A physics guided neural network approach for dose prediction in automated radiation therapy treatment planning

by

Thierry Meerbothe

to obtain the degree of Master of Science
at the Delft University of Technology,

Student number:	4479998	
Project duration:	April 20, 2020 – February 9, 2021	
Supervisors:	Dr. Zoltán Perkó	TU Delft
	Dr. Tomas Janssen	NKI
	Dr. Rita Simões	NKI

Abstract

Radiotherapy treatment planning is a complex and time consuming process prone to differences as result of choices of individual planners. Autoplanning systems have been introduced to both reduce the time consumption and to counteract the influence of individual planning choices. Although autoplanning generally increases performance of the treatment plans, the plans still need to be checked manually ensure plan accuracy. This counteracts the advantages introduced with autoplanning. Increasing plan consistency with knowledge based planning (KBP) or automatic plan evaluation would be clear ways to improve upon this problem. For both improvements, deep learning can be an important tool to produce accurate and fast 3D dose distributions that can be used for KBP or plan evaluation. In this study such a deep learning tool is developed in two parts. In the first part a structure based dose prediction model is developed. In the second part the model is enriched with physics based information to make a more realistic and accurate prediction.

The structure based deep learning model is made using a U-Net architecture with patient organ structures as input. It is used to predict the dose distribution for prostate patients treated with volumetric modulated arc therapy (VMAT) plans, generated by autoplanning. Different loss functions have been tested to see which achieve the best results. Including physics information in the neural network prediction is done using a hybrid physics-data model. For this, an approximate dose distribution, the segment dose, is used as extra input for the dose prediction network. This segment dose is created by predicting multi leaf collimator (MLC) positions and reconstructing the corresponding dose with a simple dose engine. In both parts training was done using a dataset consisting of 89 patients with structure, dose and plan information in DICOM format. The dataset was divided into a training, validation and test set for model training, which was done in two phases. The first phase includes just one patient with different translational and rotational augmentations. The second phase uses the entire dataset. Early stopping was included to prevent overfitting.

The predictions are evaluated using several dose characteristics and dose volume histogram (DVH) curves. Moreover, the structure based dose prediction is also compared with a currently used rectum prediction method for quality assurance, based on principal component analysis. The U-Net architecture proved to be able to accurately predict the dose of the patients in the test set. The best loss function for accurate dose characteristics prediction was found to be the weighted mean squared error (WMSE) loss. The prediction model with the WMSE loss predicted several PTV DVH statistics within a 1.5% error and the rectum statistics within 3.5% error. Furthermore, the model predicts the DVH points of the PTV within 0.84 ± 0.40 Gy average absolute dose difference and the rectum in 0.91 ± 0.72 Gy average dose difference. Finally, the rectum DVH prediction proved to be more accurate than currently used model based on principal component analysis. Unfortunately, the hybrid physics-data model did not improve the prediction accuracy of the dose prediction model in terms of DVH statistics and DVH prediction, as the MLC positions could not be predicted accurately enough.

The performance of the structure based prediction is similar to the performance of state-of-the-art prediction models. However, it should be taken into account that a homogeneous dataset is used. Although the hybrid model with the predicted segment doses did not improve the prediction accuracy, a significant effect could be seen from using the correct MLC positions instead of the predicted. Thus it appeared that the approximated dose distribution did not contain enough useful information for the neural network to improve the prediction. The bottleneck of the process is therefore identified as the prediction of the MLC positions. As such, it would be interesting to focus on segment prediction in follow up research.

Contents

1 Introduction

Cancer is one of the most prevalent current day causes of death in the Netherlands. According to the central bureau of statistics, almost 47000 people have died from cancer in 2019. This makes cancer responsible for roughly a third of all deaths [1]. Research on cancer treatment therefore very relevant. When cancer is diagnosed in a patient, one of the three main treatment options is radiation therapy. Radiation therapy is a complex form of treatment that aims at killing the cancer cells by damaging the DNA of the cells beyond repair.

Radiotherapy treatment makes use of external treatment beams or internal radiation sources to irradiate and kill a tumor. As this is a complex process, a lot of steps have to be taken to ensure an accurate delivery of the radiation to the tumor. One of the important steps in this process is the treatment planning, which is the process of determining how to treat the patient in the best way possible.

Conventional radiotherapy treatment planning is a manual and time consuming process required to be done for every patient individually. Planners need to have a lot of knowledge and experience to produce a good plan. Vast knowledge and experience does however not guarantee consistent treatment plans. The multiple manual steps and the different possible constraint and objectives that can be chosen by planners make the entire process very sensitive to differences as a result of subjective choices [2].

One method that has been developed in order to prevent differences between planners is automated treatment planning or autoplanning. Automated treatment planning systems try to replicate the steps a dosimetrist would take during planning in a consistent way. Multiple studies have already shown that automated treatment planning systems perform on par or better compared to plans of experts [3] [4]. Besides this increase in accuracy, autoplanning also reduces the time needed to plan a treatment.

Autoplanning is however far from perfect. As all patients are different, treatment goals that might be applicable for one patient, can be clinically less relevant for another. As there are a lot of different objectives and constraints which need to be taken into account during planning, treatment planning is a problem of finding the best compromise and different plans can yield comparable results. Because of this, autoplans are not always optimal or clinically acceptable [5]. As such, most plans are still manually checked by dosimetrists in order to ensure acceptability and clinical optimality, for which trial and error based approaches are still mostly used. Without a dosimetrist it is hard to determine if a plan is acceptable or not.

One option is to increase plan optimality and consistency by integrating treatment planning information based on earlier used treatment plans. This method is also called knowledge based planning (KBP). It generally works by predicting the patient dose volume histogram (DVH) and using the DVH to better estimate the dose objectives for the planning stage [6]. Another option is to evaluate the autoplans automatically and consistently, using a similar knowledge based method and evaluating individual plans with the DVH predictions [7]. From earlier studies it has been shown that simplistic knowledge based methods can accurately predict these DVH curves [8].

As result of the advancement in the field of artificial intelligence and deep learning over the last ten years, another option for KBP applications has recently emerged. With the development of convolutional neural networks (CNNs), deep learning proved to be a tool that can be used to predict accurate and complete 3D dose distributions, instead of specific DVH curves, by using structured image information as input. This approach has multiple benefits. It is easier to predict a DVH based on 3D prediction information as no parameters have to be engineered to predict a DVH from lower dimensional data. Also the 3D dose prediction includes spatial information that is not available in other KBP approaches. This makes it is possible to provide voxel level feedback on a treatment plan so that the dosimetrist can easily see what parts of the plan differ with the prediction. Finally, such an approach could help in the long term in a tool that can produce clinically acceptable plans automatically, without the need of a planner or optimization algorithm [9] [10].

In this study, the aim is to further investigate the clinical application of deep learning dose prediction tools, specifically for autoplanned prostate patients treated with volumetric modulated arc therapy (VMAT), by including physics information within the prediction network.

In the first part of this research a U-Net neural network architecture is used to predict dose distributions for patients originally planned with autoplanning. Using different loss functions, different prediction models are investigated to find the model with the best accuracy. Next, this prediction model is compared to models from literature. Lastly, the clinical accuracy of the model is researched and it is investigated if the model can provide an improvement over the current KBP prediction method.

In the second part of the study, a novel approach to include physics information within the dose prediction is investigated. This approach uses a physics guided neural network based on a hybrid physics-data model. For this, a second neural network and a simple dose engine are used to predict individual beam doses. The corresponding dose distribution estimate for all beam directions is used as extra input information for the neural network. The goal of this part is to predict a physically more correct dose distribution, which is both more accurate and also more realistic. Moreover, a neural network that is able to predict a physically achievable dose distributions can be a step closer to the development of an automatic treatment planning system based on artificial intelligence.

2

Theory

In this chapter, the necessary theoretical background needed to understand the research is discussed. First, the basics of radiotherapy treatment and autoplanning are explained in Section 2.1 and Section 2.2. After that, Section 2.3 introduces the basics of deep learning and more specifically the theory on neural networks for dose prediction such as convolutional neural networks in Section 2.4. Finally, the subject of physics guided neural networks is introduced in Section 2.5.

2.1. Radiation therapy treatment

Radiation therapy is one of three main modalities to treat cancer besides surgery and chemotherapy. It makes use of ionizing radiation to damage the DNA of cancer cells, while sparing as much healthy tissue as possible. There are multiple types of radiation used for radiotherapy, but the type used most often is ionizing electromagnetic waves or photons. Compared to modern radiation techniques, as for example proton therapy, photon therapy is simpler and cheaper and therefore much more widespread.

Photon therapy is delivered using various techniques. The first and simplest method of photon irradiation still used is three dimensional conformal radiation therapy (3D CRT). In 3D CRT a large homogeneous field of photons is produced using a linear accelerator (LINAC). This field is shaped to match the tumor shape from beams eye view using a multi leaf collimator (MLC) through which a homogeneous field with the shape of the tumor is projected onto the body of the patient. The MLC consists of multiple small leafs which can be moved to produce the correct shape. The tumor is generally irradiated from a couple of different directions. An improvement on 3D CRT, is intensity modulated radiation therapy (IMRT). IMRT uses the same principles as 3D CRT but instead of providing only a single field with the shape of the tumor from each direction, IMRT uses the MLC leafs in different positions for each beam direction. In this way different parts of the tumor can be irradiated with different intensities. IMRT therefore has much more degrees of freedom that can be used to produce an optimal dose distribution within the patient and improved performance over 3D CRT [11]. The most recent development is volumetric modulated arc therapy (VMAT). Instead of treating the patient with varying intensities from a couple directions, VMAT treats the patient with continuous arcs by rotation of the LINAC around the patient. During the arc rotation, the MLC leafs continuously vary in position. This gives the possibility to irradiate the target from more directions. This again results in improvements such as increased target conformity and reduced dose to OAR [12]. It also reduces the treatment time compared to IMRT.

2.1.1. Treatment, imaging and delineation

To treat a patient using radiation therapy, a lot of steps have to be taken. This starts with a physician that prescribes the treatment. Depending on the goal of the treatment, which can be either palliative or curative, the physician determines specific treatment goals. These goals include the amount of dose that needs to be deposited in the tumor and the limits to the dose in healthy tissue. When treatment has been prescribed, the process to determine how to give the treatment begins.

The first step in the process is imaging the patient properly. This is predominantly done with computerized tomography (CT), while other techniques such as magnetic resonance imaging (MRI) and positron emission

tomography (PET) are used complementary to CT. Proper images are needed in almost all steps of the radiation therapy process. Most important for treatment planning, imaging is used to determine the location of the tumor before treatment and thus to identify which part of the body needs to be irradiated. Besides, the tumor resides within healthy tissue, which needs to be spared as much as possible. This healthy tissue can include important organs which can give rise to serious complications when irradiated too much. These organs are generally referred to as organ at risk (OAR) and need to be taken into consideration when planning the treatment in order to minimize side effects. Finally, the geometry of the patient heavily influences to dose distribution. A CT scan can be used to determine the photon attenuation properties of the contents of the body and is thus essential to help determine the delivered dose in the patient. [13].

The next step is determining where in the patient the tumor and the OARs are located. This process is called delineation. Delineation is generally done manually using the CT scan and knowledge of tumor growth and anatomy. Recently there are also developments in the field of automatic segmentation with the help of machine learning and deep learning. The exact location of the tumor is hard to determine from the images. As such, delineation often has a large uncertainty. To ensure that the target volume encloses the entire tumor, some uncertainty margins are taken into account during delineation. First, the region where the tumor is visible is determined, this is called the gross tumor volume (GTV). Next, the GTV is extended with the volume that is suspected to have locally spread tumor cells present, but which are not necessarily visible through the imaging modality. This volume is called the clinical target volume (CTV). Lastly the volume is extended to account for other uncertainties, such as patient setup and treatment delivery based uncertainties. This last volume is called the planning target volume (PTV) and is the volume which is eventually treated on. This is schematically viewed in Figure 2.1, where it can also be seen that the treated volume can overlap with the OAR.

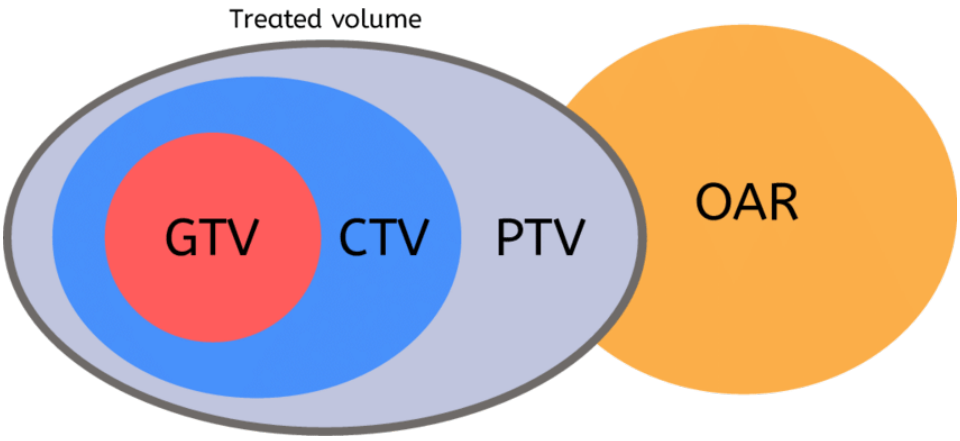


Figure 2.1: Schematic view of the GTV, CTV, PTV and OAR [14]

2.1.2. Treatment planning

When the final delineation has been determined, the optimal machine parameters of the delivery device need to be found, such that the patient is treated in the best possibly way. The dose to the tumor needs to be sufficient to kill the tumor cells, while keeping the dose as low as possible to the healthy tissues. More specifically, the dose distribution needs to be within the limits of the treatment goals. The process of finding the machine parameters that result in such a clinically acceptable dose distribution for the patient is called treatment planning.

There are two ways to approach a treatment planning problem: forward planning and inverse planning. In forward planning the machine parameters such as the collimator positions are first chosen, and with that a check is performed to see whether the treatment goals are met. The machine parameters are modified until an acceptable plan has been found. However, in current day external radiotherapy the amount of tuneable parameters is too high to efficiently find machine parameters that make a good plan. This makes forward planning only usable in relatively simple cases. Inverse planning on the other hand starts with defining specific objectives, constraints and priorities for the different structures. The fluence profile is then optimized

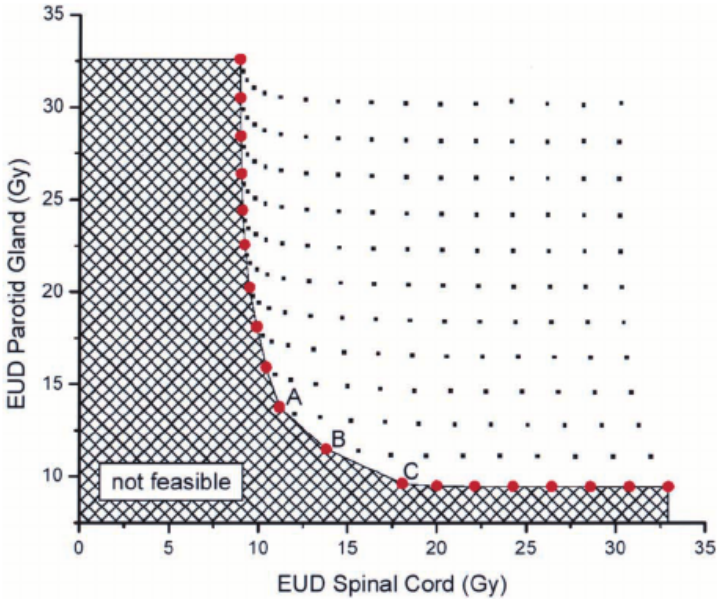


Figure 2.2: Illustration of a Pareto front, indicated by the red dots, between two objectives: the Equivalent uniform dose (EUD) in the parotid gland and the EUD in the spinal cord [17]. Improving in one objective cannot be realized without compromising the other objective when on the Pareto front.

based on the objective functions and machine parameters are found to match the constraints of the objectives. By modifying the constraints and objectives in several rounds of optimization, the dose distribution can be improved until satisfactory [15].

