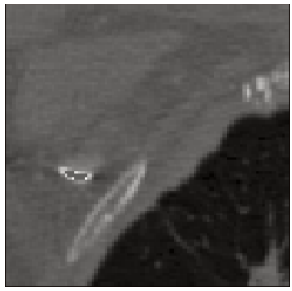
Figure 5.4: *Example of a two dimensional axial slice of a generated volume with an inserted nodule.*



(a) Example of a label map GT image with an original nodule.



(b) Generated CT image from the adversarial network.

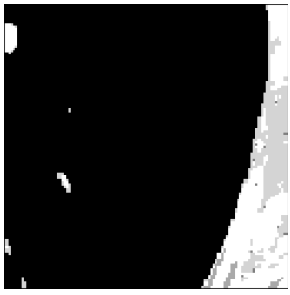


(c) The real CT image corresponding to the label map GT image.

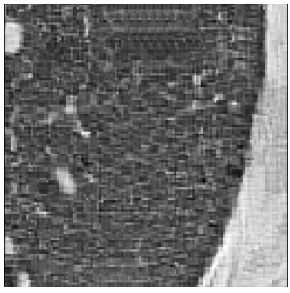
Figure 5.5: *Example of a two dimensional axial slice of a generated volume with an original nodule in the bottom right corner.*

Figure 5.6, 5.7 and 5.8 shows poor mapping results between the label map GT image and the generated CT image, compared to the real CT image.

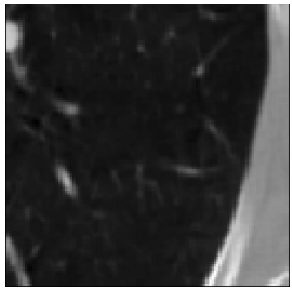
Figure 5.6 shows an example of a very noisy CT generation of the corresponding label map GT image. Figure 5.7 shows an example of the result of a collapsed generator due to the input of a completely black label map GT image. Figure 5.8 shows an example of a generated volume with an inserted nodule that looks unnatural.



(a) Example of a label map GT image.

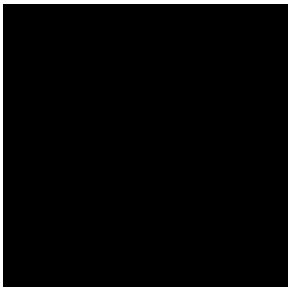


(b) Poorly generated CT image from the adversarial network.

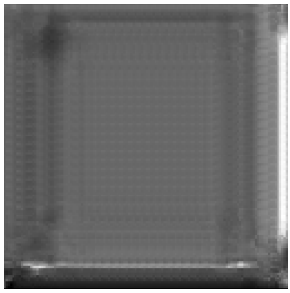


(c) The real CT image corresponding to the label map GT image.

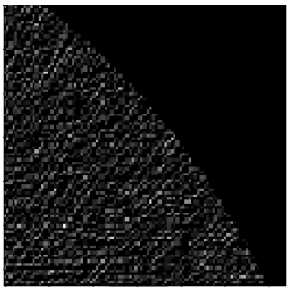
Figure 5.6: *Example of a two dimensional axial slice of a poorly generated CT image, compared to the real CT image.*



(a) Example of a label map GT image.

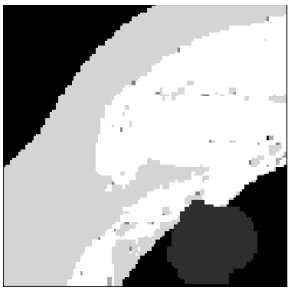


(b) Poorly generated CT image from the adversarial network.

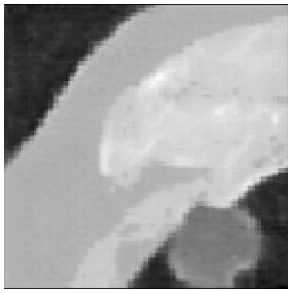


(c) The real CT image corresponding to the label map GT image.

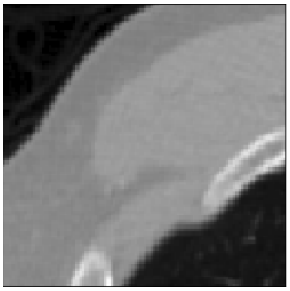
Figure 5.7: *Example of a two dimensional axial slice of a poorly generated CT image, compared to the real CT image.*



(a) Example of a label map GT image.



(b) Poorly generated CT image from the adversarial network.



(c) The real CT image corresponding to the label map GT image.

Figure 5.8: *Example of a generated CT images with an unnatural looking inserted nodule.*

5.2 U-net

The following section presents the results from the evaluation of the three U-net models trained with different combinations of synthetic images or synthetic and real images. The result of the evaluation of the comparison model, trained with only real images, is also presented. The evaluation parameters are presented for each test volume, both with nodules and without nodules. All parameters marked in bold indicates a better result compared to the comparison model. All four models were trained for 100 000 iterations and validated on the same 56 volumes from the validation data set.

The results by Bardolet Petterson [41] were presented after 120 000 training iterations, the network was therefore evaluated at 100 000 iterations to be comparable to the three synthetic models. Due to promising results, the third model was also evaluated at 120 000 iterations for comparison.

The median, mean value and standard deviation are also presented for the Dice score, the sensitivity and the F2-score for each model, excluding the no nodule volumes. The p-value and the 95 % confidence interval are presented for model 1, model 2 and model 3 compared to the comparison model.

Figure 5.9 and table 5.1 presents the mean value of the sensitivity, Dice and F2-score for model 1, model 2, model 3 and the comparison model at 100 000 iterations to give an overview of the result. More detailed results are presented below and in the appendix. From the figure and the table it can be seen that model 3 performs better compared to the comparison model regarding all three measurement parameters. Although, the performance increase is within the error marginals. Meanwhile model 1 and model 2 performs worse than the comparison model in all three cases.

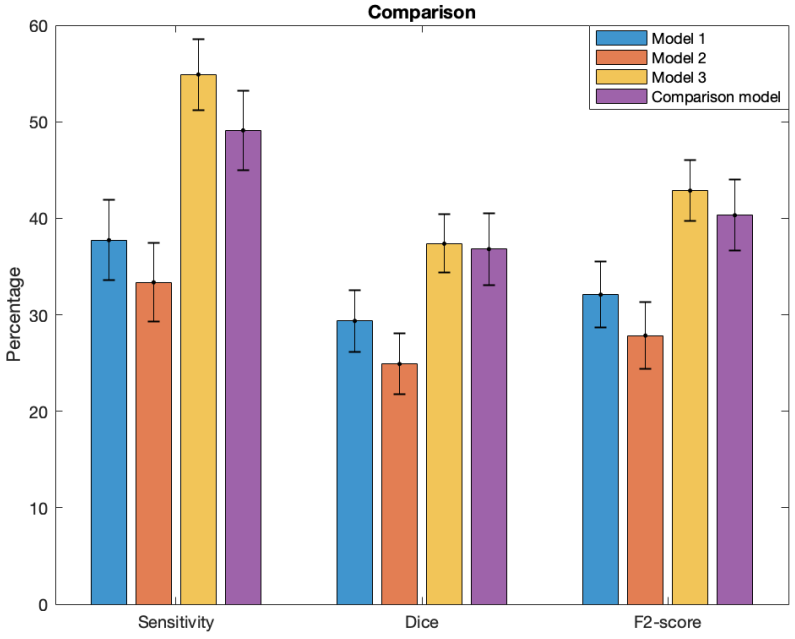


Figure 5.9: *Comparison of the mean value of the sensitivity, Dice and F2-score for all four presented models plotted together with the standard error.*

Measurement	Model 1	Model 2	Model 3	Comparison model
Sensitivity	37.73 %	33.36 %	54.90 %	49.10 %
Dice	29.36 %	24.90 %	37.36 %	36.80 %
F2-score	32.08 %	27.84 %	42.86 %	40.31 %

Table 5.1: *Mean value of the sensitivity, Dice and F2-score for all four presented models.*

5.2.1 Model 1

The following results were obtained from model 1 trained with 344 synthetic CT images and evaluated with the 56 volumes from the validation set. Figures 5.10, 5.11, 5.12 and table 5.2 are based on the result presented in table A.1 in the appendix.

Figures 5.10, 5.11 and 5.12 shows the sensitivity, Dice and F2-score for all patients in the validation images for model 1 compared to the comparison model. It can be seen that model 1 outperforms the comparison model, shown in blue, in some cases but has an overall worse performance.

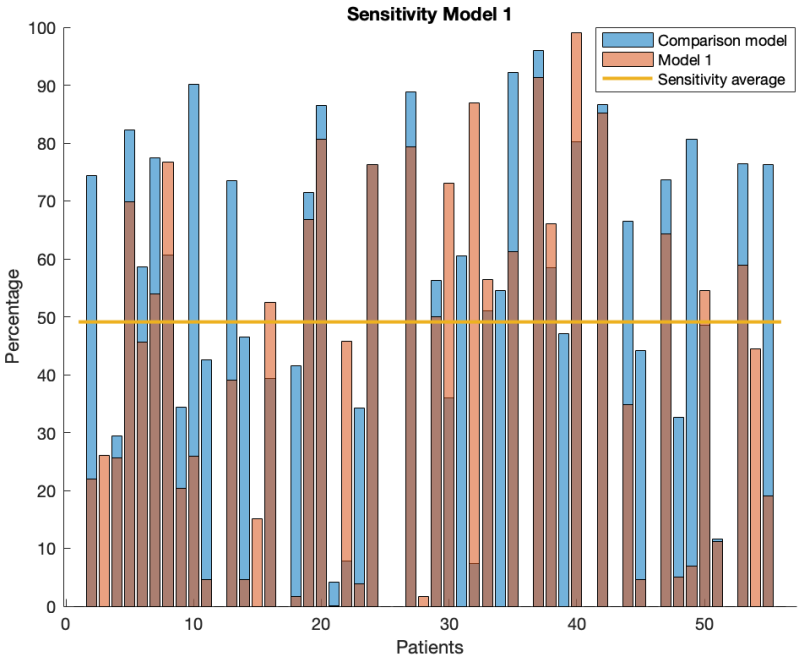


Figure 5.10: *Sensitivity for all validation volumes for model 1, trained with 344 synthetic volumes, compared to the comparison model. All bars are plotted on top of each other to aid the comparison. The mean value of the comparison model is plotted as a yellow line.*

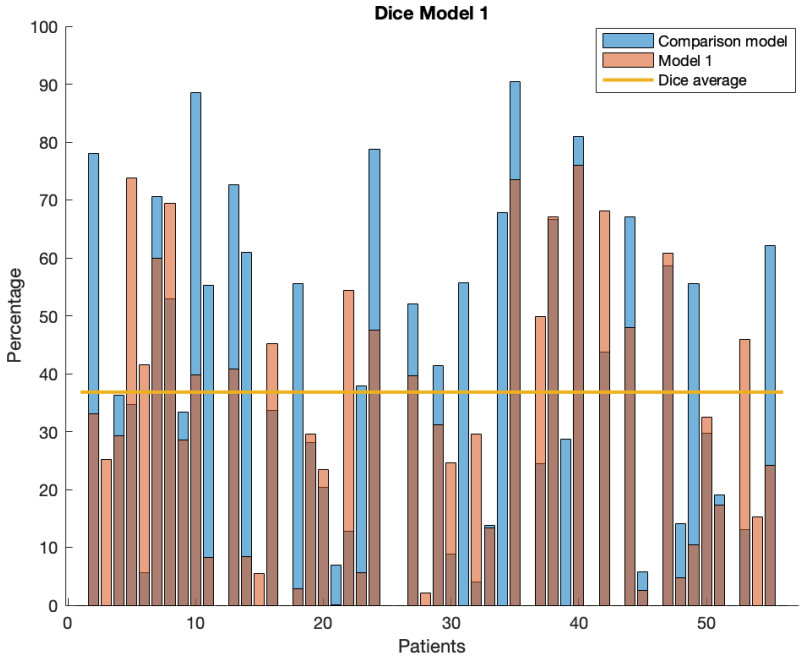


Figure 5.11: *Dice for all validation volumes for model 1, trained with 344 synthetic volumes, compared to the comparison model. All bars are plotted on top of each other to aid the comparison. The mean value of the comparison model is plotted as a yellow line.*

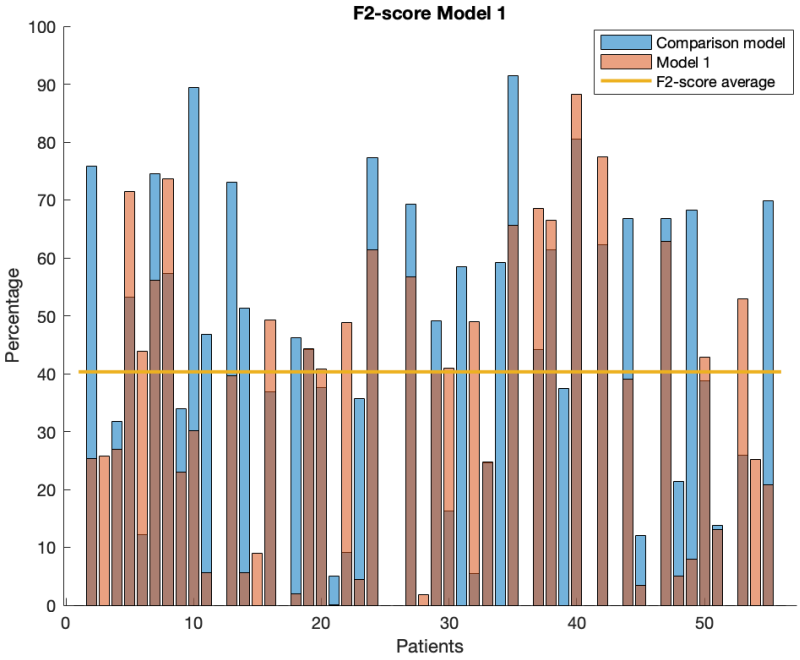


Figure 5.12: *F2-score for all validation volumes for model 1, trained with 344 synthetic volumes, compared to the comparison model. All bars are plotted on top of each other to aid the comparison. The mean value of the comparison model is plotted as a yellow line.*

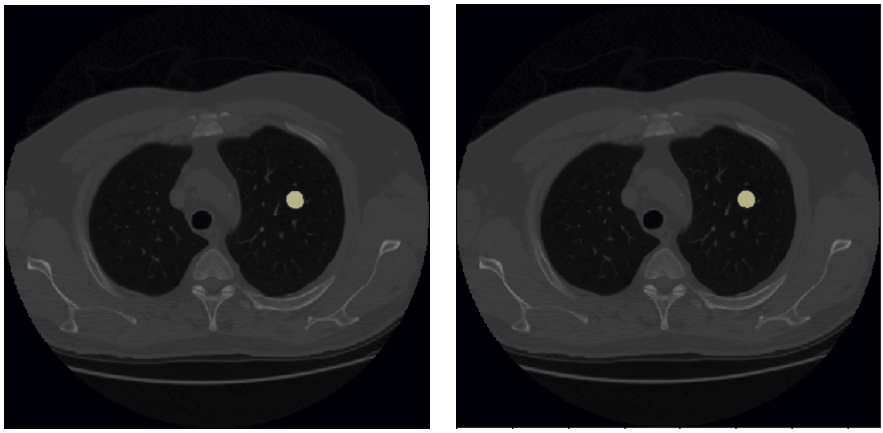
Table 5.2 shows the median, mean value and standard deviation for the sensitivity, Dice and F2-score for model 1. The p-value from the t-test is also presented along with the 95 % confidence interval of the mean value.

Measurement	**Median**	**Mean**	**Standard deviation**	**Confidence interval**	**P-value**
Sensitivity	36.89 %	37.73 %	0.311	29.58-45.88 %	*0.013*
Dice	29.94 %	29.36 %	**0.238**	23.13-35.59 %	0.065
F2-score	28.55 %	32.08 %	**0.257**	25.35-38.81 %	*0.046*

Table 5.2: *Median, mean value and standard deviation calculated for model 1, trained with 344 synthetic volumes, at 100 000 iterations. P-value from t-test between model 1 and the comparison model. All numbers in bold indicates better result compared to the comparison model. All italic values indicate significance.*

Figures 5.13, 5.14, 5.15 and 5.16 shows both good and poor segmentation results for model 1, together with the correct segmentation from the original GT

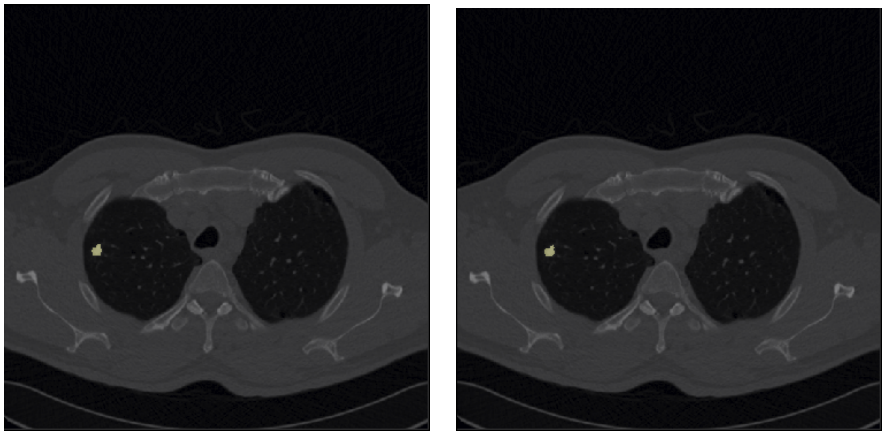
images. Figures 5.13 and 5.14 shows two examples of good segmentation results from model 1 where the network succeeds in segmenting the nodule correct. Figure 5.15 shows an example where model 1 fails to segment the correct nodule. Figure 5.16 shows an example where the model segments voxels that are not correct nodule voxels.



(a) *Segmentation result from model 1 with the nodule marked in yellow.*

(b) *Correct segmentation of the nodule with the nodule marked in yellow.*

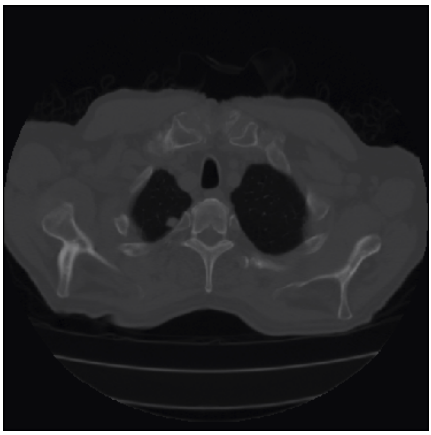
Figure 5.13: *Example of a good segmentation result from model 1 with nodules marked in yellow.*



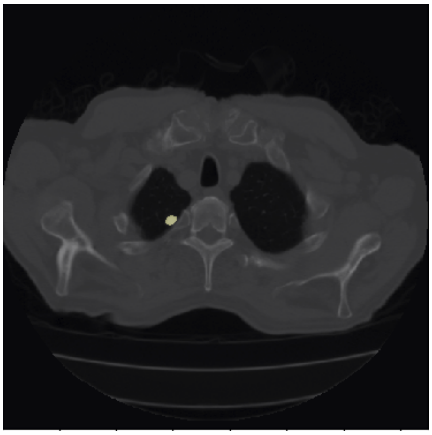
(a) *Segmentation result from model 1 with the nodule marked in yellow.*

(b) *Correct segmentation of the nodule with the nodule marked in yellow.*

Figure 5.14: *Example of a good segmentation result from model 1 with nodules marked in yellow.*



(a) Segmentation result from model 1.



(b) Correct segmentation of the nodule with the nodule marked in yellow.

Figure 5.15: *Example of a poor segmentation result from model 1 with the nodule marked in yellow.*

(a) Segmentation result from model 1 with the nodule marked in yellow.

(b) Correct segmentation of the volume.

Figure 5.16: *Example of a poor segmentation result from model 1 with the segmentation result marked in yellow.*

5.2.2 Model 2

The following results were obtained from model 2, trained with 1400 synthetic CT images and evaluated with the 56 volumes from the validation set. Figures 5.17, 5.18, 5.19 and table 5.3 are based on the result presented in table A.2 in the appendix.

Figures 5.17, 5.18 and 5.19 shows the sensitivity, Dice and F2-score for all patients in the validation images for model 2 compared to the comparison model. It can be seen that model 2 outperforms the comparison model, shown in blue, in some cases but has an overall worse performance.

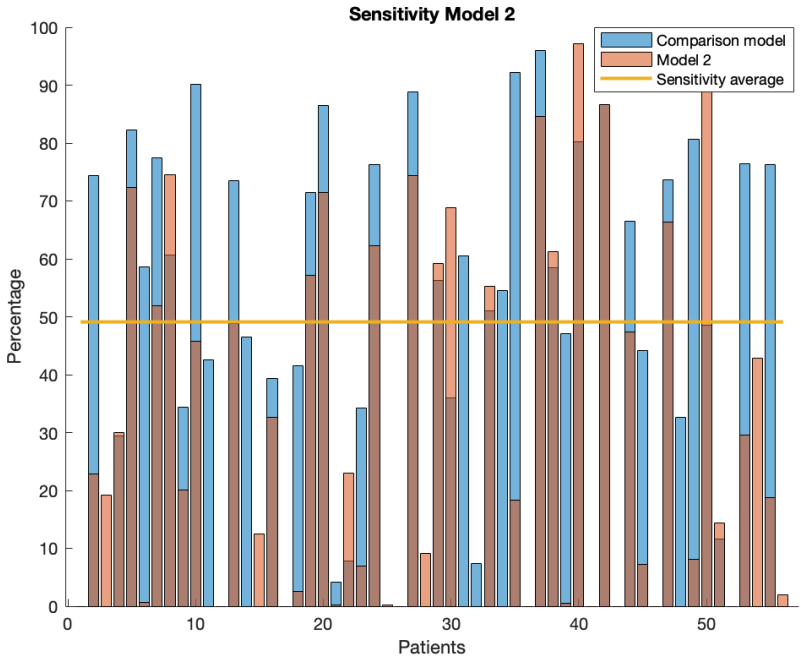


Figure 5.17: *Sensitivity for all validation volumes for model 2, trained with 1400 synthetic volumes, compared to the comparison model. All bars are plotted on top of each other to aid the comparison. The mean value of the comparison model is plotted as a yellow line.*

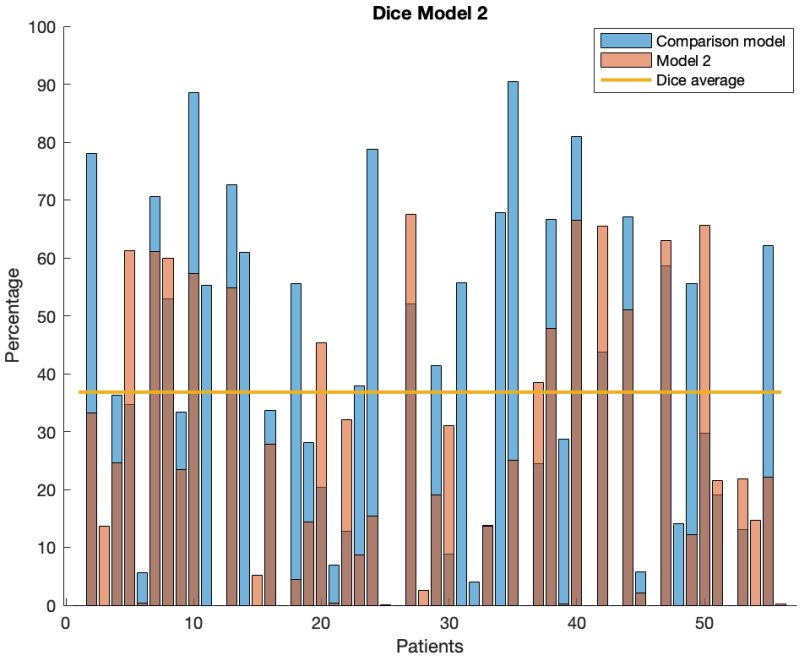


Figure 5.18: *Dice for all validation volumes for model 2, trained with 1400 synthetic volumes, compared to the comparison model. All bars are plotted on top of each other to aid the comparison. The mean value of the comparison model is plotted as a yellow line.*