The optimization problem that needs to be solved is a multi criteria optimization (MCO) problem. First, the geometry of the patient is discretized in voxels. Each voxel i receives a dose d_i which is dependent on the intensities of the individual beamlets x and the dose influence matrix A. The beamlet intensities are modified in the optimization problem while the dose influence matrix is calculated using the CT. This results in the simple relation given in Equation 2.1:

$$d(x) = Ax \tag{2.1}$$

Next, the voxels are divided in the structures earlier defined in delineation. Objective functions are defined with the objectives for the different structures instead of for every voxel individually. This vastly decreases the size of the MCO problem. The objective functions typically use dose as input variable. Therefore the general MCO problem is defined as:

$$\begin{aligned} \min_x \quad & f(x) \\ \text{subject to} \quad & g(x) = [g_1(x), g_2(x), \ldots, g_m(x)] \leq 0, \end{aligned} \tag{2.2}$$

where $g(x)$ are the constraining functions and $f(x)$ is the objective function needed to be minimized. In practice such a MCO is optimized several times by an optimization algorithm. After each optimization the radiation technician can modify the objectives to alter the dose distribution and produce a personalized treatment plan [16].

All this is done in the environment of a treatment planning system (TPS). There are different TPS commercially available, but for this project Pinnacle^3, developed by Philips healthcare, is of most interest as it is the TPS used in the planning of the treatment plans of the patients in this study.

Eventually the goal of the optimization is to produce a plan that is optimal for the patient. There is however not a single optimal plan. As there is a large amount of objectives and constraints, the main objective becomes to find a plan with the best compromise in the objectives and constraints. Such a plan must be located on the so-called Pareto front. The Pareto front is defined as a surface in the solution space where an objective cannot be improved without compromising another objective. This is also illustrated in Figure 2.2.

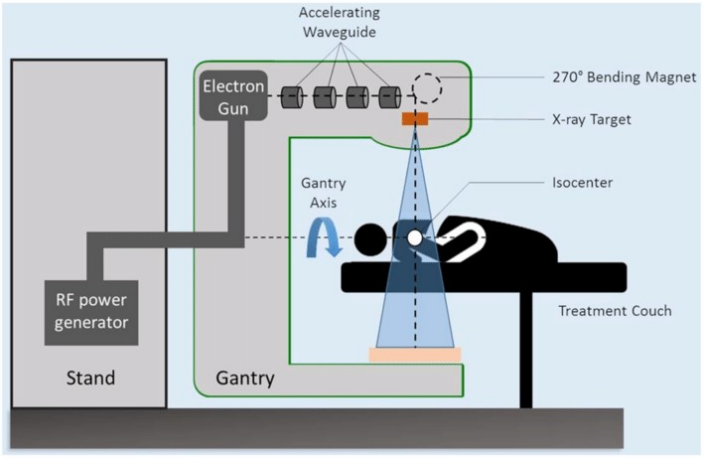


Figure 2.3: Schematic representation of the inside of a LINAC [18]

A plan on the Pareto front is optimal as there is no improvement possible without compromising another objective, but it is not guaranteed to be the best plan clinically. For example consider again the Pareto front in Figure 2.2: The plan can still be Pareto optimal when it delivers a dose of 32.5 Gy to the spinal cord, while a Pareto optimal plan with a dose of 18 Gy to the spinal cord has only a slightly higher the dose in the parotid gland. Therefore, this latter point on the Pareto front would be clinically more favorable. During planing, these kinds of considerations have to be done constantly, with often a lot more objectives than only two. This becomes a very complex problem which is very sensitive to choices of the planner. This is one of the reasons why autoplanning has been introduced, which will be further discussed in Section 2.2.

2.1.3. Treatment delivery

After the machine parameters have been chosen, the treatment can finally be delivered to the patient. The treatment machine will irradiate the patient according to the treatment parameters. The most widely used machine to deliver the treatment is the LINAC, which consists of various inner parts, schematically shown in Figure 2.3. The important parts of the LINAC are located within a large piece of equipment which can rotate around the patient called the gantry. Inside the gantry, an electron gun produces electrons, which are accelerated in a linear accelerator. Next, the electrons are bent by a magnet to direct the beam to the patient. The beam of electrons then hits an x-ray target or scattering foil to create a photon beam of the desired energy which is further shaped in the treatment head by several filters, collimators and MLCs. The patient is immobilized on a freely rotatable treatment table for delivery and only sees the outer shell of the gantry together with the treatment table, as displayed in Figure 2.4.

2.2. Autoplanning

An important recent development in treatment planning is the use of autoplanning. Autoplanning can improve on several difficulties that arise in manual planning. Manual treatment planning is a time intensive process as there are many complex steps in the process. By automating some of the steps, autoplanning can reduce the amount of time needed to spend on the production of a treatment plan. Apart from that, differences between individual planners can give rise to inconsistent treatment plans by subjective individual choices. This can be negated by using an objective method like autoplanning. Finally, a plan from an autoplan system can also provide a solid starting point to start optimizing from. This is also called a warm start [4].

For the relevant TPS Pinnacle^3, autoplanning is included within the TPS. The AutoPlanning module in Pinnacle^3 consists of two main parts. First of all, the planning process is simplified by using templates, which are called techniques. In a technique all treatment parameters are defined, such as the prescription dose for the PTV and other optimization goals for a certain patient geometry. Second, there is an automatic optimization part which is called the Auto-planning engine (APE). When all settings are defined, the APE first defines all optimization objectives from the goals according to their respective priority. Next, the plan parameters are

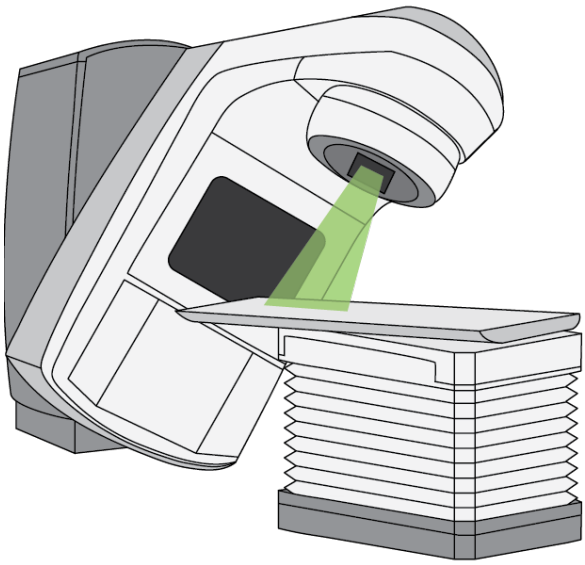


Figure 2.4: Schematic view of the outer parts of a treatment device (LINAC) with rotatable gantry and treatment table [19]

tweaked in several optimization loops in order to match the treatment goals. Next the dose in the organ at risk is lowered until the main objectives are compromised significantly when trying to reduce the dose even more. Lastly, the target conformity, body dose, and uniformity of the dose are controlled by the system as well [20].

Autoplanning, and the AutoPlan module in Pinnacle^3 specifically, have shown in multiple studies that they perform on par, but often better compared to manual treatment planning. For example, the study of Hazell et. al. showed that for different head and neck tumor cases, which had been already delivered, the dose in the OAR could have been reduced significantly while retaining the same target coverage with an autoplanned replan [20]. Another example is the study of Krayenbuehl et. al. in which an even bigger sample of head and neck tumor patients was replanned. In this study, the autoplans again proved to have very good target coverage and a better OAR sparing. Besides, it was also shown that the time needed to make a treatment plan was reduced significantly [3]. The potential benefits of autoplanning have therefore already been proven.

However, autoplanning does not always produce better plans. Because of this, the autoplans still need to be checked manually by dosimetrists to see whether the plans are acceptable. If not, the plan is either modified or replanned completely. This partly negates the advantages of autoplanning in the first place and supports the need of an automatic quality assurance pathway for autoplans.

2.3. Basics of deep learning

Deep learning can be an important tool to predict voxelwise dose within a patient, based on historical data from automated treatment plans. Deep learning is a subset of machine learning, in which the goal is to extract information or patterns from data. Deep learning makes use of multiple layers of transformations to train a model for different purposes such as classification into a small amount of groups, segmentation of images or, as relevant for this research, dose prediction.

The different layers of transformations form a network of interconnected nodes which make up acyclic graphs. The input for a network is given in its first layer and the information is fed through the network to produce an output at the end of the architecture. Using known data, the transformations within the graph can be altered to produce the best possible outcome. Because the interconnection of several nodes within the network resembles structures within a brain, the deep learning networks are often called neural networks.

2.3.1. The perceptron

The most basic form of a deep learning architecture is a feedforward network based on the perceptron algorithm. The perceptron will be used to illustrate the basic underlying concepts of deep learning. The perceptron algorithm is an algorithm that is used for binary classification. It maps an input $\mathbf{x}$ to an output value $f(\mathbf{x})$. As binary classification algorithm the perceptron creates a linear decision boundary that can give either one of two possible classifications as outcome. The algorithm consists of three main steps. First, the input values are multiplied by weights. Next, the calculated outcomes are summed and finally the Heaviside step function is taken. This is according to:

$$f(\boldsymbol{x}) = \Theta(\mathbf{w}^{\mathrm{T}}\mathbf{x} + b) = \begin{cases} 1, \\ 0, \end{cases} \tag{2.3}$$

where $\mathbf{w}$ are the weights, $\mathbf{x}$ is the input and b is a possible bias. Graphically, this would look as in Figure 2.5 where the input and the weights are multiplied and fed through the Heaviside function to generate an output value.

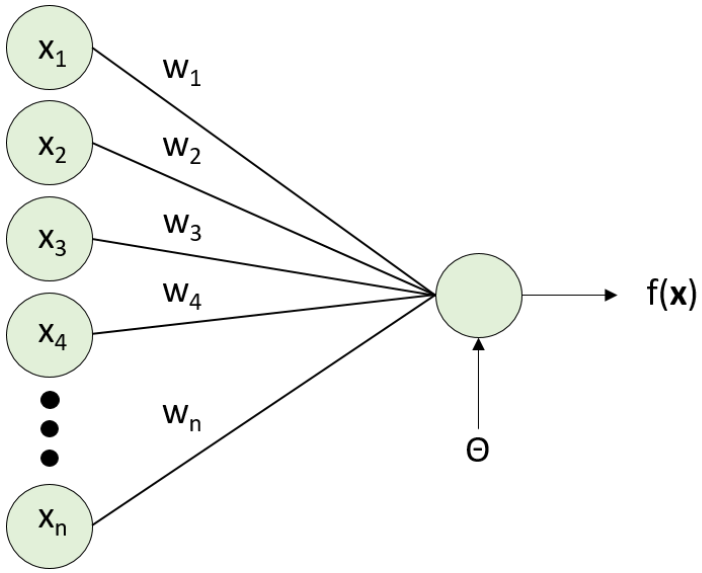


Figure 2.5: Graphic representation of the perceptron. x_n denotes the input values, w_n the weights and Θ the step function

The perceptron is very good in distinguishing two classes, but only when the classes are linearly separable, as the transformation it applies can only produce a linear decision boundary. Consequently, the perceptron on itself is not applicable for a wide range of problems.

2.3.2. The multilayer perceptron

The perceptron can be extended to lift this restriction. This is done by adding more layers between the input and the output of the algorithm. These intermediate layers are called hidden layers as they do not directly produce an output. This results in a so called multilayer perceptron (MLP), which is the most basic example of a deep neural network. Now instead of $\mathbf{x}$ being directly mapped to $f(\mathbf{x})$ with a single function, the function consists of a chain of mappings, which represent different layers, each interconnecting all the nodes between the different layers to make a fully interconnected network:

$$f(\mathbf{x}) = f^{(n)}(\cdots f^{(2)}(f^{(1)}(\mathbf{x}))) \tag{2.4}$$

Here the different subfunctions $f^{(n)}$ are the different layers of the model. Without the step function in every layer, the layers are just multiplications of the input vector with a certain weight. As such the transformation still remains linear when only adding more layers without the step function. In order to create the nonlinearity, every layer is followed by a nonlinear transformation such as the step function. Such a nonlinear transformation can consist of various transformations and is generally called an activation function. Every distinct layer is now written as in Equation 2.5:

$$\mathbf{h}^{(n)} = f^{(n)}(\mathbf{x}) = g^{(n)}(\mathbf{W}_n^T\mathbf{x} + b_n) \tag{2.5}$$

Here $g^{(n)}$ represents the activation function and $\mathbf{h}^{(n)}$ is the output of a hidden layer [21]. There are numerous possibilities for activation functions, but several instances in literature suggest that the rectified linear unit (ReLU) is a generally consistent choice for many applications [22]. The ReLU is defined as in Equation 2.6:

$$g(x) = \max\{0, x\} \tag{2.6}$$

Which graphically looks as in Figure 2.6.

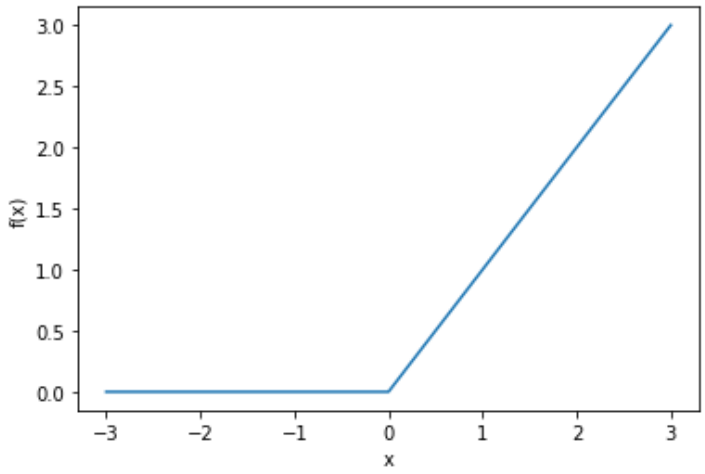


Figure 2.6: Graphical representation of the rectified linear unit

For an input value below zero, the ReLU outputs a value of zero, while it outputs the input value when the input is larger than zero. Other possible activation functions include the sigmoid function and the hyperbolic tangent function. The ReLU, or any other activation function, is applied after every summation within the network layer, just as in the perceptron. With the activation function, the entire fully connected MLP network can be specified and is no longer only linear. An example of such a network can be seen in Figure 2.7.

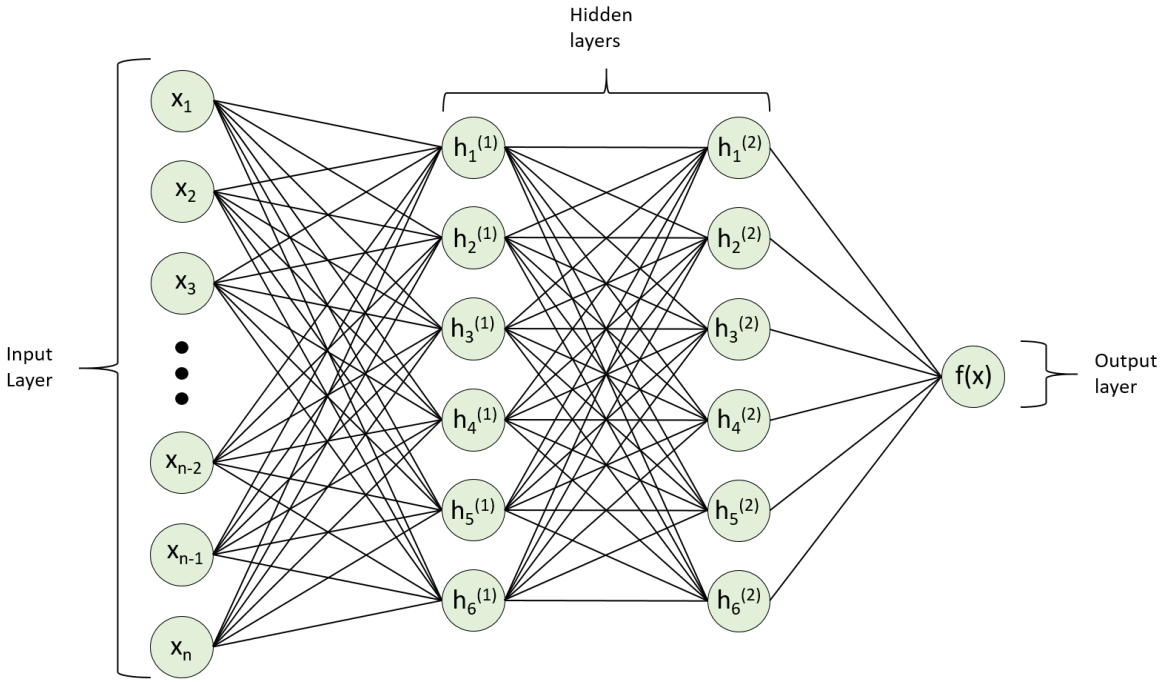


Figure 2.7: Graphical representation of the multilayer perceptron with two hidden layers.