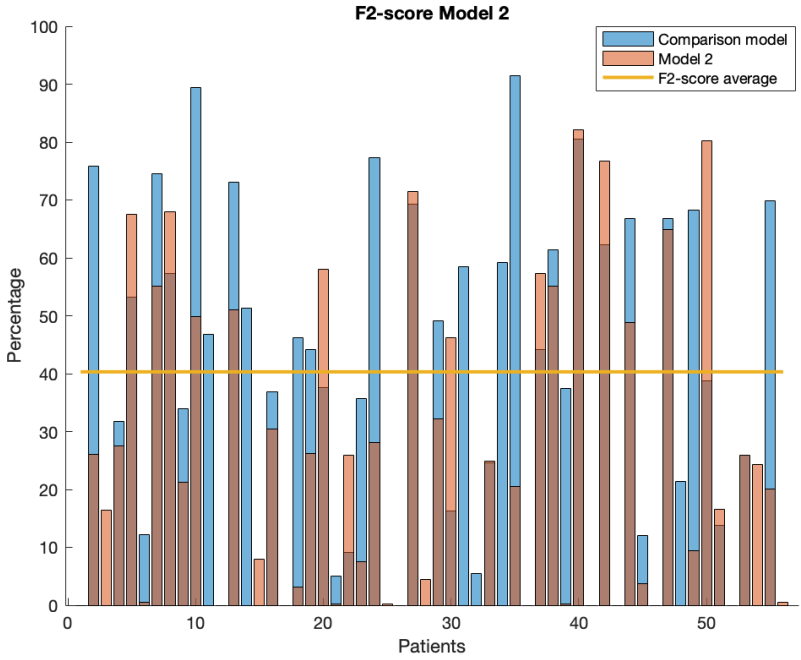


Figure 5.19: *F2-score for all validation volumes for model 2, trained with 1400 synthetic volumes, compared to the comparison model. All bars are plotted on top of each other to aid the comparison. The mean value of the comparison model is plotted as a yellow line.*

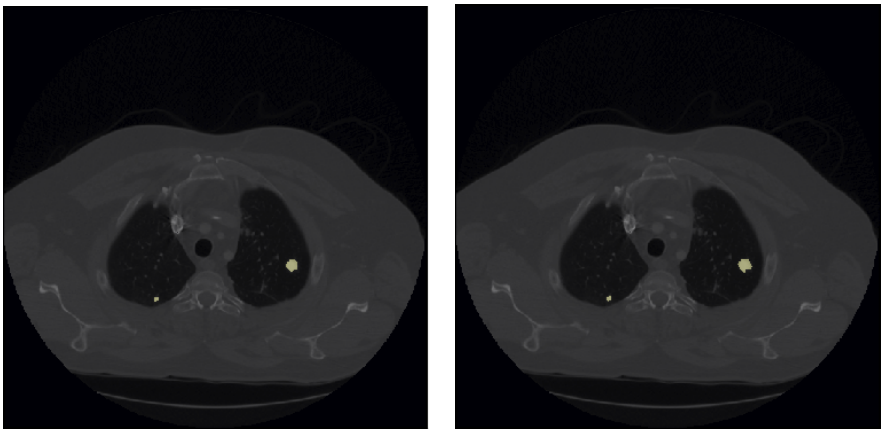
Table 5.3 shows the median, mean value and standard deviation for the sensitivity, Dice and F2-score for model 2. The p-value from the t-test is also presented along with the 95 % confidence interval of the mean value.

Measurement	**Median**	**Mean**	**Standard deviation**	**Confidence interval**	**P-value**
Sensitivity	22.88 %	33.36 %	0.306	25.35-41.37 %	*0.0004*
Dice	20.33 %	24.90 %	**0.234**	18.12-31.68 %	*0.004*
F2-score	24.58 %	27.84 %	**0.259**	21.06-34.62 %	*0.002*

Table 5.3: *Median, mean value and standard deviation calculated for model 2, trained with 1400 synthetic volumes, at 100 000 iterations. P-value from t-test between model 2 and the comparison model. All numbers in bold indicates better result compared to the comparison model. All italic values indicate significance.*

Figures 5.20, 5.21, 5.22 and 5.23 shows both good and poor segmentation results for model 2, together with the correct segmentation from the original GT

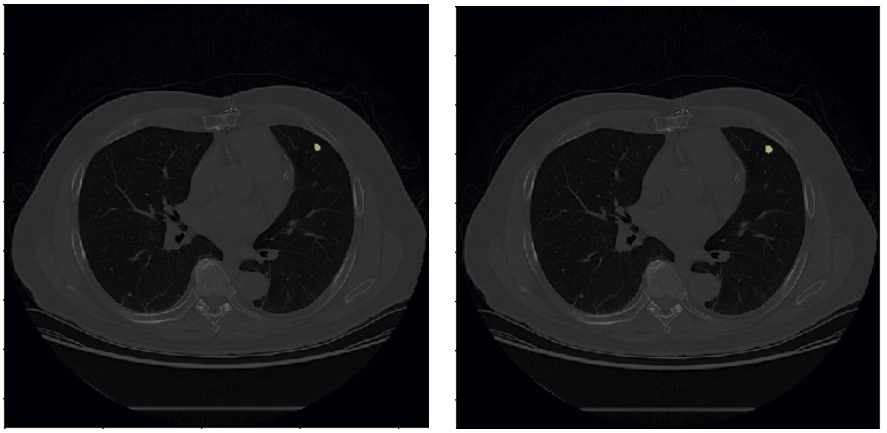
images. Figures 5.20 and 5.21 shows two examples of good segmentation results from model 2 where the network succeeds in segmenting the nodule correct. Figure 5.22 shows an example where model 2 fails to segment the correct nodule. Figure 5.23 shows an example where the model segments voxels that are not correct nodule voxels.



(a) *Segmentation result from model 2 with the nodule marked in yellow.*

(b) *Correct segmentation of the nodule with the nodule marked in yellow.*

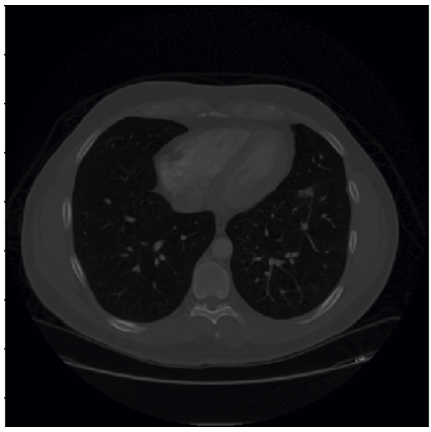
Figure 5.20: *Example of a good segmentation result from model 2 with the nodules marked in yellow.*



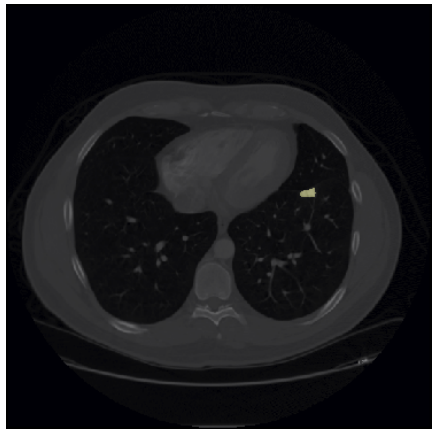
(a) *Segmentation result from model 2 with the nodule marked in yellow.*

(b) *Correct segmentation of the nodule with the nodule marked in yellow.*

Figure 5.21: *Example of a good segmentation result from model 2 with the nodules marked in yellow.*

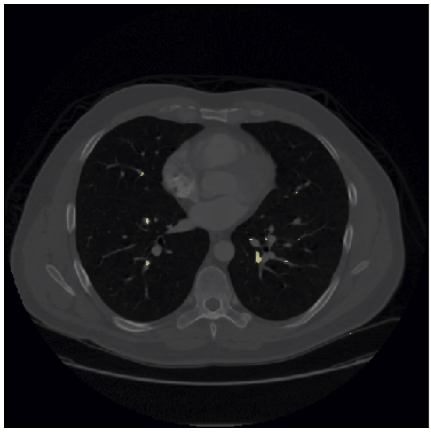


(a) *Segmentation result from model 2.*

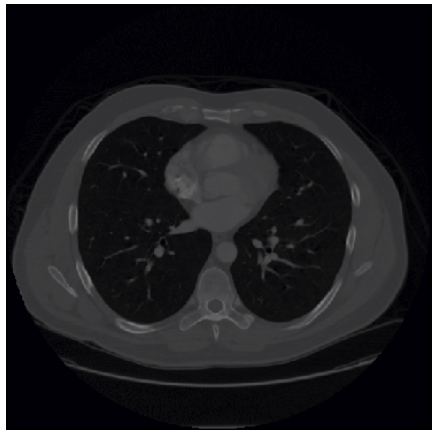


(b) *Correct segmentation of the volume with the nodule marked in yellow.*

Figure 5.22: *Example of a poor segmentation result from model 2 with the nodule marked in yellow.*



(a) *Segmentation result from model 2 with the segmentation result marked in yellow.*



(b) *Correct segmentation of the volume.*

Figure 5.23: *Example of a poor segmentation result from model 2 with the segmentation result marked in yellow.*

5.2.3 Model 3

The following results were obtained from model 3 trained with a combination of 59 real CT images and 590 synthetic CT images and evaluated with the 56 volumes from the validation set. Figures 5.24, 5.25, 5.26 and table 5.4 are based on the result presented in table A.3 in the appendix. Table 5.5 is based on the result presented in table A.4 in the appendix.

Figures 5.24, 5.25 and 5.26 shows the sensitivity, Dice and F2-score for all patients in the validation images for model 3 compared to the comparison model at 100 000 iterations. It can be seen that model 3 outperforms the comparison model, shown in blue, in many cases giving it a better performance overall.

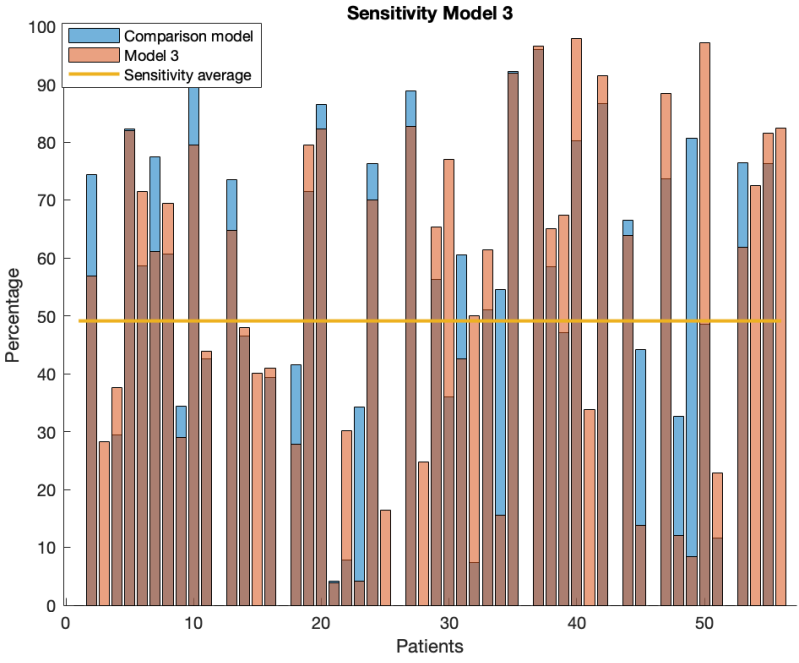


Figure 5.24: *Sensitivity for all validation volumes for model 3, trained with a combination of 59 real volumes and 590 synthetic volumes, compared to the comparison model. All bars are plotted on top of each other to aid the comparison. The mean value of the comparison model is plotted as a yellow line.*

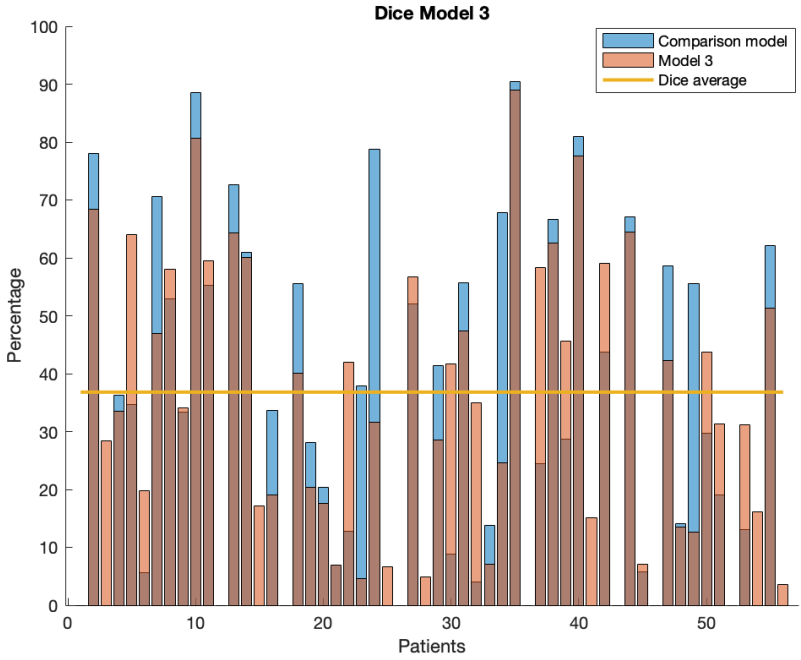


Figure 5.25: *Dice for all validation volumes for model 3, trained with a combination of 59 real volumes and 590 synthetic volumes, compared to the comparison model. All bars are plotted on top of each other to aid the comparison. The mean value of the comparison model is plotted as a yellow line.*

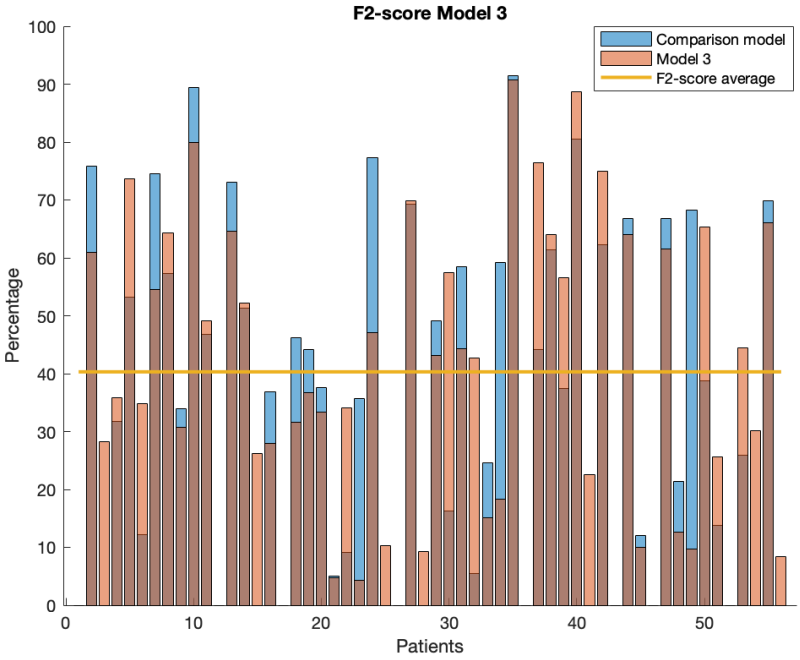


Figure 5.26: *F2-score for all validation volumes for model 3, trained with a combination of 59 real volumes and 590 synthetic volumes, compared to the comparison model. All bars are plotted on top of each other to aid the comparison. The mean value of the comparison model is plotted as a yellow line.*

Table 5.4 shows the median, mean value and standard deviation for the sensitivity, Dice and F2-score for model 3 at 100 000 iterations. The p-value from the t-test is also presented along with the 95 % confidence interval of the mean value.

Measurement	Median	Mean	Standard deviation	Confidence interval	P-value
Sensitivity	**61.61 %**	**54.90 %**	**0.275**	47.70-62.10 %	0.140
Dice	**34.52** %	**37.36** %	**0.226**	31.44-43.28 %	0.838
F2-score	**42.88** %	**42.86** %	**0.236**	36.68-49.04 %	0.384

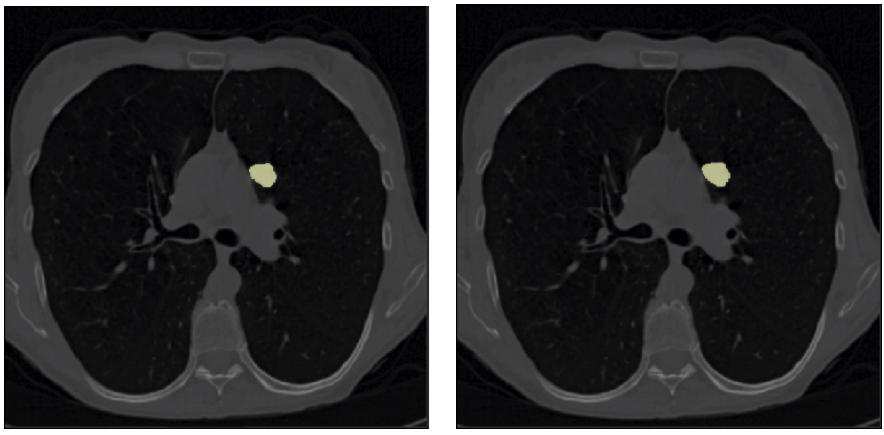
Table 5.4: *Median, mean value and standard deviation calculated for model 3, trained with 59 real volumes and 590 synthetic volumes, at 100 000 iterations. P-value from t-test between model 3 and the comparison model at 100 000 iterations. All numbers in bold indicates better result compared to the comparison model at 100 000 iterations.*

Table 5.5 shows the median, mean value and standard deviation for the sensitivity, Dice and F2-score for model 3 at 120 000 iterations. The p-value from the t-test is also presented along with the 95 % confidence interval of the mean value.

Measurement	Median	Mean	Standard deviation	Confidence interval	P-value
Sensitivity	**57.76 %**	52.44 %	**0.295**	44.71-60.17 %	0.437
Dice	21.75 %	29.34 %	**0.226**	23.42-35.26 %	*0.002*
F2-score	34.00 %	36.20 %	**0.237**	29.99-42.41 %	*0.028*

Table 5.5: *Median, mean value and standard deviation calculated for model 3, trained with 59 real volumes and 590 synthetic volumes, at 120 000 iterations. P-value from t-test between model 3 and the comparison model at 120 000 iterations. All numbers in bold indicates better result compared to the comparison model at 120 000 iterations. All italic values indicate significance.*

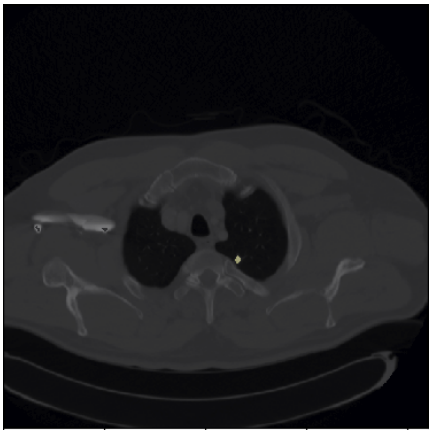
Figures 5.27, 5.28, 5.29 and 5.30 shows both good and poor segmentation results for model 3, together with the correct segmentation from the original GT images. Figures 5.27 and 5.28 shows two examples of good segmentation results from model 3 where the network succeeds in segmenting the nodule correct. Figure 5.29 shows an example where the model segments voxels that are not correct nodule voxels. Figure 5.30 shows an example where model 3 fails to segment the correct nodule.



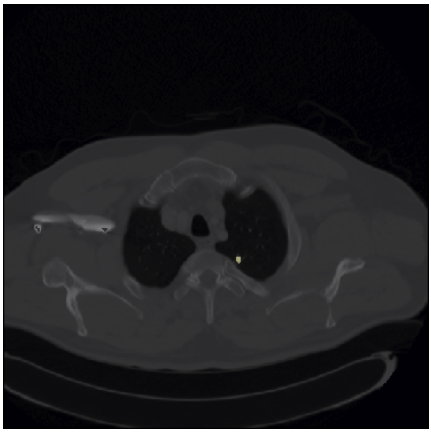
(a) *Segmentation result from model 3 with the nodule marked in yellow.*

(b) *Correct segmentation of the nodule with the nodule marked in yellow.*

Figure 5.27: *Example of a good segmentation result from model 3 with the nodules marked in yellow.*

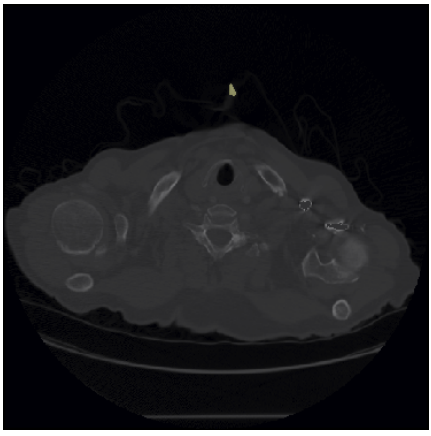


(a) *Segmentation result from model 3 with the nodule marked in yellow.*

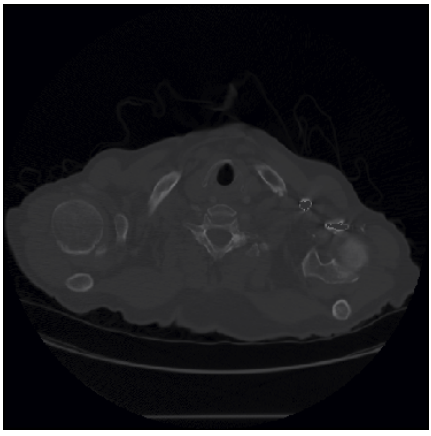


(b) *Correct segmentation of the nodule with the nodule marked in yellow.*

Figure 5.28: *Example of a good segmentation result from model 3 with the segmentation result marked in yellow.*

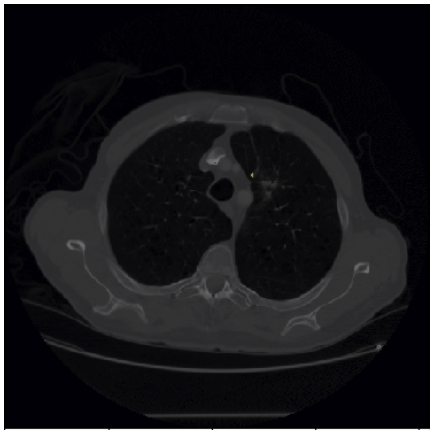


(a) *Segmentation result from model 3 with the segmentation marked in yellow.*

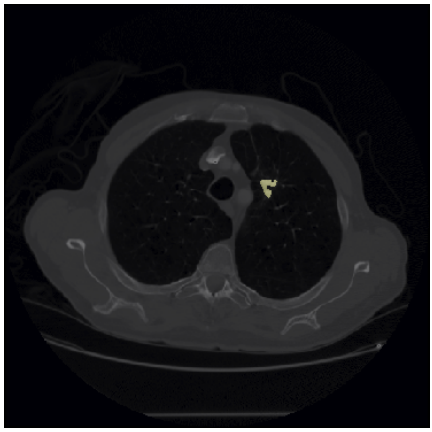


(b) *Correct segmentation of the volume.*

Figure 5.29: *Example of a poor segmentation result from model 3 with the segmentation result marked in yellow.*



(a) Segmentation result from model 3.



(b) Correct segmentation of the volume with the nodule marked in yellow.

Figure 5.30: *Example of a poor segmentation result from model 3 with the nodule marked in yellow.*

5.2.4 Comparison

The following results were obtained from the comparison model trained with 344 real CT images and evaluated with the 56 volumes from the validation set. Table 5.6 is based on the result presented in table A.5 in the appendix and table 5.7 is based on the result presented in table A.6 in the appendix.

Table 5.6 shows the median, mean value and standard deviation for the sensitivity, Dice and F2-score for the comparison model at 100 000 iterations. The p-value from the t-test is also presented along with the 95 % confidence interval of the mean value.

Measurement	Median	Mean	Standard deviation	Confidence interval
Sensitivity	52.76 %	49.10 %	0.305	41.11-57.09 %
Dice	34.18 %	36.80 %	0.278	29.52-44.08 %
F2-score	41.47 %	40.31 %	0.276	33.08-47.54 %

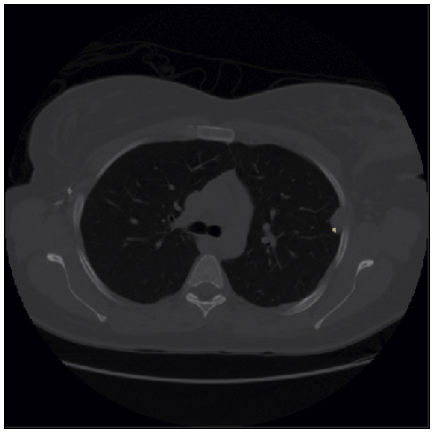
Table 5.6: *Median, mean value and standard deviation calculated for the comparison model, trained with 344 real volumes, at 100 000 iterations.*

Table 5.7 shows the median, mean value and standard deviation for the sensitivity, Dice and F2-score for the comparison model at 120 000 iterations. The p-value from the t-test is also presented along with the 95 % confidence interval of the mean value.