2.3.3. Loss functions

The neural network will give a certain output based on the input it receives, by propagating the input through all the layers. In order to achieve an optimal prediction from the input data, the parameters of the network, the weights and biases, should be optimized for the task at hand, a process generally called training. This is done by optimizing a loss or cost function, by tweaking the weights in the network. Defining the loss function is an important aspect of the design of a neural network as the loss function is a measure of how good the model is, or in other words, how big the error compared to the ground truth is. Examples of loss functions that are often used in deep learning include the mean squared error (MSE) loss and the average binary cross entropy loss given by Equation 2.7 and Equation 2.8, respectively [21] [23].

$$L_{MSE}(x) = \frac{1}{N}\sum_{i=1}^{N}(y_i - f(x_i))^2 \tag{2.7}$$

$$L_{BCE}(x) = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log\big(p(f(x_i))\big) + (1-y_i)\log\big(1-p(f(x_i))\big)\right] \tag{2.8}$$

In both equations N is the sample size. $f(x_i)$ is the output value of the model for a specific sample and y_i is the true value. Also, $p(f(x_i))$ denotes the softmax value of the output in L_{BCE}.

2.3.4. Gradient descent

Optimization of the loss function is usually done by a process called gradient descent. In gradient descent the derivative of the loss function is used to identify how a certain change in the parameters $\mathbf{w}$ of the model influences the value of the loss. The derivative can therefore be used to minimize the loss function. In mathematical terms this process is described by Equation 2.9 [24]:

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \alpha\frac{1}{N}\sum_{i=1}^{N}\nabla_{\mathbf{w}}L(x_i) \tag{2.9}$$

Here the subscript t denotes the t-th update of the weights, subscript i denotes the i-th sample and α is the learning rate, which represents the size of each gradient descent step. Gradient descent ensures that the next step is always in the opposite direction of the derivative and thus ensures a net effect per step. There are many variations on this basic idea. Another variation on gradient descent for example makes use of the hessian to calculate a variable step size, known as Newtons method. Both of these methods are computationally quite expensive however, and for the many parameters in considerably deep neural networks, that can cause problems. A method which is computationally less expensive is stochastic gradient descent. In stochastic gradient descent, the gradient is not calculated exactly, but approximated by only calculating part of the gradient given by a single example x_i. This can be seen in Equation 2.10.

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \alpha\nabla_{\mathbf{w}}L(x_i) \tag{2.10}$$

If the derivative of a loss example is randomly chosen each iteration, the loss function is directly optimized [24].

2.3.5. Backpropagation

In deep learning, the gradient descent steps are in fact a little more complex as not all the weights can be updated directly from the gradient of the loss function. Since the network consists of multiple layers, the information of the loss function needs to be propagated backward through the network to be able to update the weights in earlier layers. Therefore, it is needed to identify the gradients of all other parts of the network. Calculating this by hand is a very tedious process and not fit for complex architectures. A better option is using a method called backpropagation. Backpropagation is based on the calculus chain rule. The idea is most easily demonstrated with scalar functions and variables. Suppose that there is a single chain of computations, where $y = g(x)$ and $z = f(y) = f(g(x))$ and the local gradients of the functions $\frac{\partial y}{\partial x}$ and $\frac{\partial z}{\partial y}$ are known. Also the gradient of the calculated loss L has been determined to be: $\frac{\partial L}{\partial z}$. Then using the chain rule, the loss gradient can be calculated for the input y as:

$$\frac{\partial L}{\partial y} = \frac{\partial L}{\partial z}\frac{\partial z}{\partial y} \tag{2.11}$$

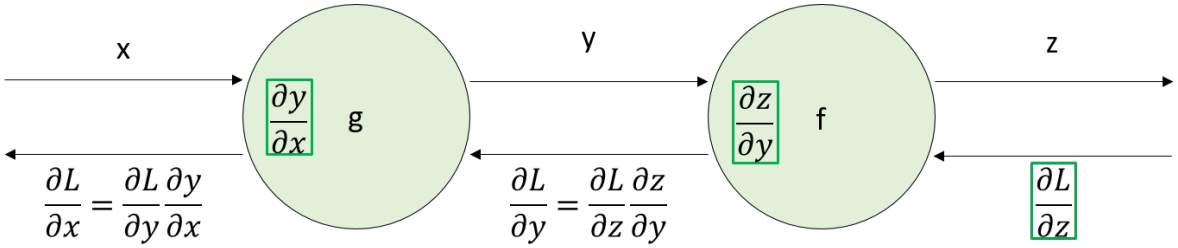


Figure 2.8: Graphical representation of backpropagation. The terms boxed with green are values that are saved during the forward propagation steps.

Propagating this even further back, we apply the same concept to see:

$$\frac{\partial L}{\partial x} = \frac{\partial L}{\partial y}\frac{\partial y}{\partial x} = \frac{\partial L}{\partial z}\frac{\partial z}{\partial y}\frac{\partial y}{\partial x} \tag{2.12}$$

Graphically, this process is a lot more clear and can be seen in Figure 2.8. The top represents the forward propagation through the network according to the defined functions. The local gradients are computed and saved to be used later. Next, the loss is calculated together with the gradient. Finally the loss is propagated backwards through the bottom arrows, using the local gradients.

The chain rule can be extended to the case for input vectors instead of scalars, which is needed in deep neural networks. For this purpose the chain rule can be written as:

$$\nabla_{\mathbf{x}} z = \left(\frac{\partial \mathbf{y}}{\partial \mathbf{x}}\right)^T \nabla_{\mathbf{y}} z \tag{2.13}$$

Where $\frac{\partial \mathbf{y}}{\partial \mathbf{x}}$ is the Jacobian matrix containing all partial derivatives of the input and output vectors, which is multiplied by the straight forward gradient. The same principle can again be used as in the much simpler scalar case. With that, the loss information can be used to update the weights in the network. [21]

2.3.6. Architectures

The MLP is the most intuitive form of a neural network which, according to the universal approximation theorem, can approximate any function with enough layers and parameters [25]. However, for a lot of complex functions this is not the most efficient method. Instead, different network architectures can be used for specific applications. There are many different categories of networks, which specialize in all kinds of different applications such as segmentation, visual recognition and dose prediction. One group of architectures is especially important for the use in dose prediction: The convolutional neural networks.

2.4. Architectures for dose prediction

2.4.1. Dose prediction with convolutional neural networks

The difference between convolutional neural networks (CNN) and normally connected networks is that a convolutional neural network assumes a certain spatial dependence in the network nodes. This makes the network especially useful for images, which always come in grid like structures. As images consist of pixels generally in 3 dimensions (two dimensions and 3 different color channels), the dimensionality of the input is often very high. Even with an image of only 200 by 200 pixels and a single hidden layer neural network, there are already 120000 different weights to be optimized. Therefore, when increasing the depth and the size of the images, the training time increases, which is eventually not feasible anymore.

By assuming a grid like structure within the input, the convolutional neural network can encode for intricate properties, with a more efficient algorithm. A convolutional neural network typically consists of three distinct layers types of layers: The convolutional layer, the pooling layer and the activation layer.

2.4.2. Layers of the convolutional neural network

Convolutional layer

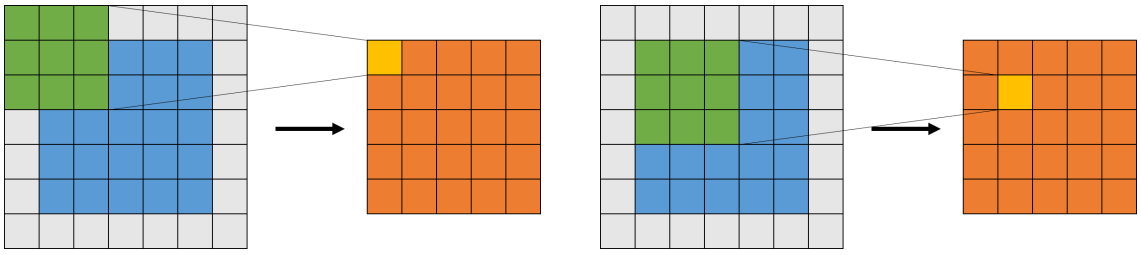
The convolutional layer is the basis of the convolutional neural network. It is based on the mathematical convolution operation:

$$f * g = \int_{-\infty}^{\infty} f(t)g(t-\tau)d\tau \tag{2.14}$$

The convolution integral determines the overlapping value of two different functions. The concept in neural networks is very similar, but is even more intuitive. In a convolutional neural network, a filter with weights of a specific size smaller than the input volume is determined. The filter is moved over the grid of the input image data and for every position, a single output is calculated. These outputs can be combined to give an output array of certain dimensions. Every convolutional layer typically consists of multiple of these equally sized filters. The output volume is determined by four hyperparameters:

- The size of the filter.
- The number of filters, which determines the depth of the output volume.
- The stride of the filter, which is the amount of pixels that the filter moves between consecutive filter steps.
- Zero padding, which is the addition of pixels with a value of zero around the borders of the input volume, with which the output dimensions can be adjusted.

Thus, the number of filters determines the depth of the of the output volume, while the other parameters define the output in the other two dimensions [26]. A graphic example of how a filter would convolve over the input can be found in Figure 2.9a and Figure 2.9b.



(a) (b)

Figure 2.9: The filter, depicted by the green squares, moves over the input volume with the blue squares, with a stride of one. The gray area indicates the zero padding such that the output in orange is from the same size as the input volume. The calculation of the filter on a specific position results in an output value in yellow, which is represented by a different filter position as can be seen from Figure 2.9a and Figure 2.9b.

All different pixels and filter squares contain a certain value, being zero by definition on the zero padded outer ring. During the operation, the filter values are multiplied with the input values, to produce an output value. For this two dimensional example, the expression for the output height and width, which are the size of the output in the first and second dimension respectively, are given by Equation 2.15:

$$S_{out,i} = \left[\frac{S_{in,i} + 2 \times \text{Pad}_i + (\text{Kernel\_size}_i - 1) - 1}{\text{Stride}_i} + 1 \right] \tag{2.15}$$

Here S denotes the size of the input or output and subscript i denotes the value in one of the two dimensions.

In summary, the convolution operation computes very local responses from the input. In this way the network is broken up into little parts which can activate when certain certain structural properties are located within the filter position.

Pooling layer

The second kind of layer that is present in most CNNs is the pooling layer. The pooling layer is a layer that downsamples the input volume based on a metric in part of the input volume. The reason to do this is twofold. First of all through downsampling, the amount of parameters is lowered, which makes it easier to train the

network. On the other hand, it also prevents overfitting: Often the exact position of a pixel is not very relevant, rather it is important to know how different edges are located with respect to each other. Therefore, by adding pooling layers, the network can be made invariant to little changes in the input [21].

The pooling operation is generally done with a filter of size two and stride two as larger filters would destroy too much information at a time. The operation done is similar to the convolution operation as there is also a filter that moves over the input volume. The difference is that the input volume is not multiplied with the filter, but another operation is done within the filter position. The most used operation done in pooling is the max pooling operation, in which the maximum value is retained. This is shown in Figure 2.10. Other operations include, retaining the average value within the filter or the L2 norm, but are used significantly less [26].

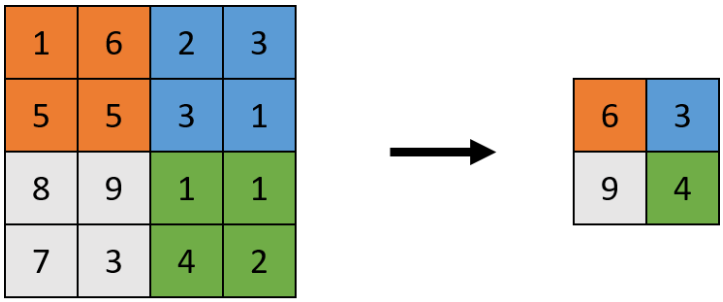


Figure 2.10: Max pooling operation

As an alternative for the pooling layer, it has also been shown that one can simply use more convolution layers, or convolution layers with a larger stride [27]. By excluding separate pooling layers, the network can be simplified a lot.

Activation layer

The last important layer present in all CNNs is the activation layer. This is the same kind of activation layer as seen earlier in the normal fully connected neural network and again has the same purpose of creating a non linearity. The ReLU is again the most widely used kind of activation layer.

Other layers

There are a lot of other layers that can be included in a CNN. One example is the normalization layer. Normalization layers are used to help the optimization process within the training by normalizing the features in mean and variance using parts of the input volume. The optimization is generally faster by including this and can also help in the converging process of deep neural networks. There are several normalization methods, depending on the dimension of the network volume the normalization takes place. For example batch normalization (BN) normalizes over the different number of input elements and group normalization (GN) normalizes over part of the channels. This is also displayed in Figure 2.11.

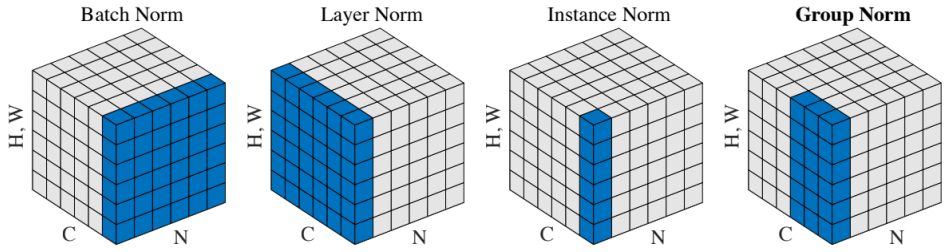


Figure 2.11: Visualization of different normalization methods [28]. H and W depict the height and width of the data, C is the amount of channels and N is the number of samples.

Another example of an often used layer is the skip connection. Skip connections do not process the features in a layer but concatenate the features into another part of the network. This can be useful when wanting to use less processed features later in a network.

Finally there is often an output layer included in the CNN which is dependent on the type of application. For classification problems this is often a fully connected layer. Such a layer is generally not needed when doing voxelwise predictions.

2.4.3. Layer stacking

To finally construct the CNN, all these layers need to be combined. This can be done by stacking the different layers after each other. A typical CNN network consists of alternating convolution and activation layers with a pooling layer after some of the convolution and activation pairs. Depending on the application, different layer stackings or architectures are used.

2.4.4. Examples of dose prediction architectures

In contrast to many applications such as image classification, using a neural network to predict dose needs a voxel to voxel prediction instead of a fully connected output layer to class scores. In a voxel to voxel prediction the output volume contains dose scores that correspond to the same location as the pixels of the input volume. This is needed for the application of dose prediction, as every voxel needs to have a predicted output value. Such a network, that has the same output as input dimensions, is called a fully convolutional neural network.

U-Net

To use a fully convolutional neural network for dose prediction, intricate architectures, made up of the different possible convolution layers, have proven to achieve good results. One of the most important architectures, which laid the basis for most other architectures that are used for dose prediction, is the U-Net, first presented by Olaf Ronnenberger et. al. [29]. This network has primarily been developed for image segmentation, but can also be used for dose prediction [23]. The U-Net also downscales the using pooling, but makes use of upsampling methods to regain the original image dimensions. There are several upsampling methods available. A common way is starting with an unpooling operation, which uses the location of the maximum values in the corresponding pooling operation and places the voxels at these locations in the larger grid. This gives a larger sample with sparse entries as the other locations in the grid are not filled with a value. This operation is followed by a deconvolution to fill the sparse activation map [30]. By doing this multiple times, the original sized output can be generated [31]. Another frequently used option is directly using transposed convolution operations. The architecture of the U-Net is displayed in Figure 2.12. As can be seen from this figure, the name of the network comes from the U shape of the network.

In the first part of the network the path is contracting, while in the second part the path is expanding. Also, the high resolution parts of the left part of the U-Net are used in the upsampled right path via skip connections to retain the information of high resolution. The amount of pooling operations determines the number of levels in the network, which is five in this case. To apply this network to 3D structures, an extension of this architecture to 3D is needed. This was presented by Çiçek, Özgün et. al. [32] [23] [33].

Other architectures

There are several other networks besides this U-Net architecture which have either proven to be suitable for dose prediction or are a specific improvement to the U-Net for dose prediction. Examples are the DenseNet, DoseNet and the HD U-Net. The DenseNet or densely connected CNN, first suggested by Huang et. al., is very different from the U-Net as it does not include pooling/downsampling operations. Instead, the DenseNet has layers which are all interconnected in forward direction. Every layer has the output of all previous layers as input. In this way, information learned in earlier part of the network is preserved better. As a result, fewer parameters are needed as redundant feature maps are not needed to be relearned [34].

DoseNet, first developed by Kearney et. al., is an architecture that is very similar to the normal U-Net. However, the difference is that the residual block on every level of the U-Net is not only concatenated to the other side of the U-Net, but also to the convoluted block deeper in the network. In this way, more detailed information can be propagated to deeper parts of the network [33].

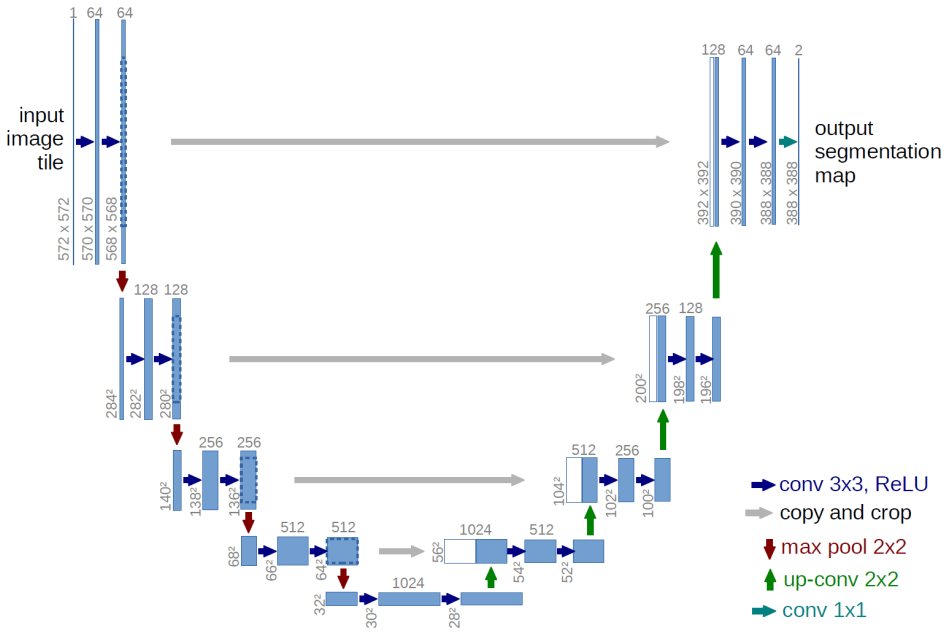


Figure 2.12: U-Net for image segmentation [29]. The arrows represent the different operations done. Each box represents a feature map and the arrows indicate different operations. The vertical number depict the image dimension and the horizontal numbers represent the number of channels

Finally, the hierarchical densely connected U-Net (HD U-Net), is a combination of DenseNet and a normal U-Net, first designed by Nguyen et. al. [23].

All these architectures use the same input arrays and train on the same loss functions, but are more complex. Also the performance varies compared to the simple U-Net architecture. The DenseNet does not generally preform better compared to the normal U-Net, however, DoseNet, and the HD U-Net have been shown to do so [23] [33].

2.5. Physics guided neural networks

Neural networks have proven to perform well at dose prediction within a patient. However, the prediction from deep learning does not incorporate any physics whatsoever as the network learns to predict a generalized result, by incorporating all information of the training data. This implies that the predicted dose distributions are neither realistically achievable or physically correct by default as the underlying physical principles are not part of the prediction.

Recently, a new area within deep learning has emerged in which prior physics knowledge is used in the training process of the network. This can help tackling the problem of the lack of underlying information in neural network prediction, which thus helps improving the prediction. Also, another advantage is that training can often effectively be done with smaller datasets [35]. This field is called physics guided neural networks (PGNN), or physics informed neural networks (PINN).

2.5.1. Basic principles of physics guided neural networks

There are generally two methods to use the physics information within a neural network: The first and simpler method is to make a hybrid physics-data (HPD) model [36]. In a HPD, the training input samples are enriched with input from a physical simulation. Where a neural network tries to predict output Y from input X as $Y = f_{NN}(X)$, HPD includes Y_{PHY}, an output from physical simulation model f_{PHY}, into the training as input: $Y = f_{NN,PHY}(X, Y_{PHY})$.

The second option is to incorporate the physics within the loss function. Instead of minimizing a simple

loss with one component $L(f(X), Y)$, the loss now consists of two terms:

$$L_{tot} = L(f(X), Y) + L_{PHY}(f(X)) \tag{2.16}$$

Thus, a prediction which does not comply with the physical solution is penalized. It therefore acts as a regularizer for the neural network. Apart from improving the prediction,

2.5.2. Dose engines

Dose calculated using a physics based model can provide the input for a HPD approach for radiotherapy. There are several methods that are currently used to calculate a physical dose. The two most important are Monte Carlo methods and convolution methods.

Monte Carlo methods

The Monte Carlo method is a dose calculation method that uses random number and statistical analysis in order to solve the Boltzmann transport equation. The transport equations describes the transport of particles, in this case photons, in a phantom. The Boltzmann equation is in most cases unsolvable analytically, but an approximation of the solution can be simulated using the Monte Carlo method. The idea is to track individual particles which undergo interactions according to a certain probability distribution based on physics. By simulating a large amount of particles and tracking the average behavior, a solution to the Boltzmann equation can be approximated.

Although Monte Carlo methods produce very good results, the method is computationally expensive and is therefore not directly useful in dose calculations of individual patients yet. As the limits of computation change over time, Monte Carlo can become usable on a daily basis in the future.

Convolution methods

A method that is more useful for dose calculation in patients is a convolution method. This method is based on the notion that each source photon has the same energy deposition effect on average. Therefore, if this distribution or deposition kernel is known, the resulting dose distribution can be calculated for a specific photon energy deposition rate at a certain point in the phantom. The dose deposition kernel is dependent on the photon energy. After calculating it once, using for example Monte Carlo, the results can be used indefinitely. However, the phantom composition in which the kernel is calculated, usually water, should be taken into account.

Apart from the dose deposition kernel, the total energy distribution is dependent on the energy released per unit mass at different points in the body. The total energy released per unit mass (TERMA) is a measure for this amount of energy deposited in a certain point in the tissue. It can easily be calculated in the body with Equation 2.17 using the attenuation in the phantom:

$$T(d) = W_{beam} \frac{\mu}{\rho} e^{-\mu d} E, \tag{2.17}$$

where T is the TERMA value, W_{beam} is the beam intensity, ρ the density of the material, d the distance traveled in the medium, μ the attenuation coefficient and E the energy of the photons. Finally both the TERMA and also the kernel need to be scaled for non homogeneities in the phantom for a better approximation.

With the kernel and the deposition values within the body, the kernel can be convoluted over all dose points to calculate a proper dose distribution, which is still an expensive process. Approximations such as the collapsed cone approximation are methods to account for this problem in practice as they reduce the complexity of the calculation [37], [38].

By using one of the methods for dose calculation, an approximate dose distribution including physics information can be calculated for a patient. With such a physics based approximate distribution, input for the HPD model is generated. The information of the physics based dose engine can then be used as prior to the dose prediction model.

3

Method

In this chapter, a detailed overview will be given on the methods used during this research to acquire the results. In Section 3.1 to Section 3.3, an overview will be given on the used data and deep learning libraries. Next, in Section 3.4 the structure based dose prediction model will be extensively discussed. Finally in Section 3.5 to Section 3.8, a novel multi stage PGNN learning approach that uses a dose engine and segment prediction for dose prediction will be discussed.

3.1. Patient and plan data

The available dataset for this project consists of data of 100 different patients with prostate cancer provided by The Netherlands Cancer Institute (NKI). The treatment plans for the patients were generated using the Pinnacle^3 autoplanner, resulting in a homogeneous dataset of patients with a prescribed dose of 60 Gy to the prostate planned in 20 fractions. All patients were planned to be treated with volumetric modulated arc therapy (VMAT), on an Elekta machine with a nominal photon energy of 10 MeV and a 80 leaf MLC. A homogeneous dataset can be important for model training, especially when not a lot of data on outlier patients is available for the model. As a result, not all patients can be included in the final dataset. Most importantly, patients with a bowel loop have a bowel which is much bigger and very close to the PTV. This results in dose distributions that differs a lot from patients without the bowel loop. Therefore these patients are excluded from the dataset. For the same reasons, patients with hip implants are also excluded from the data. Moreover the prescribed PTV dose is also checked to be homogeneous over the dataset. Lastly, it is checked that all patient data is planned on a treatment machine with the same characteristics. For example patients planned on a treatment machine with 160 MLC leafs are excluded. The remaining filtered dataset contains 89 patients in total.

For each patient the plan (RTPLAN), structure set (RTSTRUCT) and dose (RTDOSE) information is available in Digital Imaging and Communications in Medicine format (DICOM). The structure contours, available from the RTSTRUCT files have a resolution of 0.5 mm in every direction, while the dose grid has a lower resolution of 4 mm as retrieved from the DICOM files. The resolution is the same in every direction. The CT images are not included in this study. As all patients are shaped differently, the dose grid and the structure maps vary in size between patients. The dimensions of the grid vary between a minimum of 100 x 69 x 52 and a maximum of 138 x 104 x 63 in voxel size over all patients.

Five different structures are included in this study. The different structures taken as input are: PTV, rectum, rectal wall and anal sphincter and the entire body volume. The bladder is not included in this study as NKI does not consider the bladder as an OAR for prostate patients. The left and right femur heads are excluded to save memory, because these structures have an insignificant clinical effect on the treatment plan as the objectives for the left and right femurs are easily met. The structure contours are projected onto the coarser dose grid to form three dimensional Boolean maps with the same dimensions as the dose, and to be able to use them as input for the model. Every structure map is added as a different channel in the fourth dimension to create the input for the neural network.

3.2. Input preparation

Before the data from the DICOM files is ready to be used as input for the neural network, different operations need to be done. PyTorch is used as framework for the deep learning, which will be more extensively covered in Section 3.3. PyTorch requires so called tensors as input, which can be easily generated from NumPy arrays [39] [40]. Thus, the DICOM files need to be transformed into NumPy arrays. Apart from that, the tensors need to be consistent in size and shape for training of the neural network. Lastly, to enhance the dataset, data augmentations are done by modifications on the NumPy array.

3.2.1. Data loading modules

For the first step of the input preparation, specific python modules developed by NKI are used to acquire the dosegrid and the structure masks from the DICOM files. The modules, called AVS modules, have not been modified and are considered as the correct way to acquire the dose and the structure sets projected on the dose grid. The structure set also has a resolution of 4 mm as a result.

3.2.2. Resizing

There are several requirements for the input of the deep learning model. First, the model requires a standard input size of the structures, because the model dimensions and the amount of channels are predefined. Consequently the output dose is also of standard size. Also, the dimensions of the 3D image are halved with every pooling operation in the neural network. Therefore it is beneficial to have image dimensions that are multiples of $2^{(k-1)}$, as the dimensions will then remain integers while downscaling with pooling operations. Here k is the number levels in the network, which is five in this case. Next, it would be preferred to lose as little information as possible during resizing. Lastly, the memory needed for training is influenced by the size of the input. Modifying the input size to match the standardized size can be done in various ways:

- Image manipulation
- Cropping around the isocenter.
- Zero padding

With image manipulation packages such as SciPy or openCV, the dosegrid can be easily reshaped into the preferred size [41] [42]. However this requires to do an interpolation operation between voxels. This can alter the dose information, which is not preferable. Cropping can only reduce the size of the input image. Therefore the standardized size needs to be smaller than the dimensions of every individual input array. This makes it prone to information loss errors. The last option is zero padding, in which the arrays are padded with zeros until it matches the standardized size. The disadvantage of zero padding is the redundant use of memory.

In this project a combination of the last two methods is used: Cropping and zero padding. A standard size is first determined in which is made sure that there is no loss of information. Taking the previous considerations into account, the standardized input size is set to 144 x 96 x 64 voxels. Every patient is then either cropped or padded, depending on the initial size of the dosegrid of the patient. In this way, all information is preserved, while there is less excessive memory usage in training. Schematically, this is shown in Figure 3.1.

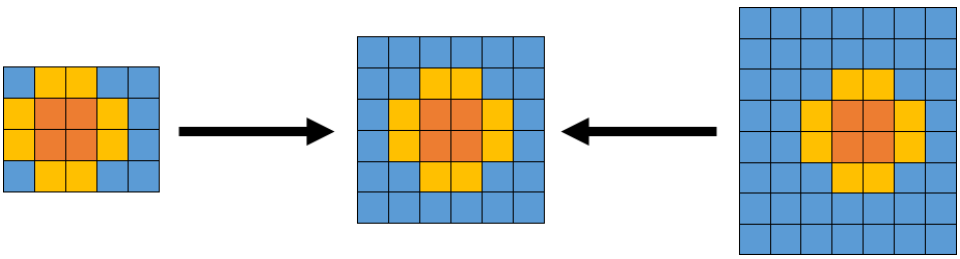


Figure 3.1: Example of resizing in 2D. In the resizing operation, smaller arrays are padded with zeros, depicted by blue squares, and larger arrays are cropped.

3.2.3. Data augmentation

When more data is available, the neural network is better able to generalize training data trained for a new prediction. Therefore, data augmentation is done to enhance the dataset of 89 patients. The data of every patient is modified in various ways to produce more input samples for the neural network. Transformations that retain physically correct dose information include translations and rotations. Other transformation as for example mirroring and scaling do not. Mirroring can counteract biases from the autoplanning system or the radiation direction, while scaling can have an influence on the path length of the radiation beam. All augmentations need to be calculated both for the dose and the structures. Creating local anomalies in the structure shape is therefore not an option, as an anomaly in the structure cannot be correctly transformed to the dose distribution. Translations and rotations are included in this study. The translation is done in either of the three dimensions by translating the data voxel wise. Translating in this way avoids pixel value errors as a result of interpolation. A visual representation of a voxelwise translational augmentation can be seen in Figure 3.2. Besides the translations, rotational augmentations of up to two degrees in the axial plane are included as well. The rotations are done using the Python SciPy package[41].