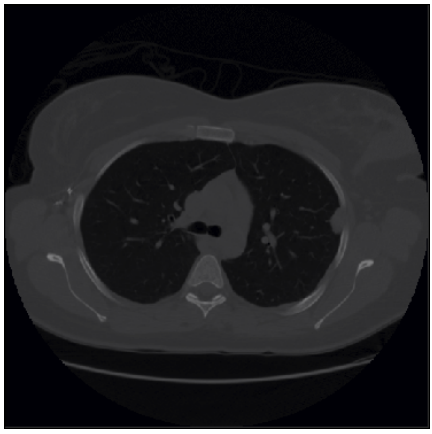
Measurement	Median	Mean	Standard deviation	Confidence interval
Sensitivity	54.77 %	53.09 %	0.312	44.92-61.26 %
Dice	43.14 %	40.65 %	0.285	33.19-48.11 %
F2-score	46.65 %	44.17 %	0.286	36.68-51.66 %

Table 5.7: *Median, mean value and standard deviation calculated for the comparison model, trained with 344 real volumes, at 120 000 iterations.*

Figures 5.31 and 5.32 shows two examples of segmentation results for all models together with the correct segmentation from the original GT images. The results are shown with all models trained at 100 000 iterations for easier comparison.



(a) *Segmentation result from model 1.*



(b) *Segmentation result from model 2.*

(c) *Segmentation result from model 3.*

(d) *Segmentation result from comparison model.*

(e) *Correct segmentation result.*

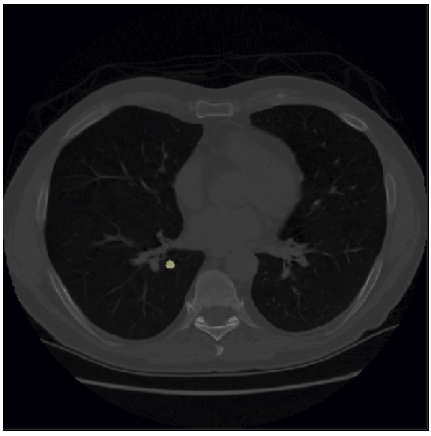
Figure 5.31: *Example of segmentation results from all models with the segmentation results marked in yellow.*

(a) Segmentation result from model 1.

(b) Segmentation result from model 2.

(c) Segmentation result from model 3.

(d) Segmentation result from comparison model.



(e) Correct segmentation result.

Figure 5.32: *Example of segmentation results from all models with the segmentation results marked in yellow.*

6

Discussion

The following chapter presents an analysis and a discussion concerning the results obtained in the previous chapter. A discussion is presented for both the visual GAN result and the evaluation of the four U-net models. The discussion is mainly focused on the GAN result and the result of the U-net models compared to the comparison model. Some discussion points regarding the general performance of the U-net is presented, but not focused on. A short discussion concerning the thesis limitations and future work is also presented.

6.1 GAN results

The synthetic images show a visually good result in general, but with certain errors occurring occasionally. The images can appear somewhat blurry, which can be a side effect when using GAN [8] as well as a result from the small image size. There is also some difference in pixel intensities when comparing the generated images with the real CT images, which can be seen in Figure 5.4 and Figure 5.8. Comparing the intensity of the bright soft tissue between the generated CT images and the real CT images, it can be seen that the intensity can be both brighter and darker than the real CT images. This seems to be a common occurrence between real CT images as well, where the Hounsfield units differ slightly within the tissue causing the difference between images, which can be seen when comparing Figure 5.4 c) and 5.5 c). The difference in the generated images should therefore not make a large difference when using the them. A second common intensity variation in the generated images can be seen in the top left corner of the generated CT image in Figure 5.8, where there are gray variations in the black patch. Using adversarial networks, there are often intensity variations within the same "color" in the output since there is no restriction that the output needs to be a discrete set of values. This is rather used by the CycleGAN to be able to hide

information, like stenography, and to recover it [45].

The goal of the generated images is not necessary to mimic the real images completely, but to represent a possible mapping of the input GT image. Therefore, some dissimilarities are wanted when comparing the generated CT images and the real CT images. Otherwise, it could indicate that the adversarial network has only learned to copy the training images.

By using label map GT images instead of the original binary GT images, the adversarial network has more information about the context of the nodule and can therefore learn and perform a better image domain mapping. This also demands that the images are large enough to include context information. As seen in Figure 5.7, the generator collapses without any label map information. With a too small image size, not enough information is included to facilitate the image-to-image translation. Good mapping is obtained when the label map GT images contain several labels. This could be improved by constraining the randomization, when generating new label map GT volumes, so that it does not contain any volumes with completely black label map slices for example.

The adversarial network is able to generate synthetic images from the generated label map GT volumes. This indicates that the adversarial network has learned to generalize the target mapping of label map GT images to CT images and has not only learned each mapping from the training images. Since all of the training images used for the adversarial network were created from volumes centered around the nodules in the training data set, there is a very small risk that any of the created new volumes would be the same as the ones used in the training images. The created volumes of size 128x128x128 were completely randomized from the training data set, both concerning original volume and sub volume.

By being able to insert nodules into otherwise healthy tissue, completely new CT images are obtained. It also gives the ability to generate specific images and volumes containing nodules and which can then be used to balance the data set regarding nodule tissue compared to healthy tissue. The presented method gives the opportunity to completely extract all nodules available in the data set and combine them with healthy tissue, from the data set, in any arbitrary way. Using this, it is possible to generate a very large amount of data from the original data set. This can be used to create large synthetic data sets with completely new CT data, which is not connected to any real patient.

When selecting the random cubes, the sub volumes, from the original volumes, there are, as for now, no way of determining if the sub volume contains parts located outside of the body. Due to the pre-processing of the data set, the voxels outside of the body was set to the Hounsfield units corresponding to air. This means that in general, the voxels outside of the body receives the same class, *Other tissue*, as the voxels inside the lungs when converting the image to the label map GT images. Therefore, when the sub volume is used to paste nodules

onto the nodules could be located outside of the body. When the sub volume is passed to the generator, the generator treats the voxels outside of the body the same as inside the lungs which can give rise to generated lung tissue formations, the bright strings on the black background seen in Figure 5.1, even outside of the body. This should however not affect the training of the segmentation network since the network only sees cubes of 96x96x96 voxels with no context of the surroundings.

Since all volumes are generated as two dimensional slices and then re-stacked together, there are some variabilities between adjacent slices. The variabilities concern both variations in pixel intensities and in shapes and presence of different lung tissue formations and could affect the result of the segmentation network.

6.2 U-net results

From the results of model 1, by using only the same amount of synthetic training data compared to real training data it is not possible to achieve a better, or equal, performance of the segmentation network. Model 1 outperforms the comparison model in some cases, but has a lower sensitivity, Dice and F2-score in general. The model also fails to detect any nodule tissue in several volumes where nodules are present, decreasing its performance considerably. Model 1 has somewhat lower standard deviation compared to the comparison model, indicating that it delivers more consistent results compared to the comparison model. The comparison model delivers a larger variation in the results, which increases the standard deviation.

Model 2 shows a worse segmentation result compared to both the first model and the comparison model. It outperforms the comparison model in some cases, just as model 1, but has overall lower scores. Just like model 1, it fails to detect nodules in several volumes which decreases the performance considerably. Model 2 contains the same generated volumes as model 1, together with 1056 additional volumes. The decrease in performance between model 1 and 2 could be a result of bad GAN results in the additional volumes. It could also be caused by an overfitting of the network towards synthetic images, due to lack of enough variability in the large training data set. With the current method, only 15 nodule templates are used to create the new label map GT volumes. This could affect the networks ability to generalize different looking nodules negatively. Although, more investigation is needed to be able to establish the cause of the decrease in performance.

Model 3 performs better than the comparison model regarding the sensitivity measure, Dice and F2-score. It also has a lower standard deviation compared to the comparison model, indicating that it is more robust than the comparison model. The result demonstrates that it is possible to increase the performance of the segmentation network by training it with a combination of real and synthetic images, compared to training it with only synthetic images. Model 3 has more

training data in total, compared to the comparison model, which could affect the performance. However, it is hard to predict the impact of the increase in data on the performance of the model, but the result of model 2 indicates that an increase in synthetic training data does not necessarily improve the performance of the model.

Model 1 shows lower results, with significance, regarding both sensitivity and F2-score compared to the comparison model. Model 2 shows lower results, with significance, regarding all evaluation parameters compared to the comparison model. Meanwhile, model 3 does not show any significant results, neither lower nor higher, for any of the evaluation parameters compared to the comparison model. It is possible to see that by using a combination of real and synthetic images, the results are no longer significantly lower compared to the comparison model as when using only synthetic images. This indicates a positive performance trend.

The sensitivity score is the percentage of all nodule tissue voxels the network identifies in the volumes. A high sensitivity score indicates that the model can identify the majority of all nodule voxels present, which is important in this kind of application. The Dice score and the F2-score are measurements of how well the network detects the positive cases. Compared to the sensitivity, the false positives are also included in the calculations of Dice and F2-score. The Dice score rewards true positives while the F2-score rewards true positives and penalizes false negatives. The dice and F2-score are in general lower than the sensitivity for all models, including the comparison model. This is due to that all models detects a large part of false positives, which is penalized in both Dice and F2-score but not regarded at all when calculating the sensitivity. It could be argued that this should not be regarded as a large problem for this kind of application since it is much better for the network to find too many nodule voxels compared to finding too few. When detecting lung nodules, or other malignant tissue, it is important to find all possible or suspicious cases. Therefore, it is better for an application to identify too many cases which later can be dismissed by a clinician, than for an application to find too few with the risk of them being not found at all.

All models have a very high accuracy, which can seem like a very good result at first glance. This is however a result of the imbalance in the data set, where over 98 % of all voxels represent healthy tissue [41]. It is therefore very easy for the network to achieve a high accuracy, by classifying all voxels as healthy tissue. The accuracy is thus a bad measurement of performance of the models. The imbalance in the data set used is also the reason why the specificity measurement is excluded when evaluating the segmentation networks. Due to the imbalance in the data set, the specificity is always very high and does not contribute to the evaluation of the models.

The detection of a large portion of false positives is also the reason for the low Dice score in the models trained with synthetic images, which could also be an

explanation of the higher sensitivity score of the third model. As noted by Bardolet Pettersson [41], the network seems to be biased to nodule tissue. This could be a result caused by the network being trained on volumes with equal probability of containing nodules and healthy tissue. Therefore, nodules have a larger presence in the training data compared to in reality with the consequence of a high false positive detection rate. Improvement of the segmentation network has however not been a part of this master thesis.

An effect of the previously described bias can be seen when comparing the result from model 3 and the comparison model at 100 000 iterations. It is possible to see that model 3 performs better regarding both median and average sensitivity, Dice and F2-score - indicating a better quantitative performance compared to training the network with only real images. However, when running the network training for 120 000 iterations, the performance of model 3 decreases meanwhile the performance of the comparison model increases. The decrease in performance of the third model could be explained by the bias towards nodules, rather than faultiness in the synthetic data. Both the Dice and F2-score decreases considerably from 100 000 iterations to 120 000 iterations due to a large increase in detection of false positives, which points to imperfections in the network architecture, as mentioned by Bardolet Petterson [41]. It could also be an effect of that model 3 has converged at 100 000 iterations, giving the decrease in performance. The increase in the performance of the comparison model with increasing iterations could indicate that the network has not yet converged.

Even though the performance of model 1 and model 2 are significantly lower than the comparison model for almost all evaluation parameters, the performance is still not worthless. It could be argued that the synthetic images still carry some vital information for the segmentation of nodule in thoracic CT images, otherwise the performance of both models should be worse. All synthetic models, trained with only synthetic images or a combination of synthetic and real images, outperforms the comparison model in some validation cases. This also implies that the synthetic images contain significant information for the segmentation network to learn the intended task. Based on this study, it is not possible to only use synthetic training images to achieve equal performance as the same network trained with only real images.

Since the performance of model 3 is neither significantly better nor worse than the performance of the comparison model at 100 000 iterations, there is an indication that it could be possible to achieve equal or almost equal performance with the use of a combination of real images and synthetic images where there is an abundance of synthetic images. Model 3 uses six times fewer real images compared to the comparison model with equal results. Even though the model has more training data in total, the result from model 2 implicates that the increase in synthetic training data does not consequently have a positive impact on the performance. The result would then imply that an equal performance of the segmentation algorithm could be achieved with a combination of fewer real images

and a large amount of synthetic images or the performance could be boosted by the addition of synthetic images - created at close to no cost.

6.3 Limitations

One of the main challenge in this thesis work has been to achieve a good image-to-image translation and providing the GAN with enough information to perform the image mapping. Different methods were tested to achieve enough information, resulting in the use of label map GT images. Since all label map GT images are quantifications of the corresponding CT images, it limited the creation of new GT data. In an ideal scenario it would be possible to randomly create completely new label map GT images, to ensure completely new data. In the current method, all label map GT images are randomly selected from the current data.