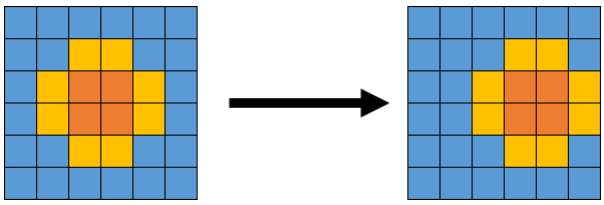


Figure 3.2: Example of voxelwise translation in a 2D representation.

To prevent the excessive use of storage, the augmentations are calculated in every training sample individually. This increases the training time, but the increase is not significant.

3.3. PyTorch

The framework in which the deep learning architecture is built is PyTorch. PyTorch is a machine learning library that is designed to be usable in the same object oriented programming style as Python. This makes it very easy to use, while also providing competitive performance compared to other libraries [39]. PyTorch holds all variables for the neural network in the tensors. Tensors are similar to arrays in NumPy and can be modified using the same operations such as addition and multiplication. The special thing about Torch tensors is that they can be easily moved from the CPU to the GPU. Since optimizing parameters of the neural network does not require difficult operations, but rather a lot of simple operations, the GPU is a better fit for the task. As such, running the neural network on the GPU makes training a lot faster.

PyTorch also makes training a neural network easy. In the neural network, the parameters need to be updated by backpropagation. Instead of having to write the entire backpropagation algorithm manually, backpropagation can be done automatically. Two parts of PyTorch play an essential role in this automation: the computation graph, which tracks operations, and the automatic gradient calculation.

3.3.1. Computation graphs, the autograd module, and backpropagation

The computation graph is simply a method to track what operations are done on the tensors during the forward pass through the network. Without such a graph, it is untraceable what operations have been done within the network. In that case, an individual backpropagation algorithm needs to be manually defined to propagate information backwards through the network. To overcome this, a tensor can be given the require_grad = True statement. This lets PyTorch collect all necessary information for backpropagation, which it stores in the computation graph.

The module that PyTorch uses to create these graphs is called the Autograd module. The goal of this module is to make backpropagation as easy as possible. For the specific points in the created graph, the Autograd module also automatically calculates the local gradients. These local gradients are needed to backpropagate the information and thus update the parameters. With both the computation graph and the automatic gradient calculation together, backpropagation can be easily done without the need to manually define it. As a result, the parameters can be updated with two lines of code.

3.4. Dose prediction with deep learning

3.4.1. Architecture specifics

In this project a U-Net is used for dose prediction [29]. For dose prediction, 3D structure data is used as input. Therefore, the original U-Net is modified to a 3D convolutional U-Net. This has already been done in many instances in literature. The network used in this project is inspired by the standard U-Net used by Nguyen et. al. [23]. The architecture is similar, however the input dimension and amount of channels differ. In this study, five input channels following the different structures are taken into account. As there are less structures in the prostate region compared to the head and neck region studied in the article of Nguyen, the amount of channels in each layer is halved. The U-Net that is used is schematically shown in Figure 3.3

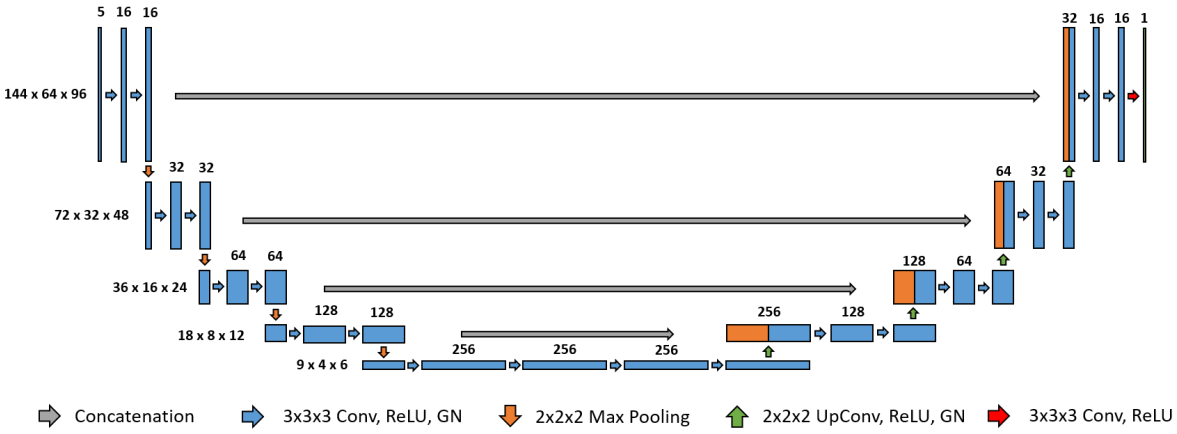


Figure 3.3: Schematic view of the U-Net used for dose prediction. The rectangles represent the feature maps and the different colored arrows indicate the different operations with set filter sizes. Conv is a convolution operation, ReLU is the ReLU activation function, GN is a group normalization and UpConv is a transposed convolution operation. Finally, the values above the rectangles indicate the amount of channels and the values in front of the rectangles display the tensor dimensions.

As can be seen from Figure 3.3, the network has five levels. In the contracting left side of the U-Net, each block at a certain depth consists of two 3D convolution operations with stride one and a padding to ensure output dimensions equal to the input dimensions. The convolution layers are followed by a ReLU as activation function and a group normalization (GN) operation to speed up the learning process [10]. GN is used instead of the more common batch normalization (BN) as memory restrictions prevent the batch size to be large enough for accurate normalization statistics. GN is an alternative for this [28]. With the exception of the input layer, the first convolution layer doubles the amount of channels. The second convolution layer keeps the amount of channels equal. After a complete convolution block, a copy is saved to use as skip connection to the expanding path, to retain detailed feature information. The output of the block is then fed into a max pooling operation, halving the layer dimensions and increasing the depth of the network. In total this process is repeated four times, creating a network of depth five, with a total number of channels of 256 in the deepest part of the network.

On the right side of the network, the network is expanding again, starting with an upsampling operation, which is a 2x2x2 transposed convolution with stride two. After the upsampling, the features from the contracting side are concatenated to the upsampled features. Lastly, again a double convolution block is done before upsampling again. The upsampling process repeats until the depth reaches the initial value and a last output layer is included, constructing a single output from the 16 remaining channels.

3.4.2. Loss functions

The loss function generally used for dose prediction model training is the mean squared error loss (MSE), as for example in the studies of Nguyen et. al., Kearney et. al. and Kandalan et. al. [23] [33] [43]. The MSE loss is given by:

$$L_{MSE}(D_{pr}) = \frac{1}{N}\sum_{n}(D_{pr,n} - D_{tr,n})^2 \tag{3.1}$$

Where the loss is given by L_{MSE}, N is the amount of voxels, $D_{pr,n}$ is the predicted dose for individual voxel n and $D_{tr,n}$ is the true dose for individual voxel n. The MSE loss has proven to be effective and is suitable for

voxel wise prediction.

To possibly improve the performance of the model, modifications to the MSE loss have been investigated. One simple modification to the MSE loss is the addition weights to different voxels, depending on the structure they belong to. As it is more important to have an accurate prediction for the OARs and the PTV for clinical applications, voxels belonging to these structures can be assigned a higher weight. This weighted MSE loss function is then given by:

$$L_{WMSE}(D_{pr}) = \frac{1}{N}\sum_{n} w_n (D_{pr,n} - D_{tr,n})^2 \tag{3.2}$$

The weights w_n can be arbitrarily chosen, but it should be taken into account that the volume of the PTV and the OARs is much smaller than the volume of the entire body. Therefore, the weights in the PTV and OAR might need to be higher with a factor similar to the volume difference to see an actual effect of adding weights.

The last modification investigated in this study is based on the fact that not all prediction errors within structures are equally important, depending on the application. If a model is used to detect all situations where underdosage in the PTV might be present, it would be better to have a model which has a bias to underprediction of the target than overprediction. In such a case, an overprediction error would be more severe than an underprediction error as you might not catch all possible outliers. Based on this a loss function can be constructed with different weights depending on the prediction value, using a Heaviside function. This alternative will be indicated as Heaviside loss and is described as:

$$L_H(D_{pr}) = \frac{1}{N}\sum_{n}\Big[w_{1,n}(D_{pr} - D_{tr})^2 H(D_{pr} - D_{tr}) + w_{2,n}(D_{tr} - D_{pr})^2 H(D_{tr} - D_{pr})\Big] \tag{3.3}$$

Where the Heaviside step function H is one or zero depending on whether there is over- or underprediction. Again the same holds for the weights as within the weighted MSE: PTV and OAR size are of influence on the weight effect.

The different weight values used for the loss functions in the deep learning prediction, including weights compensating for the structure size, are summarized in Table 3.1.

Structure	Weighted MSE	High weighted MSE	Heaviside MSE	High weight Heaviside MSE
Body	1	1	1	1
PTV	8	100	5 & 10	50 & 100
OAR	4	50	3 & 6	30 & 60

Table 3.1: Weight values used in the different structures for the weighted MSE and Heaviside MSE objective functions

3.4.3. Training and evaluation

Of the in total 89 patients available in the data, 64 were used for training and 13 were used for validation. The other 12 were set aside as test data. Before training, the weights within the model were initialized as random values with uniform distribution on [-1, 1] with the Xavier uniform initialization. Training is done in loops over the training data called epochs. The training method itself is based on the approach of Kearney et. al., where the training is split in multiple phases [33]. In this study the training is split in two phases. In the first phase an epoch consists of a single patient with all different augmentations. In the second phase the augmentations are excluded and the network is trained on all training data. The idea behind the two phases of training is that in the first phase the model learns to deal with possible augmentations and produces a first guess, while in the second phase the model learns to generalize the prediction for different patients and dose distributions. By not including the augmentations in the second phase, the training can be done much faster.

In both phases, the model was trained for a number of epochs until the validation loss did not decrease any more. This early stopping is done to prevent overfitting of the model. During training the Adam optimizer is used, with a learning rate of $\alpha = 1 \cdot 10^{-3}$. Training is done in batches of only one sample at a time to prevent memory problems on the GPU machine. The machine used for training contains a NVIDIA GeForce GTX-750 GPU with 2GB RAM and was used for all neural network training purposes.

Part of the evaluation of the prediction will be done using several dose statistics. Dose statistics are used to evaluate the quality of the treatment plan in practice, but can also be used to compare predictions with different loss functions among each other. Besides, they can be used to compare with dose prediction models and corresponding dose statistics from earlier studies. Several dose statistics are used within this study. First of all there are PTV coverage statistics, such as the D95 and D98, which represent the minimum dose that 95% and 98% of the PTV volume receives respectively. Next, the maximum dose and the mean dose in both the PTV and the rectum are used. For the rectum, the V45, the volume fraction that gets at least 45 Gy dose, is also considered as useful evaluation statistic, because of the clinical relevance in plan acceptance. Lastly, the homogeneity index (HI) and van 't Riet conformation index (CI) are compared [44]. For the homogeneity index, many different formulations exist [45], but the one used here follows the HI from Nguyen et. al. [10] and is given by:

$$HI = \frac{D_2 - D_{98}}{D_{50}}, \tag{3.4}$$

where D_2 is the minimum dose that the 2% target volume with the highest dose receives. Similarly, the D_{98} is the minimum dose for the highest 98% of the target volume and D_{50} for 50% of the target volume. The CI is given by Equation 3.5 [44]:

$$CI = \frac{(V_{PTV} \cap V_{100})^2}{V_{PTV} \cdot V_{100}}, \tag{3.5}$$

where V_{PTV} is the volume of the PTV and V_{100} is the volume that receives 100% of the prescription dose. The $\cap$ represents the intersection between the V_{PTV} and the V_{100}.

Furthermore, the dose volume histograms (DVH) of several structures within the body are predicted to give a further indication of the clinical accuracy as plans and corresponding dose distribution are often rated on their DVH during the planning phase.. In a DVH, the relative volume receiving at least a certain dose is plotted for distinct organs. This gives an indication of the quality of the 3D dose distribution within a 2D plot. It also distinguishes predictions in different structures.

Finally, NKI earlier developed a knowledge based planning approach to predict the rectum DVH. The model uses principal component analysis (PCA) to predict the rectum DVH, depending on only two DVH points, the V95 and the V_{mean}, which are predicted using the patient overlap volume histogram (OVH). The accuracy of the rectum DVH from the structure based dose prediction can be compared to this simpler model, which can be a good indication of the clinical usability of the deep learning prediction.

3.5. Physics guided neural networks

The hybrid physics-data approach for a PGNN is investigated to see if including physics information within the prediction can lead to better prediction results. For the HPD model, a dose distribution that includes physics information would be the extra input to the prediction model. However, this dose distribution can only be based on the structure set that is also provided as input to the structure based dose prediction network, as that is the only available prior information.

A dose distribution is a combination of contributions from individual segments. Therefore, the total dose distribution can be recreated by reproducing the individual segments. As all patients are treated on the same treatment device, the treatment angles, number of MLC leafs and number of control points are equal. The control points are discretized points at which the cumulative relative beam weight and the MLC positions are exactly known for the VMAT arc. The relative treatment intensity per control point and the MLC positions are the only two variables that are not fixed. As a result, when both the MLC positions and the relative intensity are known, a dose distribution can be calculated for each segment using a fast physics based dose engine. The MLC positions and the relative intensities are approximated by again using a neural network based on the patient structures.

3.6. Segment prediction

Obviously, predicting segments is no easy task as the MLC positions are very variable and there are many degenerate MLC position solutions for an optimal dose distribution. Besides, when one is able to accurately

predict individual segment MLC positions with weights, there is no need for a dose prediction algorithm, as the dose can be reconstructed directly from the correctly predicted MLC positions. The idea in this segment prediction is therefore not to predict 100% accurate segments, but to predict segments that are accurate enough to produce an approximately correct dose distribution, which still contains useful physics information.

To predict segments, a U-Net based neural network with structure information as input is again used. However, the output should now be a two dimensional contour of the MLC opening for every different beam angle and a relative weight of the segment.

3.6.1. Input and output data

The shape of the segments are roughly dependent on the PTV and OAR structures which lie around the axis of the treatment direction. The shape of a 2D projection of the tumor is dependent on the beam angle from which it is viewed and thus on the angle of the beam. It would be preferred to irradiate the tumor from directions where no OAR is in front or behind the tumor. Therefore, the structure information is mostly relevant seen from beams eye view (BEV) for every radiation angle.

As input for the neural network BEV projections of the PTV and OARs are used. The projections are made by first rotating the structure information in the axial plane to BEV using the rotation algorithm also used in the data augmentation part. The amount of structure voxels from BEV are then summed to acquire a 2D grid with the thickness of the different respective structures at each part of the projection. As only parts directly around the PTV are relevant, the 2D project is cropped to a 64x64 window around the isocenter in the PTV. Finally, an inverse gantry head rotation is applied to make it easier to reconstruct the MLC positions later on. The resulting input is now an array of 64x64 BEV projections for every beam angle, containing individual structure information as separate channels.

The output of the network is an equally sized 64x64 window with weighted voxel values corresponding to the MLC contour and relative weights. The relative weights are derived from the maximum value of the output window for every angle. Normalized over all segments this gives the relative weight for every individual angle. Every pixel value is then normalized on [0,1], again using the maximum value, from which the MLC contour is finally derived. A schematic overview of the pipeline from input to output is given in Figure 3.4

3.6.2. Architecture specifics

There are several U-Net based architectural options for the implementation of the segment prediction network. The most obvious method would be to use a 2D U-Net with the different BEV projections as different channels. The different structures would be added in the same way as RGB information is processed in 2D convolution. However, individual sequential segments are correlated to each other since the MLC positions are limited in movement between two control points. Besides, the projected structures in BEV are comparable between consecutive segments as the rotation between the segments is limited to only 4 degrees. Because of this correlation, another option is to use a 3D U-Net again. For this 3D U-Net, the consecutive segments would be located in the third dimension, while the structures are represented in the channels. Both methods are investigated to find the best preforming method.

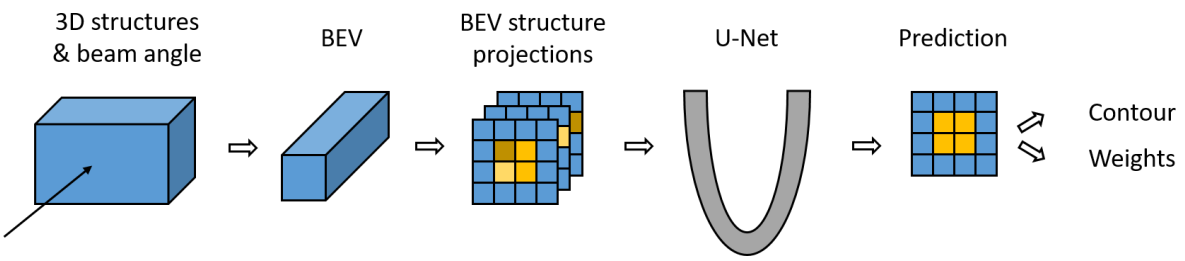


Figure 3.4: Pipeline of the MLC contour prediction. Starting with the 3D structure and a beam angle, giving a contour and relative weights as output.

The networks have the same structure as the dose prediction U-Net given in Figure 3.3. One difference for both networks is the different input and output dimensions, which have the shape of the 64 x 64 BEV projection. Apart from that, the only difference between the structure dose prediction network and the 3D segment prediction network is the amount of channels. The amount of channels in the fifth level of the network is 128 instead of 256. For the 2D architecture, there are a bit more differences. The network has a large amount of input channels, starting at 256 channels in the first layer. Apart from that, the network only has three levels, as the large amount of channels take up too much memory to include more.

The prediction does not directly deliver a Boolean contour map, but a weighted probability map that voxels belong to the contour or not. To create a contour, first the relative weight is extracted by the maximum value. After normalization of all pixels, the pixels with a value larger than a predefined constant are assigned as part of the contour. The value of this constant is on [0,1] and is calculated as the value that creates the best DICE agreement with the actual contour, averaged over all validation data. The DICE statistic is given in Equation 3.6 [46].

$$DICE(X) = \frac{2|X \cap Y|}{X + Y} \tag{3.6}$$

Here X and Y are the volumes of the predicted and the actual MLC contour respectively.

3.6.3. Loss functions

Also several loss functions have been tested to investigate what achieves the best results. First of all, the basic MSE loss was used on the 64x64 MLC contour weighted with the relative weights. Apart from that, a more intricate and task specialized combination of the binary cross entropy loss (BCEloss) with a softmax function and the DICE loss have been used together. This is based on the loss functions described by Sudre et. al. and Milletari et. al. [47] [48]. The BCEloss is given in Equation 3.7:

$$BCELoss(X) = -\frac{1}{N}\sum_{n=1}^{N} r_n \log\big(p(X_n)\big) + (1 - r_n)\log\big(1 - p(X_n)\big), \tag{3.7}$$

with r_n the true voxel values and $p(X_n)$ the softmax values of the predicted voxel contour. For the BCEloss, the output layer of the neural network has two layers: one to encode the the MLC contour with nonzero values and one inverted prediction which encodes the parts outside of the MLC contour with nonzero values. The DICE loss is given by Equation 3.8:

$$DICELoss = 1 - \frac{2|X \cap Y|}{X + Y} \tag{3.8}$$

The difference between this DICE loss and the original DICE statistic is that a value of 0 is considered to be a good agreement within the loss function, as the loss function obviously needs to be minimized. This is opposed to the value of 1 being a perfect agreement in the normal DICE statistic. The DICE works well for larger segments, but for smaller contours the statistic becomes more volatile as small contour displacements can already have big effects on the outcome. The final loss is calculated by summing the DICE loss with the BCE loss.

In total three different models with different architectures or loss functions have been trained. A 2D U-Net model with MSE loss, a 3D U-Net model with MSE loss and a 3D U-Net model with the combined DICE and BCE loss.

3.6.4. Training and evaluation

Training the segment prediction network was very similar to the training of the dose prediction network. The same training, validation and test split of the patients is done, which again means 64 training, 13 validation and 12 test of the 89 total patients. The parameters of the network are again randomly initialized according to a uniform distribution on [-1,1]. Training is done in just one phase, with the entire dataset present and no augmentations. Also, again a learning rate of $1 \cdot 10^{-3}$ was used. Finally, the model was trained on the same GPU machine.

The segments are evaluated by looking at the overlap of the predicted contour with the actual contour using the DICE statistic, earlier given in Equation 3.6.

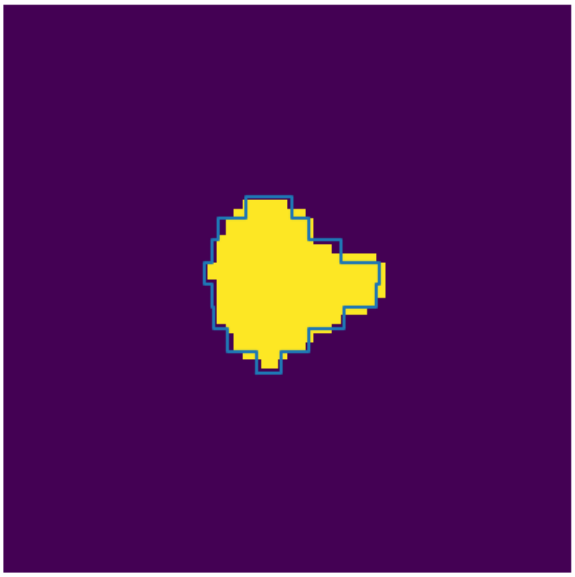


Figure 3.5: Example of post processed MLC contour in blue over the predicted MLC opening

3.6.5. Post processing

After predicting the projection of the MLC map on a 64x64 voxel grid, a post processing step needs to be done to convert the map into realisable MLC positions. This is done by using the prior knowledge of the location and width of the MLC leafs in comparison to the isocenter. The vertical position of the MLC edges is therefore fixed and the horizontal position of the leaf edges can be interpolated from the contour on the 64x64 grid. An example of such a post processing done on an actual predicted contour can be found in Figure 3.5. From this example it can be seen that indeed the MLC opening roughly follows the shape of the predicted contour. The last step is the application of the gantry head rotation on the MLC edge positions. The MLC positions can then be used as input for the dose engine.

3.7. Segment extraction and dose engine

The RTPLAN contains all machine parameters of a treatment plan, which can be used to reproduce the dose to the patient per segment. The VMAT plan consists of two beams, with 70 different control points, each four degrees apart. The VMAT beam itself is continuous, but is discretized with the control points in 140 parts. The parameters that are important for the reconstruction of the final dose can be put into two categories. First of all, there are the parameters that are available without a dose calculation. These parameters include the location of the isocenter, the individual voxel locations, the distance from the source to the isocenter and the beam angles. These retrievable parameters are also summarized in Figure 3.6. The second category of parameters are the parameters that are not known before the treatment planning has been executed. These are the MLC positions and relative weights at the different control points, and are predicted using the method described in Section 3.6.

To reproduce the dose per segment, an approach similar to the approach of Cho et. al. [37] is used, which consists of three main steps. First, it is determined which voxels are hit by the photon beam for every segment and corresponding MLC positions. Next, the total energy released per unit mass (TERMA) is calculated. Lastly, a collapsed cone convolution is done between the calculated TERMA distribution and dose deposition kernels from literature to account for scatter.

3.7.1. Segment extraction and hit checking

The first step is the determination of the position of the MLC leafs and blocks, which shape the photon beam. When not using predicted MLC segments, the positions of the central part of the MLC can be directly extracted from the RTPLAN. Together with the leaf width and leaf length, the edge positions can be calculated, from which a contour is extracted that follows the edge points. The final contour is then rotated taking the

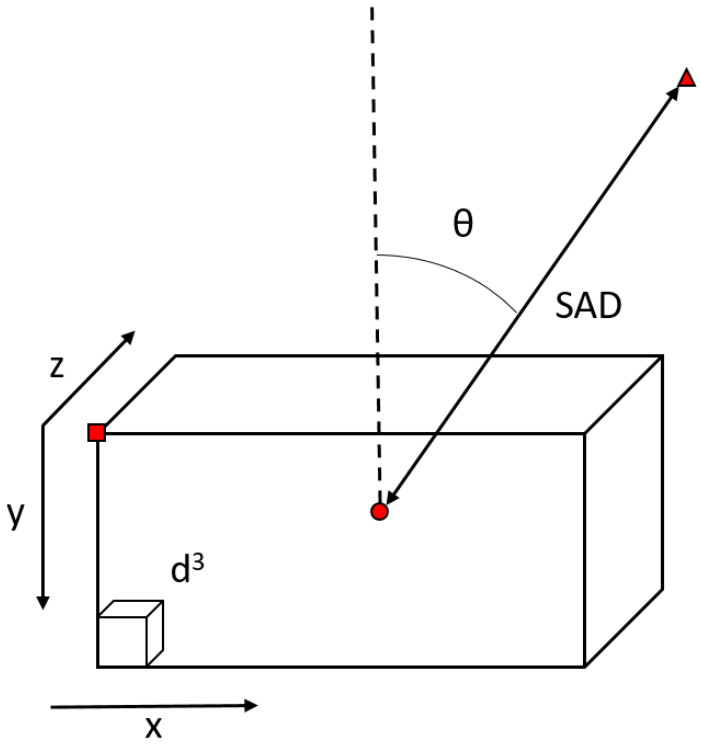
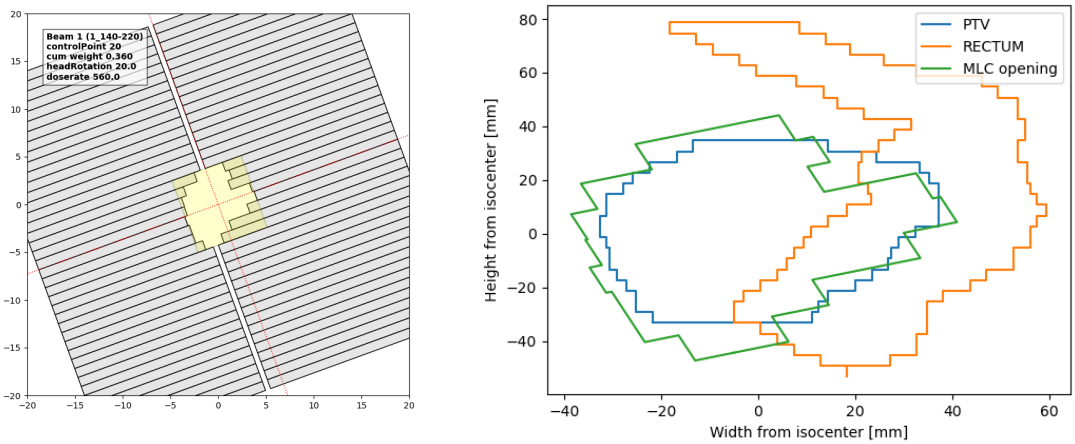


Figure 3.6: Schematic view of the parameters which can be retrieved for a patient. The red square represents the position of the left upper voxel, the red circle represents the isocenter while the red triangle depicts the source position. Furthermore, the beam angle, rotating around the isocenter is given by θ. SAD is the source axis distance, which is a constant value. Finally, the voxel size can be retrieved, here given by d^3, with a value of 4 mm in every dimension.



(a) MLC opening for same patient and beam, distance in cm. (b) Projection of random patient structures

Figure 3.7: Projection of random patient from beamseye view with corresponding MLC positions

gantry head rotation into account. As no virtual source point for the radiation was available, the approximation is made that the incoming radiation is parallel to the source isocenter axis. Every part of the parallel photon beam falling within the MLC contour goes through to the patient, while every part outside the contour does not. An example of the MLC positions for a patient from a specific beam angle can be found in Figure 3.7a. The corresponding projection of the MLC opening on the relevant structures can be found in Figure 3.7b.

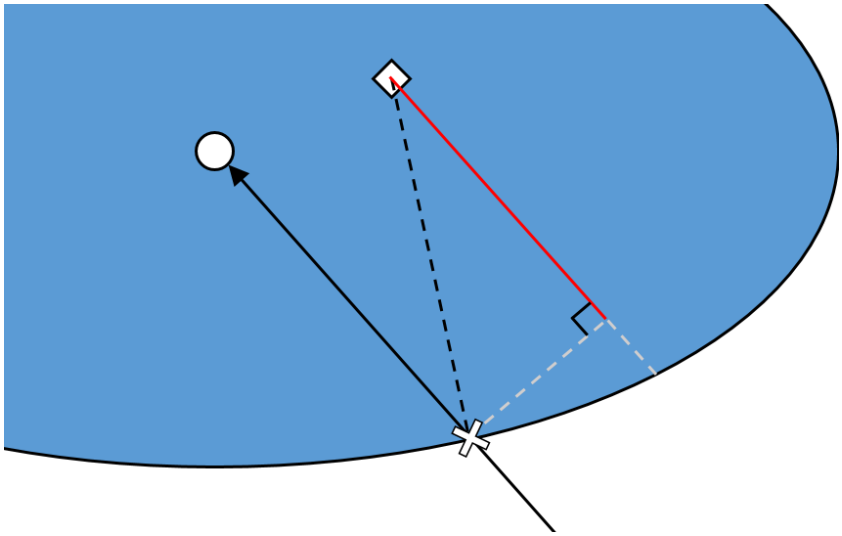


Figure 3.8: Schematic view of the entry point approximation. The circle is the isocenter, the diamond shape is the voxel center at $\vec{r}$ and the cross represents the entry point $\vec{r}_{entry}$. The black arrow is the beam from the source to the isocenter or $\vec{I}$. The black dashed line represents the vector from voxel center to the entry point $(\vec{r}_{entry} - \vec{r})$. Finally, the red line visualizes the approximation of the entry distance $\hat{I} \cdot (\vec{r}_{entry} - \vec{r})$, clearly in the direction of source isocenter axis.

Next, this MLC contour can be used to calculate which voxels in the patient are hit by the photon beam from a certain angle. This is done with a hit checker script. This script calculates the location of the center of the checked voxel on a plane perpendicular to the source isocenter axis. If the location of this voxel center is within the opened MLC contour, projected onto the perpendicular plane, the voxel is hit. If the location is outside of the contour, the voxel is not hit.

3.7.2. Calculation of released energy in the body.

After determining which voxels are hit for a certain angle and MLC positions, the TERMA values needs to be calculated for these voxels. As seen earlier in the theory and specifically Equation 2.17, the TERMA is a function of the distance traveled through the tissue and the photon attenuation in the tissue, again given in Equation 3.9.

$$T(d) = W_{beam} \frac{\mu}{\rho} e^{-\mu d} E \tag{3.9}$$

Thus, to calculate the TERMA for a voxel, the distance travelled through the patient body to that voxel needs to be calculated. As it proved computationally too expensive to raytrace every voxel to find the exact entry point in the body, the distance is approximated by the distance to a constant entry point on the center line between the isocenter and the source location, compensated for the parallel beam direction. The exact calculation is summarized in Equation 3.10:

$$d(\vec{r}) = \hat{I} \cdot (\vec{r}_{entry} - \vec{r}) \tag{3.10}$$

In the approximation the entry point r_{entry} is the entry point of the source to isocenter vector, $\vec{I}$. This is calculated once using a raytracing algorithm and is then used as reference point for the entry points of all voxels locations. The inner product is taken between the unit vector $\hat{I}$ and the vector between the voxel location and the reference entry point. This approximates the distance in the direction of the photon beam. A schematic view of this process is given in Figure 3.8. From the figure can already be seen that the approximation is not perfect, especially when the beam does not enter the body perpendicular. However, the calculated TERMA value is more accurate then using only the entry point.

The hitchecking process and TERMA calculation are done simultaneously per beam and per angle. The final result is a TERMA distribution of all individual beams summed.