A lot of focus was put on the Pix2Pix GAN architecture, since it seemed most promising for the task, with mixed results. It took some time before the CycleGAN architecture was explored, which in itself was somewhat time consuming. The ultimate choice fell eventually on CycleGAN due to saturation problems with the Pix2Pix architecture which could not be solved. This limited the time that could be spent on improving the CycleGAN mapping and training, which could have resulted in a better image-to-image translation.

A large limitation in this thesis work has been hardware connected limitations. It was originally intended to perform the image-to-image translation using 3D GAN networks, instead of the 2D CycleGAN used. Due to limited GPU memory, it was not possible to use CycleGAN in 3D. The GPU memory also constrained the image size used, where the CycleGAN network architecture had to be slightly modified to be able to process images of size 128x128.

During this thesis, three U-net models were trained, validated and evaluated with a total time of 40 hours each. The CycleGAN model took on average 30 hours to train and 10 minutes to generate the images. This limited the tuning of parameters and iteration lengths for all networks. It would have been interesting to be able to iterate all U-net models further to see if the performance would change.

6.4 Future work

The use of synthetic images for training machine learning algorithms still needs more investigation and fine tuning. However, the usage of GAN to perform image-to-image translation seems promising. More investigation is also needed in order to establish why the performance decreases between model 1 and model 2 together with the impact of increased amount training data. This could be done by studying the quality of the data and by training several models with a small increase in training data between the models. To be able to establish why the performance decreases with increasing iterations in model 3, the convergence of

the network could be studied.

To fully be able to investigate the impact of combining real and synthetic data, it would be interesting to compare the performance of the comparison model, trained with 344 real volumes, with the performance of a model trained with 344 real volumes and an abundance of synthetic images. It would also be interesting to increase the amount of synthetic images gradually, to investigate the impact of the synthetic data as well as comparing the model trained with 344 real images with a model trained with a combination of real and synthetic images with 344 images in total. Unfortunately, there was no time for this during this thesis work.

To further investigate the problem formulations, it would be interesting to evaluate the influence of the 2D generation of the volumes, compared to 3D generation, on the performance of the segmentation results of the U-net as well as to investigate the result on a different type of data set. As well as to evaluate the result of the image-to-image translation with CycleGAN using full size images from the data set used in this thesis, instead of using sub volumes of size 128x128x128.

7

Conclusion

This thesis shows that it was possible to generate synthetic thoracic CT images using Generative Adversarial Networks. However, it was not possible to achieve an equal quantitative performance of a segmentation network trained with synthetic images compared to a segmentation network trained with the same amount of real images in the scope of this thesis.

An increase in synthetic training images did not increase the quantitative performance of a segmentation network, which implies that a network trained solely on synthetic images could not outperform a network trained only on real images.

It was possible achieve equal quantitative performance of a segmentation network, as a segmentation network trained with only real images, by training it with a combination of real and synthetic images, where a majority of the images were synthetic images and a minority were real images. However, it was not possible to increase the quantitative performance compared to a segmentation network trained with only real images. No significant difference was found between a segmentation network trained with a combination of 59 real images and 590 synthetic images and a segmentation network trained with 344 real images regarding sensitivity, Dice and F2-score. By optimizing GAN and the U-net, it might be possible to achieve a better quantitative performance.

Equal quantitative performance of a segmentation network could thus be achieved by using fewer real images together with an abundance of synthetic images, created at close to no cost, indicating a usefulness of synthetically generated images.

Appendix

A

Detailed results

A.1 Model 1

Table A.1: *Detailed results for all validation volumes for model 1. All numbers marked in bold indicates better result compared to the comparison model.*

Patient	Sensitivity	Accuracy	Dice	F2-score	Nodule present
1	0 %	99.99 %	0 %	0 %	No
2	21.92 %	99.99 %	33.12 %	25.35 %	Yes
3	**26.12 %**	99.99 %	**25.16 %**	**25.71 %**	Yes
4	25.56 %	99.99 %	29.29 %	26.94 %	Yes
5	69.84 %	99.99 %	**73.84 %**	**71.39 %**	Yes
6	45.66 %	99.99 %	**41.60 %**	**43.94 %**	Yes
7	53.89 %	99.99 %	59.88 %	56.13 %	Yes
8	**76.72 %**	99.99 %	**69.42 %**	**73.62 %**	Yes
9	20.33 %	99.99 %	28.59 %	22.98 %	Yes
10	25.96 %	99.98 %	39.85 %	30.16 %	Yes
11	4.63 %	99.99 %	8.29 %	5.62 %	Yes
12	0 %	99.99 %	0 %	0%	No
13	39.02 %	99.99 %	40.74 %	39.69 %	Yes
14	4.55 %	99.99 %	8.38 %	5.57 %	Yes
15	**15.18 %**	99.99 %	**5.50 %**	**8.91 %**	Yes
16	**52.42 %**	99.99 %	**45.19 %**	**49.27 %**	Yes
17	0 %	99.99 %	0 %	0 %	No
18	1.64 %	99.99 %	2.82 %	1.97 %	Yes
19	66.82 %	99.99 %	**29.50 %**	**44.36 %**	Yes

20	80.60 %	99.99 %	23.46 %	40.83 %	Yes
21	0.06 %	99.99 %	0.10 %	0.07 %	Yes
22	**45.77 %**	99.99 %	**54.45 %**	**48.89 %**	Yes
23	3.89 %	99.99 %	5.62 %	4.44 %	Yes
24	76.26 %	99.99 %	47.45 %	61.36 %	Yes
25	0 %	99.99 %	0 %	0 %	Yes
26	0 %	99.99 %	0 %	0 %	No
27	79.31 %	99.99 %	39.69 %	56.67 %	Yes
28	**1.74 %**	99.99 %	**2.05 %**	**1.85 %**	Yes
29	50.00 %	99.99 %	31.16 %	40.26 %	Yes
30	**73.10 %**	99.99 %	**24.63 %**	**40.90 %**	Yes
31	0 %	99.99 %	0 %	0 %	Yes
32	**86.94 %**	99.99 %	**29.58 %**	**48.96 %**	Yes
33	**56.44 %**	99.99 %	13.39 %	**24.69 %**	Yes
34	0 %	99.99 %	0 %	0%	Yes
35	61.18 %	99.99 %	73.56 %	65.60 %	Yes
36	0 %	99.99 %	0 %	0 %	No
37	91.29 %	99.99 %	**49.91 %**	**68.55 %**	Yes
38	**66.06 %**	99.99 %	**67.05 %**	**66.45 %**	Yes
39	0 %	99.99 %	0 %	0 %	Yes
40	**99.08 %**	99.99 %	75.93 %	**88.31 %**	Yes
41	0 %	99.99 %	0 %	0 %	Yes
42	85.13 %	99.99 %	**68.10 %**	**77.39 %**	Yes
43	0 %	99.99 %	0 %	0 %	No
44	34.75 %	99.99 %	48.01 %	39.06 %	Yes
45	4.55 %	99.99 %	2.59 %	3.49 %	Yes
46	0 %	99.99 %	0 %	0 %	No
47	64.30 %	99.99 %	**60.84 %**	62.87 %	Yes
48	5.09 %	99.99 %	4.81 %	4.97 %	Yes
49	6.92 %	99.99 %	10.43 %	7.99 %	Yes
50	**54.55 %**	99.99 %	**32.45 %**	**42.87 %**	Yes
51	11.17 %	99.98 %	17.35 %	13.03 %	Yes
52	0 %	99.99 %	0 %	0 %	No
53	58.96 %	99.99 %	**45.91 %**	**52.94 %**	Yes
54	**44.43 %**	99.99 %	**15.26 %**	**25.18 %**	Yes
55	19.02 %	99.99 %	24.23 %	20.81 %	Yes
56	0 %	99.99 %	0 %	0 %	Yes

A.2 Model 2

Table A.2: *Detailed results for all validation volumes for model 2. All numbers marked in bold indicates better result compared to the comparison model.*

Patient	Sensitivity	Accuracy	Dice	F2-score	Nodule present
1	0 %	99.99 %	0 %	0 %	No
2	22.79 %	99.99 %	33.16 %	26.05 %	Yes
3	**19.14 %**	99.99 %	**13.66 %**	**16.49 %**	Yes
4	**30.05 %**	99.99 %	24.54 %	27.57 %	Yes
5	72.28 %	99.99 %	**61.30 %**	**67.45 %**	Yes
6	0.68 %	99.99 %	0.30 %	0.45 %	Yes
7	51.83 %	99.99 %	61.03 %	55.16 %	Yes
8	**74.51 %**	99.99 %	**59.91 %**	**67.90 %**	Yes
9	20.02 %	99.99 %	23.36 %	21.24 %	Yes
10	45.77 %	99.98 %	57.36 %	49.79 %	Yes
11	0 %	99.99 %	0 %	0 %	Yes
12	0 %	99.99 %	0 %	0 %	No
13	48.88 %	99.99 %	54.80 %	51.09 %	Yes
14	0 %	99.99 %	0 %	0 %	Yes
15	**12.52 %**	99.99 %	**5.15 %**	**7.96 %**	Yes
16	32.62 %	99.99 %	27.80 %	30.51 %	Yes
17	0 %	99.99 %	0 %	0 %	No
18	2.58 %	99.99 %	4.41 %	3.10 %	Yes
19	57.18 %	99.99 %	14.41 %	26.14 %	Yes
20	71.45 %	99.99 %	**45.39 %**	**58.10 %**	Yes
21	0.25 %	99.99 %	0.32 %	0.27 %	Yes
22	**22.97 %**	99.98 %	**32.10 %**	**25.92 %**	Yes
23	6.87 %	99.99 %	8.75 %	7.51 %	Yes
24	62.30 %	99.99 %	15.37 %	28.04 %	Yes
25	**0.16 %**	99.99 %	**0.14 %**	**0.15 %**	Yes
26	0 %	99.99 %	0 %	0 %	No
27	74.32 %	99.99 %	**67.59 %**	**71.47 %**	Yes
28	**9.13 %**	99.99 %	**2.50 %**	**4.43 %**	Yes
29	**59.20 %**	99.99 %	19.12 %	32.20 %	Yes
30	**68.76 %**	99.99 %	**31.02 %**	**46.25 %**	Yes
31	0 %	99.99 %	0 %	0 %	Yes
32	0 %	99.99 %	0 %	0 %	Yes
33	**55.30 %**	99.99 %	13.65 %	**24.92 %**	Yes
34	0 %	99.99 %	0 %	0 %	Yes
35	18.27 %	99.99 %	25.09 %	20.50 %	Yes
36	0 %	99.99 %	0 %	0 %	No
37	84.65 %	99.99 %	**38.54 %**	**57.25 %**	Yes
38	**61.29 %**	99.99 %	47.82 %	55.08 %	Yes
39	0.52 %	99.99 %	0.17 %	0.28 %	Yes
40	**97.22 %**	99.99 %	66.52 %	**82.07 %**	Yes
41	0 %	99.99 %	0 %	0 %	Yes
42	86.64 %	99.99 %	**65.52 %**	**76.74 %**	Yes
43	0 %	99.99 %	0 %	0 %	No

44	47.42 %	99.98 %	51.03 %	48.80 %	Yes
45	7.27 %	99.99 %	2.14 %	3.72 %	Yes
46	0 %	99.99 %	0 %	0 %	No
47	66.29 %	99.99 %	**63.06 %**	64.96 %	Yes
48	0 %	99.99 %	0 %	0 %	Yes
49	8.12 %	99.99 %	12.15 %	9.36 %	Yes
50	**94.18 %**	99.99 %	**65.67 %**	**80.24 %**	Yes
51	**14.43 %**	99.98 %	**21.54 %**	**16.62 %**	Yes
52	0 %	99.99 %	0 %	0 %	No
53	29.57 %	99.99 %	**21.84 %**	25.90 %	Yes
54	**42.86 %**	99.99 %	**14.67 %**	**24.24 %**	Yes
55	18.83 %	99.99 %	22.07 %	20.01 %	Yes
56	**2.01 %**	99.99 %	**0.24 %**	**0.51 %**	Yes

A.3 Model 3

A.3.1 100 000 iterations

Table A.3: *Detailed results for all validation volumes for model 3 at 100 000 iterations. All numbers marked in bold indicates better result compared to the comparison model at 100 000 iterations.*