3.7.3. Photon dose kernel and collapsed cone convolution.

The next step of the dose calculation is to use a photon dose kernel, which is an expression of the energy deposition around the position where an incoming photon deposits its energy and thus accounts for the scatter. The kernel is generally calculated using a Monte Carlo approach. The dose kernel can be convolved over the TERMA values to produce a physically correct dose distribution. For the photon dose calculation kernel, a tabulated analytical approximation from Ahnesjö for a polyenergetic kernel in water is used [49]. The analytical description reads:

$$h_\Theta(r) = \frac{A_\Theta e^{-a_\Theta r} + B_\Theta e^{-b_\Theta r}}{r^2} \tag{3.11}$$

Where r is the distance to the source point, and Θ the angle to the photon direction. Furthermore, the A_Θ, B_Θ, a_Θ and b_Θ are tabulated constants that can be derived with Θ.

One problem with this method is that practically all individual TERMA voxels will contribute to the dose in every voxel in the patient. Therefore, simply convolving the dose kernel over all TERMA voxels would lead to an immense amount of computations. To counteract this, a collapsed cone approximation is often used within practical dose calculation settings. The collapsed cone approximation is an approximation where the energy that is normally released into equally sized coaxial cones is transported linearly over the voxels in the axis of the cone instead of through the entire cone. This reduces dimensionality and therefore the computation time a lot. The new collapsed cone kernel also called the differential kernel is given by Equation 3.12 [38].

$$k^\Omega(r) = \int_{\Omega^{m,n}} r^2 h_\Theta(r) \sin(\theta) d\theta d\phi \tag{3.12}$$

For the collapsed cone approximation, the azimuthal angle ϕ has been divided into 24 cones, while the altitudinal angle θ has been divided into 12 parts. As the voxel size of 4 mm is fairly large, using this kernel directly for the convolution operation would not yield the most accurate results [38]. Instead of using the normal collapsed cone kernel $k^\Omega(r)$, the cumulative collapsed cone kernel $K(r)$ is used for more accurate results. The cumulative kernel can be calculated by simply integrating the differential kernel as in Equation 3.13 [38]:

$$K^\Omega(r) = \int_0^r k^\Omega(t) dt \tag{3.13}$$

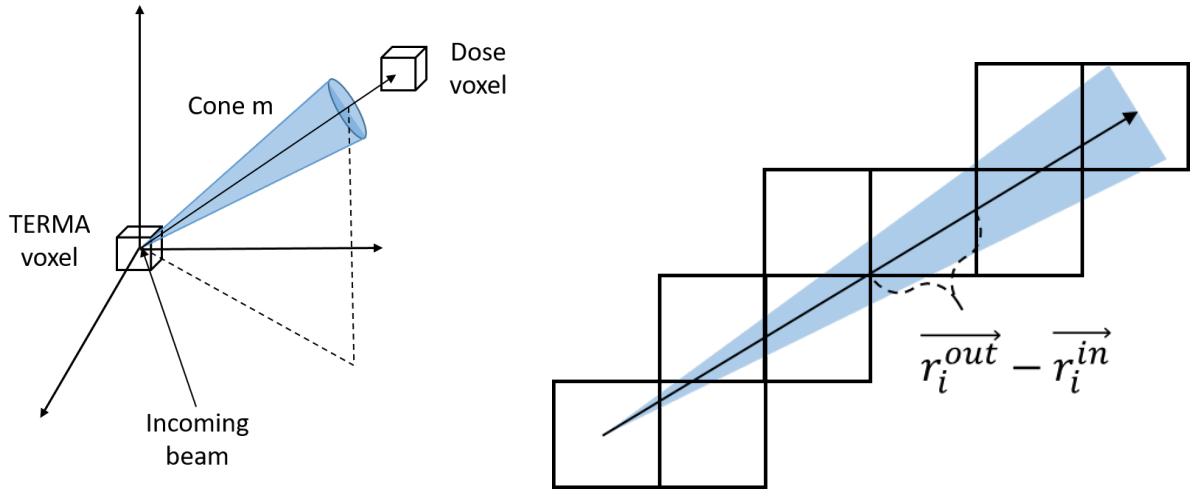
The dose in a certain voxel can now be calculated by adding all contributions from the cones as in Equation 3.14

$$D(\vec{X}) = \sum_{\Omega=1}^{M} \sum_{i=0}^{N} T(d_i) \left[K^\Omega(|\vec{X} - \vec{r}_i^{\,out}|) - K^\Omega(|\vec{X} - \vec{r}_i^{\,in}|) \right] \tag{3.14}$$

Here M is the number of cones and N is the number of voxels traversed by the axial cone line $\vec{r}$. K^Ω is the cumulative kernel value for the specific cone. r_i are the individual voxels on the axial cone line, with the superscript denoting the entrance and exit of the voxel. Finally, $\vec{X}$ denotes the voxel in which the dose is calculated and consequently $|\vec{X} - \vec{r}_i|$ is the distance from the dose voxel to the point on the cone line. To decrease the computational load even more, only a fixed number of 10 traversed voxels per cone are considered within the approximation. This assumption is made since the contribution of voxels decreases very fast with r, resulting in insignificant dose contributions. A schematic example of this collapsed cone approximation process can be seen in Figure 3.9. The final distribution corresponding to the TERMA beam is now calculated, again per beam, as the kernel shape is dependent on the orientation of the beam.

3.8. Multi stage learning

Using the segment prediction and the simple dose engine, an approximation of the actual dose distribution as input for the hybrid model is calculated. The final step is to retrain the prediction model with the predicted dose and the structure sets. The neural network is now trained to improve the dose distribution from the segment prediction as well as making a prediction based on the structures. The complete hybrid model now consists of various steps, with two neural networks. Such a multi stage learning approach is similar to the multi stage segmentation approach for small organs of Zhou et. al. [50]. Schematically, the entire model



(a) Visualization of one of the cones m from a TERMA voxel as result of scatter of the incoming beam. The axial line of cone m intersects the dose voxel.

(b) Visualization of a 2D cone path through several voxels, with vectors $\vec{r}_i^{in}$ and $\vec{r}_i^{out}$, used as input for the cumulative kernel calculation.

Figure 3.9: Visualization for the collapsed cone algorithm, images inspired by Cho et. al.[37]

structure is shown in Figure 3.10.

3.8.1. Dose prediction

The last neural network used for dose prediction is again very similar to the neural network displayed in Figure 3.3. The only architectural difference is one extra input channel which contains the normalized dose distribution from the dose engine. The dose has been normalized on the max dose per patient in order to let the neural network predict the level of the dose, similar to the structure prediction network. Besides the network architecture, another difference is that only one loss function is used here: the MSE loss. The MSE loss is chosen because it is the most basic loss function and makes comparisons with literature easier. Apart from these small differences, the rest of the training method is completely equivalent.

The final dose prediction algorithm can learn to produce an output based on two different inputs. First, it can learn the same information from the input structures as the normal prediction model does. Besides, it can learn a mapping from the input dose to the output dose. To investigate what the network actually learns, and to detect possible bottlenecks, the model is also trained using just the predicted dose as input, as well as with only the correct segments. Finally three models are trained. A model with both structures and predicted segment dose as input, a model with predicted segment dose as input and finally a model with a dose distribution based on the correct segments as input. These models will be referred to as combined segment model, segment model and correct segment model, respectively. The models will again be compared by the DVH statistics and by visually comparing the dose distributions.

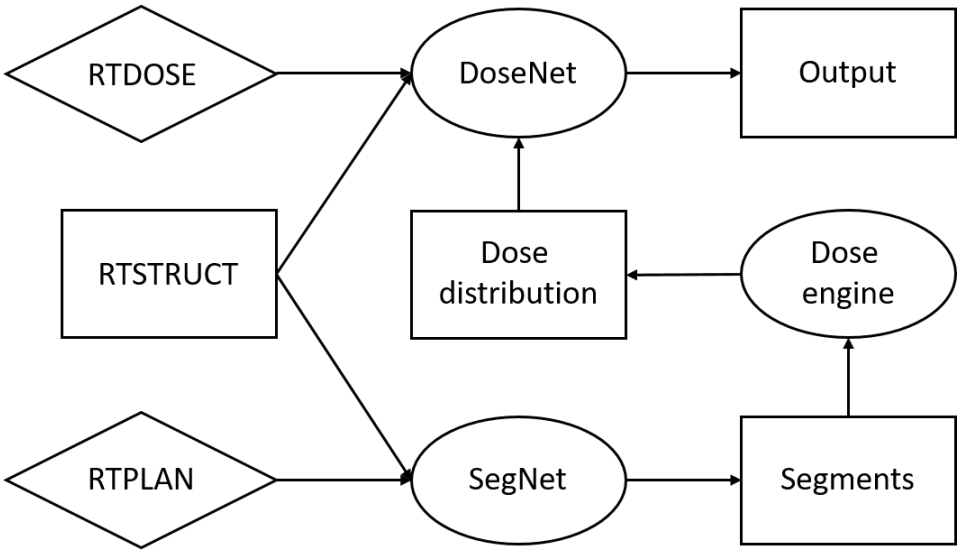


Figure 3.10: Schematic overview of the multi stage learning process. Rectangles represent data which is also used as input for the prediction, diamond shapes represent data only used for training and ovals represent a module such as a neural network (SegNet and DoseNet) and the dose engine.

4

Results

In this chapter the results of the simulations will be presented. In Section 4.1 the results of the structure based dose prediction model are shown. In Section 4.2, the performance of the simple dose engine is discussed. Next, in Section 4.3 the results of the segment prediction will be presented. Lastly, in Section 4.4 the results of the multi stage dose prediction model are discussed.

4.1. Structure based dose prediction

Training of all models has been done in two stages, in which the model was trained until an optimal validation loss was reached. Training in the first phase took about 250 seconds per epoch, while training in the second phase took about 520 seconds per epoch. All models with corresponding loss functions had a different amount of epochs to train before the optimal validation loss was achieved. The values of the validation loss vary since the loss functions are different and therefore the results cannot be compared directly. All the training characteristics have been summarized in Table 4.1.

Table 4.1: Training characteristics of the models, with the amount of epochs both training phases take and the final validation loss.

Loss function	**Epochs, first phase**	**Epochs, second phase**	**Validation loss**
MSE	6	37	0.77
Weighted MSE	15	43	0.97
High weight MSE	8	13	4.16
Heaviside MSE	9	51	0.94
High weight heaviside MSE	12	36	2.36

An example of the loss during the entire training process is given in Figure 4.1. This specific example is of the two phases of the weighted MSE model. It can be seen that in the first phase the validation loss barely decreases. Training and validation losses for models with other objective functions are summarized in Appendix A.

A typical example of the output of the dose prediction model for a patient from the test group is given in Figure 4.2. This particular example summarizes the prediction of a model with MSE loss together with the contour masks, the original dose distribution and the dose difference. Summaries of dose prediction models with different loss functions can be found in Appendix B. It stands from the examples that the predicted dose outside of the PTV is visually uniform.

4.1.1. Dose prediction characteristics

The performance of the models can be measured and compared to each other using various dose characteristics. The average absolute percentage dose difference of the PTV dose coverage statistics of the models, are summarized in Table 4.2. The same difference in the rectum dose statistics, together with the conformity index and the homogeneity index is given in Table 4.3.

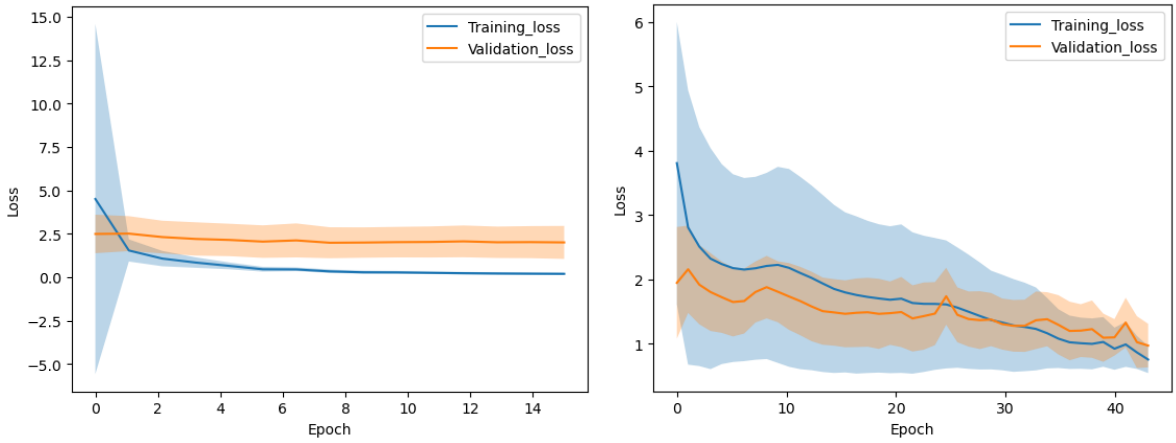


Figure 4.1: Example of the loss during training in two different phases of the WMSE loss function, with the training loss and the validation loss. The shaded area gives the standard deviation of the loss.

Table 4.2: Average absolute % dose difference of several PTV coverage statistics. The average difference is calculated over the patients of the test set, just as the standard deviation.

	Average Absolute % dose difference: **100** $\left\lvert\frac{D_{true}-D_{pred}}{D_{pres}}\right\rvert$			
	PTV coverage statistics			
	D95	**D98**	D_{max}	D_{mean}
Loss function	Mean ± SD	Mean ± SD	Mean ± SD	Mean ± SD
MSE	1.87 ± 1.64	2.04 ± 1.86	2.77 ± 1.82	1.26 ± 0.92
Weighted MSE	1.42 ± 1.26	1.35 ± 1.47	2.43 ± 1.01	0.99 ± 0.67
High weight MSE	4.24 ± 3.42	4.82 ± 3.93	1.93 ± 1.67	2.68 ± 1.77
Heaviside MSE	1.51 ± 1.60	1.91 ± 1.76	3.90 ± 1.65	2.00 ± 1.43
High weight heaviside MSE	2.62 ± 2.12	2.79 ± 2.45	2.83 ± 2.56	2.27 ± 1.94

Table 4.3: Absolute average % dose difference of several rectum coverage statistics, Conformity index and Homogeneity index. The average difference is calculated over the patients of the test set, just as the standard deviation.

	Average Absolute % dose difference: **100** $\left\lvert\frac{D_{true}-D_{pred}}{D_{true}}\right\rvert$				
	Rectum coverage statistics, CI, HI				
	D_{max}	D_{mean}	**V45**	**CI**	**HI**
Loss function	Mean ± SD	Mean ± SD	Mean ± SD	Mean ± SD	Mean ± SD
MSE	1.19 ± 0.91	5.64 ± 3.17	7.79 ± 4.89	15.64 ± 10.67	4.35 ± 2.37
Weighted MSE	1.81 ± 0.99	3.39 ± 2.47	3.50 ± 2.47	13.80 ± 8.38	3.42 ± 2.19
High weight MSE	2.59 ± 1.71	4.27 ± 2.55	5.14 ± 2.81	31.67 ± 13.93	5.65 ± 3.03
Heaviside MSE	1.69 ± 1.26	3.25 ± 2.62	2.77 ± 2.54	17.66 ± 13.15	5.16 ± 0.99
High weight heaviside MSE	2.31 ± 1.94	3.81 ± 2.24	5.21 ± 4.40	20.30 ± 14.18	3.60 ± 2.45

From the statistics in Table 4.2 and Table 4.3 it can be seen that overall the weighted MSE model has the lowest average absolute dose difference within almost all categories. Only the PTV D_{max} and rectum D_{max} statistics are significantly lower within the high weight MSE and the normal MSE respectively. Another important observation is that the model with high weights underperform compared to the other models. Increasing the weights in the organs to compensate for the size of the organ does therefore not lead to an improvement in the model, while a small weight increase for the MSE does. Finally it can bee seen that the CI has very high difference values, up to over a 30% difference. This is the case for all different loss functions. The WMSE does outperform the other loss functions also for the CI and the HI, although the difference values are still very high especially for the CI.