Patient	Sensitivity	Accuracy	Dice	F2-score	Nodule present
1	0 %	99.99 %	0 %	0 %	No
2	56.91 %	99.99 %	68.35 %	60.99 %	Yes
3	**28.26 %**	99.99 %	**28.34 %**	**28.31 %**	Yes
4	**37.58 %**	99.99 %	33.44 %	**35.81 %**	Yes
5	82.03 %	99.99 %	**63.95 %**	**73.69 %**	Yes
6	**71.43 %**	99.99 %	**19.73 %**	**34.88 %**	Yes
7	61.13 %	99.99 %	46.87 %	54.50 %	Yes
8	**69.39 %**	99.99 %	**57.99 %**	**64.33 %**	Yes
9	28.97 %	99.99 %	**34.04 %**	30.80 %	Yes
10	79.45 %	99.99 %	80.66 %	79.93 %	Yes
11	**43.94 %**	99.99 %	**59.49 %**	**49.07 %**	Yes
12	0 %	99.99 %	0 %	0 %	No
13	64.74 %	99.99 %	64.25 %	64.55 %	Yes
14	**48.02 %**	99.99 %	60.04 %	**52.20 %**	Yes
15	**40.14 %**	99.99 %	**17.21 %**	**26.18 %**	Yes
16	**40.89 %**	99.99 %	19.02 %	28.01 %	Yes
17	0 %	99.99 %	0 %	0 %	No
18	27.79 %	99.99 %	40.06 %	31.67 %	Yes
19	**79.47 %**	99.99 %	20.30 %	36.69 %	Yes
20	82.25 %	99.99 %	17.60 %	33.32 %	Yes
21	3.89 %	99.99 %	**6.95 %**	4.72 %	Yes

22	**30.21 %**	99.99 %	**42.04 %**	**34.04 %**	Yes
23	4.14 %	99.99 %	4.53 %	4.29 %	Yes
24	69.93 %	99.99 %	31.64 %	47.12 %	Yes
25	**16.40 %**	99.99 %	**6.68 %**	**10.36 %**	Yes
26	0 %	99.99 %	0 %	0 %	No
27	82.77 %	99.99 %	**56.70 %**	**69.91 %**	Yes
28	**24.78 %**	99.99 %	**4.82 %**	**9.32 %**	Yes
29	**65.40 %**	99.99 %	28.51 %	43.09 %	Yes
30	**77.01 %**	99.99 %	**41.68 %**	**57.51 %**	Yes
31	42.53 %	99.99 %	47.44 %	44.34 %	Yes
32	**49.97 %**	99.99 %	**35.00 %**	**42.67 %**	Yes
33	**61.36 %**	99.99 %	7.13 %	15.18 %	Yes
34	15.59 %	99.99 %	24.64 %	18.27 %	Yes
35	91.87 %	99.99 %	88.93 %	90.67 %	Yes
36	0 %	99.99 %	0 %	0 %	No
37	**96.63 %**	99.99 %	**58.25 %**	**76.48 %**	Yes
38	**65.06 %**	99.99 %	62.60 %	**64.05 %**	Yes
39	**67.44 %**	99.99 %	**45.63 %**	**56.62 %**	Yes
40	**97.87 %**	99.99 %	77.62 %	**88.62 %**	Yes
41	**33.76 %**	99.99 %	**15.08 %**	**22.57 %**	Yes
42	**91.41 %**	99.99 %	**59.02 %**	**74.96 %**	Yes
43	0 %	99.99 %	0 %	0 %	No
44	63.81 %	99.99 %	64.45 %	64.06 %	Yes
45	13.86 %	99.99 %	**7.07 %**	10.02 %	Yes
46	0 %	99.99 %	0 %	0 %	No
47	**88.44 %**	99.99 %	42.27 %	61.55 %	Yes
48	12.02 %	99.99 %	13.48 %	12.57 %	Yes
49	8.33 %	99.99 %	12.60 %	9.64 %	Yes
50	**97.12 %**	99.99 %	**43.79 %**	**65.31 %**	Yes
51	**22.78 %**	99.98 %	**31.28 %**	**25.56 %**	Yes
52	0 %	99.99 %	0 %	0 %	No
53	61.85 %	99.99 %	**31.24 %**	**44.44 %**	Yes
54	**72.48 %**	99.99 %	**16.07 %**	**30.15 %**	Yes
55	**81.51 %**	99.99 %	51.36 %	66.01 %	Yes
56	**82.46 %**	99.99 %	**3.57 %**	**8.39 %**	Yes

A.3.2 120 000 iterations

Table A.4: *Detailed results for all validation volumes for model 3 at 120 000 iterations. All numbers marked in bold indicates better result compared to the comparison model at 120 000 iterations.*

Patient	Sensitivity	Accuracy	Dice	F2-score	Nodule present
1	0 %	99.99 %	0 %	0 %	No
2	64.76 %	99.99 %	69.64 %	66.62 %	Yes

3	**30.78 %**	99.99 %	**11.83 %**	**18.75 %**	Yes
4	27.30 %	99.99 %	20.16 %	23.92 %	Yes
5	77.05 %	99.99 %	**54.12 %**	**65.88 %**	Yes
6	3.28 %	99.99 %	0.50 %	1.03 %	Yes
7	56.95 %	99.99 %	36.57 %	46.57 %	Yes
8	**76.53 %**	99.99 %	59.82 %	**68.84 %**	Yes
9	26.25 %	99.99 %	27.49 %	26.73 %	Yes
10	82.51 %	99.99 %	79.88 %	81.44 %	Yes
11	**42.59 %**	99.99 %	**52.83 %**	**46.17 %**	Yes
12	0 %	99.99 %	0 %	0 %	No
13	67.42 %	99.99 %	54.61 %	61.64 %	Yes
14	0.05 %	99.99 %	0.09 %	0.06 %	Yes
15	**43.05 %**	99.99 %	**11.93 %**	**21.07 %**	Yes
16	**55.20 %**	99.99 %	20.78 %	33.20 %	Yes
17	0 %	99.99 %	0 %	0 %	No
18	27.39 %	99.99 %	33.33 %	29.49 %	Yes
19	**75.59 %**	99.99 %	15.75 %	30.00 %	Yes
20	75.82 %	99.99 %	23.22 %	39.94 %	Yes
21	**4.46 %**	99.99 %	**6.58 %**	**5.12 %**	Yes
22	**43.80 %**	99.99 %	**53.84 %**	**47.33 %**	Yes
23	4.40 %	99.99 %	4.65 %	4.50 %	Yes
24	70.79 %	99.99 %	19.26 %	34.20 %	Yes
25	0.48 %	99.99 %	0.16 %	0.26 %	Yes
26	0 %	99.99 %	0 %	0 %	No
27	82.45 %	99.99 %	35.74 %	54.14 %	Yes
28	**56.96 %**	99.99 %	**11.85 %**	**22.58 %**	Yes
29	58.16 %	99.99 %	17.18 %	29.76 %	Yes
30	**76.68 %**	99.99 %	**19.61 %**	**35.43 %**	Yes
31	**41.01 %**	99.99 %	**44.47 %**	**42.33 %**	Yes
32	**3.59 %**	99.99 %	**1.68 %**	**2.47 %**	Yes
33	**56.94 %**	99.99 %	6.26 %	13.44 %	Yes
34	**50.14 %**	99.99 %	54.18 %	51.68 %	Yes
35	87.26 %	99.99 %	82.28 %	85.20 %	Yes
36	0 %	99.99 %	0 %	0 %	No
37	94.26 %	99.99 %	29.17 %	49.81 %	Yes
38	**64.81 %**	99.99 %	38.56 %	50.94 %	Yes
39	**57.36 %**	99.99 %	22.72 %	35.63 %	Yes
40	**98.25 %**	99.99 %	67.00 %	82.80 %	Yes
41	0 %	99.99 %	0 %	0 %	Yes
42	**89.23 %**	99.99 %	45.16 %	64.21 %	Yes
43	0 %	99.99 %	0 %	0 %	No
44	58.36 %	99.99 %	61.37 %	59.53 %	Yes
45	**69.09 %**	99.99 %	8.24 %	17.47 %	Yes
46	0 %	99.99 %	0 %	0 %	No
47	**78.47 %**	99.99 %	47.76 %	62.42 %	Yes
48	**12.29 %**	99.99 %	**10.24 %**	**11.38 %**	Yes

49	20.20 %	99.99 %	14.94 %	17.71 %	Yes
50	**94.01 %**	99.99 %	29.30 %	49.91 %	Yes
51	**13.14 %**	99.98 %	**18.29 %**	**14.81 %**	Yes
52	0 %	99.99 %	0 %	0 %	No
53	62.69 %	99.99 %	**19.98 %**	**33.79 %**	Yes
54	**68.38 %**	99.99 %	**11.69 %**	**23.25 %**	Yes
55	79.33 %	99.99 %	49.00 %	63.59 %	Yes
56	**87.47 %**	99.99 %	4.57 %	10.60 %	Yes

A.4 Comparison

A.4.1 100 000 iterations

Table A.5: *Detailed results for all validation volumes for the comparison model at 100 000 iterations.*

Patient	Sensitivity	Accuracy	Dice	F2-score	Nodule present
1	0 %	99.99 %	0 %	0 %	No
2	74.44 %	99.99 %	77.98 %	75.82 %	Yes
3	0 %	99.99 %	0 %	0 %	Yes
4	29.38 %	99.99 %	36.28 %	31.80 %	Yes
5	82.28 %	99.99 %	34.71 %	53.14 %	Yes
6	58.59 %	99.99 %	5.60 %	12.25 %	Yes
7	77.46 %	99.99 %	70.64 %	74.58 %	Yes
8	60.63 %	99.99 %	52.97 %	57.32 %	Yes
9	34.33 %	99.99 %	33.33 %	33.92 %	Yes
10	90.11 %	99.99 %	88.54 %	89.48 %	Yes
11	42.50 %	99.99 %	55.30 %	46.83 %	Yes
12	0 %	99.99 %	0 %	0 %	No
13	73.44 %	99.99 %	72.58 %	73.09 %	Yes
14	46.47 %	99.99 %	60.88 %	51.33 %	Yes
15	0 %	99.99 %	0 %	0 %	Yes
16	39.31 %	99.99 %	33.64 %	36.83 %	Yes
17	0 %	99.99 %	0 %	0 %	No
18	41.46 %	99.99 %	55.53 %	46.14 %	Yes
19	71.53 %	99.99 %	28.06 %	44.17 %	Yes
20	86.52 %	99.99 %	20.37 %	37.63 %	Yes
21	4.21 %	99.99 %	6.90 %	4.99 %	Yes
22	7.76 %	99.98 %	12.72 %	9.19 %	Yes
23	34.40 %	99.99 %	37.84 %	35.70 %	Yes
24	76.26 %	99.99 %	78.81 %	77.26 %	Yes
25	0 %	99.99 %	0 %	0 %	Yes
26	0 %	99.99 %	0 %	0 %	No
27	88.84 %	99.99 %	52.01 %	69.23 %	Yes

28	0 %	99.99 %	0 %	0 %	Yes
29	56.21 %	99.99 %	41.32 %	49.13 %	Yes
30	36.01 %	99.99 %	8.90 %	16.23 %	Yes
31	60.55 %	99.99 %	55.72 %	58.52 %	Yes
32	7.39 %	99.99 %	4.01 %	5.53 %	Yes
33	51.01 %	99.99 %	13.83 %	24.57 %	Yes
34	54.51 %	99.99 %	67.88 %	59.17 %	Yes
35	92.26 %	99.99 %	90.38 %	91.50 %	Yes
36	0 %	99.98 %	0 %	0 %	No
37	96.04 %	99.99 %	24.43 %	44.21 %	Yes
38	58.41 %	99.99 %	66.66 %	61.45 %	Yes
39	47.03 %	99.99 %	28.66 %	37.43 %	Yes
40	80.29 %	99.99 %	80.95 %	80.55 %	Yes
41	0 %	99.99 %	0 %	0 %	Yes
42	86.69 %	99.99 %	43.77 %	62.27 %	Yes
43	0 %	99.99 %	0 %	0 %	No
44	66.47 %	99.99 %	67.15 %	66.74 %	Yes
45	44.09 %	99.99 %	5.77 %	12.06 %	Yes
46	0 %	99.99 %	0 %	0 %	No
47	73.60 %	99.99 %	58.57 %	66.75 %	Yes
48	32.67 %	99.99 %	14.15 %	21.45 %	Yes
49	80.65 %	99.99 %	55.52 %	68.29 %	Yes
50	48.57 %	99.99 %	29.75 %	38.76 %	Yes
51	11.64 %	99.98 %	19.12 %	13.80 %	Yes
52	0 %	99.99 %	0 %	0 %	No
53	76.43 %	99.99 %	13.05 %	25.97 %	Yes
54	0 %	99.99 %	0 %	0 %	No
55	76.31 %	99.99 %	62.05 %	69.88 %	Yes
56	0 %	99.99 %	0 %	0 %	Yes

A.4.2 120 000 iterations

Table A.6: *Detailed results for all validation volumes for the comparison model at 120 000 iterations.*

Patient	Sensitivity	Accuracy	Dice	F2-score	Nodule present
1	0 %	99.99 %	0 %	0 %	No
2	75.26 %	99.99 %	79.94 %	77.07 %	Yes
3	5.90 %	99.99 %	2.21 %	3.54 %	Yes
4	53.54 %	99.99 %	52.81 %	53.26 %	Yes
5	81.59 %	99.99 %	45.85 %	62.20 %	Yes
6	69.98 %	99.99 %	15.25 %	28.73 %	Yes
7	79.19 %	99.99 %	72.50 %	76.37 %	Yes
8	64.32 %	99.99 %	61.46 %	63.14 %	Yes
9	46.71 %	99.99 %	41.96 %	44.69 %	Yes

10	90.17 %	99.99 %	87.66 %	89.15 %	Yes
11	36.23 %	99.99 %	51.51 %	41.11 %	Yes
12	0 %	99.99 %	0 %	0 %	No
13	70.99 %	99.99 %	70.55 %	70.81 %	Yes
14	15.31 %	99.99 %	25.57 %	18.24 %	Yes
15	0 %	99.99 %	0 %	0 %	Yes
16	40.52 %	99.99 %	49.35 %	43.64 %	Yes
17	0 %	99.99 %	0 %	0 %	No
18	54.49 %	99.99 %	66.21 %	58.68 %	Yes
19	68.96 %	99.99 %	19.31 %	34.00 %	Yes
20	88.96 %	99.99 %	30.50 %	50.36 %	Yes
21	4.24 %	99.99 %	5.92 %	4.78 %	Yes
22	16.38 %	99.98 %	24.89 %	18.98 %	Yes
23	14.67 %	99.99 %	21.41 %	16.78 %	Yes
24	83.02 %	99.99 %	89.04 %	85.33 %	Yes
25	17.04 %	99.99 %	1.67 %	3.64 %	Yes
26	0 %	99.99 %	0 %	0 %	No
27	88.73 %	99.99 %	52.22 %	69.34 %	Yes
28	1.74 %	99.99 %	1.42 %	1.60 %	Yes
29	62.76 %	99.99 %	58.40 %	60.94 %	Yes
30	36.12 %	99.99 %	10.18 %	17.89 %	Yes
31	39.83 %	99.99 %	40.97 %	40.28 %	Yes
32	1.45 %	99.99 %	1.14 %	1.31 %	Yes
33	55.05 %	99.99 %	10.53 %	20.46 %	Yes
34	47.88 %	99.99 %	61.54 %	52.54 %	Yes
35	92.22 %	99.99 %	93.24 %	92.63 %	Yes
36	0 %	99.99 %	0 %	0 %	No
37	99.80 %	99.99 %	41.83 %	64.21 %	Yes
38	54.27 %	99.99 %	53.81 %	54.08 %	Yes
39	51.94 %	99.99 %	44.32 %	48.60 %	Yes
40	92.23 %	99.99 %	82.18 %	87.93 %	Yes
41	4.70 %	99.99 %	1.78 %	2.84 %	Yes
42	88.20 %	99.99 %	66.66 %	78.11 %	Yes
43	0 %	99.99 %	0 %	0 %	No
44	74.69 %	99.99 %	70.66 %	73.02 %	Yes
45	50.00 %	99.99 %	10.17 %	19.48 %	Yes
46	0 %	99.99 %	0 %	0 %	No
47	77.44 %	99.99 %	80.66 %	78.70 %	Yes
48	8.13 %	99.99 %	8.32 %	8.20 %	Yes
49	84.20 %	99.99 %	75.19 %	80.35 %	Yes
50	91.32 %	99.99 %	48.24 %	67.28 %	Yes
51	11.92 %	99.98 %	19.85 %	14.19 %	Yes
52	0 %	99.99 %	0 %	0 %	No
53	78.12 %	99.99 %	14.52 %	28.39 %	Yes
54	17.44 %	99.99 %	11.41 %	14.40 %	Yes
55	81.15 %	99.99 %	64.20 %	73.40 %	Yes

56	79.70 %	99.99 %	12.30 %	24.97 %	Yes

Bibliography

[1] Nishio M, Sugiyama O, Yakami M, Ueno S, Kubo T, Kuroda T, et al. Computer-aided diagnosis of lung nodule classification between benign nodule, primary lung cancer, and metastatic lung cancer at different image size using deep convolutional neural network with transfer learning. PloS one. 2018;13(7):e0200721.

[2] Shin HC, Tenenholtz NA, Rogers JK, Schwarz CG, Senjem ML, Gunter JL, et al. Medical image synthesis for data augmentation and anonymization using generative adversarial networks. In: International Workshop on Simulation and Synthesis in Medical Imaging. Springer; 2018. p. 1–11.

[3] Murphy KP. Machine learning. a probabilistic perspective. Adaptive computation and machine learning series. Cambridge, MA : MIT Press, c2012.; 2012.

[4] Goodfellow I, Bengio Y, Courville A. Deep Learning. MIT Press; 2016. `http://www.deeplearningbook.org`.

[5] Fakoor R, Ladhak F, Nazi A, Huber M. Using deep learning to enhance cancer diagnosis and classification. In: Proceedings of the International Conference on Machine Learning. vol. 28. ACM New York, USA; 2013. .

[6] El Emam K, Rodgers S, Malin B. Anonymising and sharing individual patient data. bmj. 2015;350:h1139.

[7] Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, et al. Generative adversarial nets. In: Advances in neural information processing systems; 2014. p. 2672–2680.

[8] Isola P, Zhu JY, Zhou T, Efros AA. Image-to-image translation with conditional adversarial networks. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE; 2017. p. 5967–5976.

[9] Guibas JT, Virdi TS, Li PS. Synthetic Medical Images from Dual Generative Adversarial Networks. arXiv preprint arXiv:170901872. 2017;.

[10] Haykin SS, Haykin. Neural networks and learning machines. vol. 3. Pearson Upper Saddle River; 2009.

[11] Glorot X, Bordes A, Bengio Y. Deep sparse rectifier neural networks. In: Proceedings of the fourteenth international conference on artificial intelligence and statistics; 2011. p. 315–323.

[12] Kingma DP, Ba J. Adam: A method for stochastic optimization. arXiv preprint arXiv:14126980. 2014;.

[13] Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R. Dropout: a simple way to prevent neural networks from overfitting. The Journal of Machine Learning Research. 2014;15(1):1929–1958.

[14] Ulyanov D, Vedaldi A, Lempitsky V. Instance normalization: The missing ingredient for fast stylization. arXiv preprint arXiv:160708022. 2016;.

[15] Yi X, Walia E, Babyn P. Generative adversarial network in medical imaging: A review. arXiv preprint arXiv:180907294. 2018;.

[16] Salimans T, Goodfellow I, Zaremba W, Cheung V, Radford A, Chen X. Improved techniques for training gans. In: Advances in Neural Information Processing Systems; 2016. p. 2234–2242.

[17] Mirza M, Osindero S. Conditional generative adversarial nets. arXiv preprint arXiv:14111784. 2014;.

[18] Zhu JY, Park T, Isola P, Efros AA. Unpaired image-to-image translation using cycle-consistent adversarial networks. In: Proceedings of the IEEE International Conference on Computer Vision; 2017. p. 2223–2232.

[19] Taha AA, Hanbury A. Metrics for evaluating 3D medical image segmentation: analysis, selection, and tool. BMC medical imaging. 2015;15(1):29.

[20] DePoy E, Gitlin LN. Introduction to research-e-book: Understanding and applying multiple strategies. Elsevier Health Sciences; 2013.

[21] Kalender W. Computed tomography : fundamentals, system technology, image quality, applications. Erlangen [Germany] : Publicis Publishing, [2011]; 2011.

[22] Dove EL. Physics of Medical Imaging–An Introduction. Biomedical Engineering The University of Lowa. 2003;p. 3–13.

[23] Dove EL. Notes on computerized tomography. Bioimaging Fundamentals. 2001;.

[24] Patz Jr EF, Campa MJ, Gottlin EB, Trotter PR, Herndon JE, Kafader D, et al. Biomarkers to help guide management of patients with pulmonary nodules. American journal of respiratory and critical care medicine. 2013;188(4):461–465.

[25] Armato III SG, McLennan G, Bidaut L, McNitt-Gray MF, Meyer CR, Reeves AP, et al. The lung image database consortium (LIDC) and image database resource initiative (IDRI): a completed reference database of lung nodules on CT scans. Medical physics. 2011;38(2):915–931.

[26] Larobina M, Murino L. Medical image file formats. Journal of digital imaging. 2014;27(2):200–206.

[27] Mildenberger P, Eichelberg M, Martin E. Introduction to the DICOM standard. EUROPEAN RADIOLOGY. 2002;(4):920.

[28] Whitcher B, Schmid VJ, Thornton A. Working with the DICOM and NIfTI Data Standards in R. Journal of Statistical Software. 2011;(6).

[29] Salehinejad H, Valaee S, Dowdell T, Colak E, Barfett J. Generalization of deep neural networks for chest pathology classification in x-rays using generative adversarial networks. In: 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE; 2018. p. 990–994.

[30] Chuquicusma MJ, Hussein S, Burt J, Bagci U. How to fool radiologists with generative adversarial networks? a visual turing test for lung cancer diagnosis. In: 2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018). IEEE; 2018. p. 240–244.

[31] Radford A, Metz L, Chintala S. Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:151106434. 2015;.

[32] Medical Image Synthesis with Deep Convolutional Adversarial Networks. IEEE Transactions on Biomedical Engineering, Biomedical Engineering, IEEE Transactions on, IEEE Trans Biomed Eng. 2018;(12):2720.

[33] End-to-End Adversarial Retinal Image Synthesis. IEEE Transactions on Medical Imaging, Medical Imaging, IEEE Transactions on, IEEE Trans Med Imaging. 2018;(3):781.

[34] Nie D, Trullo R, Lian J, Petitjean C, Ruan S, Wang Q, et al. Medical image synthesis with context-aware generative adversarial networks. In: International Conference on Medical Image Computing and Computer-Assisted Intervention. Springer; 2017. p. 417–425.

[35] Yang Q, Li N, Zhao Z, Fan X, Chang EI, Xu Y, et al. MRI Cross-Modality NeuroImage-to-NeuroImage Translation. arXiv preprint arXiv:180106940. 2018;.

[36] Chartsias A, Joyce T, Dharmakumar R, Tsaftaris SA. Adversarial image synthesis for unpaired multi-modal cardiac data. In: International Workshop on Simulation and Synthesis in Medical Imaging. Springer; 2017. p. 3–13.

[37] Zhang Y, Miao S, Mansi T, Liao R. Task driven generative modeling for unsupervised domain adaptation: Application to x-ray image segmentation. In: International Conference on Medical Image Computing and Computer-Assisted Intervention. Springer; 2018. p. 599–607.

[38] Wolterink JM, Dinkla AM, Savenije MH, Seevinck PR, van den Berg CA, Išgum I. Deep MR to CT synthesis using unpaired data. In: International Workshop on Simulation and Synthesis in Medical Imaging. Springer; 2017. p. 14–23.

[39] Welander P, Karlsson S, Eklund A. Generative adversarial networks for image-to-image translation on multi-contrast MR images-A comparison of CycleGAN and UNIT. arXiv preprint arXiv:180607777. 2018;.

[40] Dar SUH, Yurt M, Shahdloo M, Ildız ME, Çukur T. Synergistic Reconstruction and Synthesis via Generative Adversarial Networks for Accelerated Multi-Contrast MRI. arXiv preprint arXiv:180510704. 2018;.

[41] Pettersson SB. Managing imbalanced training data by sequential segmentation in machine learning; 2019. `http://urn.kb.se/resolve?urn=urn:nbn:se:liu:diva-155091`.

[42] Çiçek Ö, Abdulkadir A, Lienkamp SS, Brox T, Ronneberger O. 3D U-Net: learning dense volumetric segmentation from sparse annotation. In: International conference on medical image computing and computer-assisted intervention. Springer; 2016. p. 424–432.

[43] Ronneberger O, Fischer P, Brox T. U-net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical image computing and computer-assisted intervention. Springer; 2015. p. 234–241.

[44] Gibson E, Li W, Sudre C, Fidon L, Shakir DI, Wang G, et al. NiftyNet: a deep-learning platform for medical imaging. Computer methods and programs in biomedicine. 2018;158:113–122.

[45] Chu C, Zhmoginov A, Sandler M. CycleGAN, a master of steganography. arXiv preprint arXiv:171202950. 2017;.