As the weighted MSE seems to outperform the other models, it is used to assess the performance of the dose prediction more extensively. The normal MSE is also included in the assessment as it is the loss function most

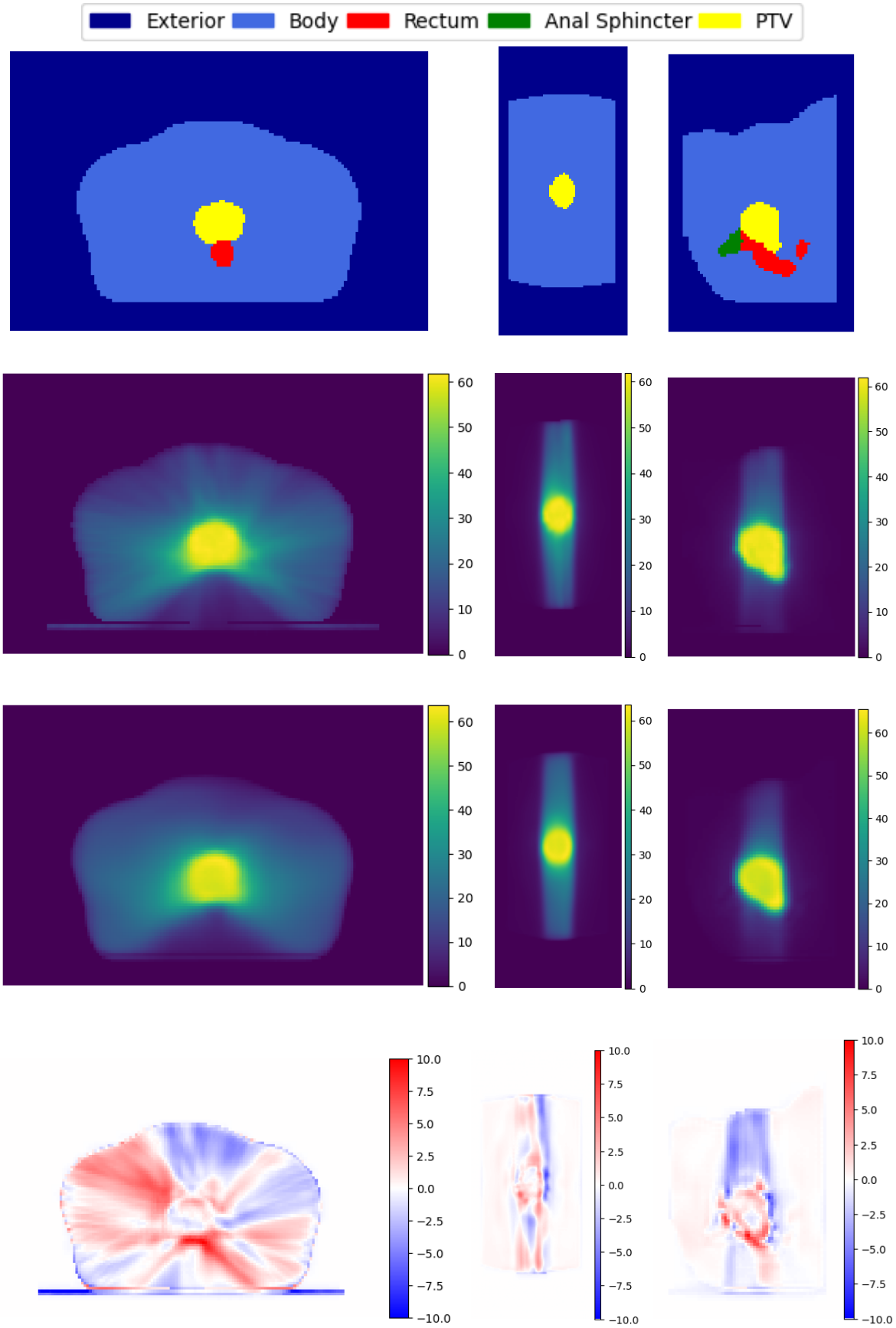
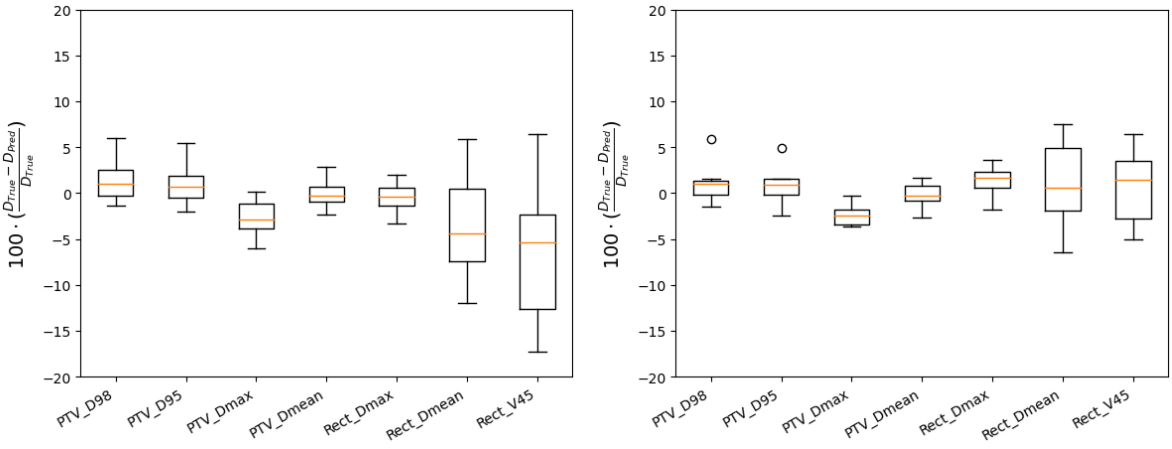


Figure 4.2: Summary of dose prediction model with MSE loss function from the axial, rotated coronal and sagittal view respectively. First row represents the positions of the contours, second row is the true dose, the third row is the prediction and the last row contains the dose difference. Doses and differences are plotted in Gy. Overprediction in the dose difference plot represents a higher prediction compared to the truth value.

generally used in literature.



(a) Boxplot of MSE model

(b) Boxplot of weighted MSE model

Figure 4.3: Boxplot of average absolute dose difference of the models with MSE and WMSE as loss function.

For this, first a box plot is made to further visualize the prediction spread of the models, given in Figure 4.3. From the plot can be seen that the spread in the MSE model is generally higher compared to the WMSE model, which again gives the indication that the WMSE model performs better. Both models seem to predict the dose characteristics well, with a spread generally around 0, with the exception of the rectum D_{mean} and rectum V45 of the normal MSE model. These statistics both have a negative bias.

4.1.2. Clinical accuracy

The clinical accuracy is especially relevant when using the dose prediction to help make treatment decisions. The DVH is an important indicator for clinical accuracy of a prediction on patient level. An example of a DVH plot based on the model predictions for a test patient is given in Figure 4.4. It can be seen that the WMSE model is generally closer to the actual DVH curve for the PTV and rectum, indicating that for this patient the WMSE model produces a better prediction.

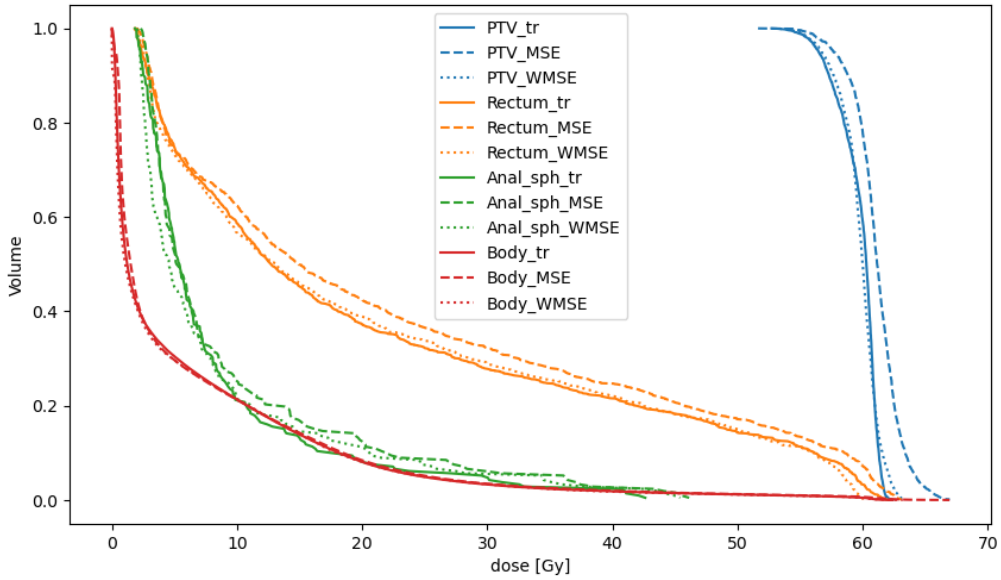


Figure 4.4: DVH plot with MSE and WMSE models for PTV, rectum, anal sphincter and body. The solid line indicates the true DVH, the dashed line indicates the MSE model and the dotted line represents the WMSE model.

DVH scores

For a more quantifiable assessment of the DVH, the absolute average DVH difference over all DVH points for the different structures is calculated for all patients in the test set. The results are presented in Table 4.4. From the table can be seen that on average, the DVH accuracy of the PTV is approximately equal for both models, with an average difference of 0.87 Gy and 0.84 Gy for the MSE and WMSE respectively, which is an error of about 1.5% of the PTV prescription dose.

For the rectum and anal sphincter, the average DVH prediction accuracy differs between the two models. The rectum is predicted more accurately by the WMSE model, while the anal sphincter is predicted better with the MSE model. Furthermore, it stands out that the variation in the MSE model for the PTV and OARs is significantly bigger compared to the variation in the WMSE. However, in the entire body the variation is bigger in the WMSE model.

DICE scores

Another indication of the clinical accuracy of a model can be given by the overlap of isodose volumes of the prediction with the isodose volumes of the actual dose distribution. This is shown in Figure 4.5 where the DICE is plotted for different isodose volumes in the PTV. Here it can be seen that isodose volumes of the MSE model overlap better in the lower isodose volumes, while it is the other way around in the higher isodose volumes. Overall, the DICE score has an average value of 0.91 for both the MSE and the WMSE and has a dip around 40% isodose volume.

Table 4.4: Individual average absolute DVH dose difference per patient for the PTV, the rectum, the anal sphincter and the entire body. The values represent the average difference per DVH point p. The standard deviation indicates the variation in DVH point difference in the patient. An average calculation of these values is included at the bottom of the table.

		Average Absolute dose difference: $\frac{1}{N}\sum_{p=0}^{N}\|D_{true,p} - D_{pred,p}\|$			
		Quantified DVH difference (Gy)			
		PTV	**Rectum**	**Anal Sph.**	**Body**
Patient	Model	Mean ± SD	Mean ± SD	Mean ± SD	Mean ± SD
1	MSE	0.55 ± 0.79	0.96 ± 0.69	0.30 ± 0.27	0.20 ± 0.11
	WMSE	0.39 ± 0.53	0.72 ± 0.66	0.66 ± 0.44	0.37 ± 0.41
2	MSE	0.36 ± 0.37	1.66 ± 1.09	0.67 ± 0.37	0.26 ± 0.23
	WMSE	1.08 ± 0.61	0.85 ± 0.43	0.97 ± 0.41	0.91 ± 0.73
3	MSE	1.31 ± 0.40	1.13 ± 1.00	1.36 ± 1.10	0.52 ± 0.49
	WMSE	0.49 ± 0.25	0.85 ± 1.22	1.00 ± 0.93	0.43 ± 0.65
4	MSE	0.52 ± 0.38	0.86 ± 0.98	0.66 ± 1.48	0.25 ± 0.16
	WMSE	0.51 ± 0.31	0.57 ± 0.54	1.13 ± 0.80	0.17 ± 0.15
5	MSE	1.67 ± 0.83	0.95 ± 0.74	0.94 ± 1.77	0.59 ± 0.62
	WMSE	0.63 ± 0.35	1.04 ± 0.61	0.96 ± 1.16	0.68 ± 0.73
6	MSE	0.64 ± 0.38	1.27 ± 0.95	0.54 ± 0.77	0.43 ± 0.40
	WMSE	1.60 ± 0.27	0.67 ± 0.53	1.02 ± 0.88	0.51 ± 0.69
7	MSE	1.39 ± 0.63	1.97 ± 1.70	1.14 ± 1.75	0.25 ± 0.23
	WMSE	0.25 ± 0.18	0.56 ± 0.43	0.96 ± 1.07	0.14 ± 0.10
8	MSE	0.22 ± 0.22	0.64 ± 0.67	0.73 ± 0.89	0.50 ± 0.61
	WMSE	1.03 ± 0.40	1.11 ± 1.00	0.62 ± 0.55	0.46 ± 0.72
9	MSE	1.06 ± 0.49	1.49 ± 1.12	0.75 ± 0.72	0.26 ± 0.18
	WMSE	0.55 ± 0.27	1.83 ± 1.33	1.51 ± 0.63	0.33 ± 0.34
10	MSE	1.73 ± 0.79	2.49 ± 1.71	0.87 ± 0.40	0.20 ± 0.28
	WMSE	1.23 ± 0.91	0.88 ± 0.58	1.17 ± 0.43	0.49 ± 0.98
11	MSE	0.36 ± 0.25	0.38 ± 0.36	0.48 ± 0.36	0.35 ± 0.52
	WMSE	0.54 ± 0.36	0.41 ± 0.25	0.92 ± 0.46	0.28 ± 0.31
12	MSE	0.58 ± 0.58	1.53 ± 1.22	0.47 ± 0.73	0.17 ± 0.12
	WMSE	0.85 ± 0.39	1.48 ± 1.09	0.76 ± 0.46	0.35 ± 0.37
Average	MSE	0.87 ± 0.51	1.28 ± 1.02	0.74 ± 0.88	0.33 ± 0.33
	WMSE	0.84 ± 0.40	0.91 ± 0.72	0.97 ± 0.69	0.43 ± 0.52

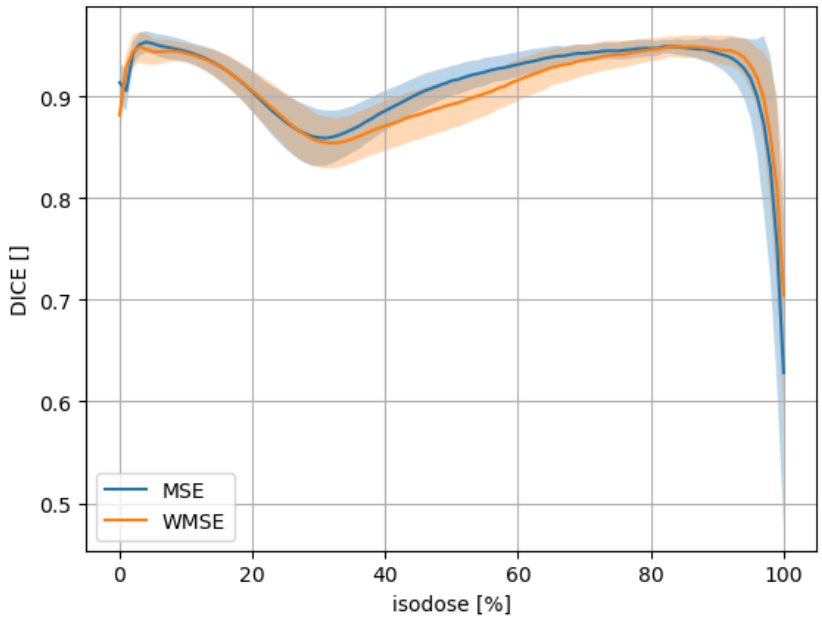


Figure 4.5: DICE plot for different PTV isodose volumes for the MSE and weighted MSE predictions of the test patients. The shaded area indicates the standard deviation of the average DICE prediction on the test set.

Performance compared to NKI model

Of both the OVH based prediction and deep learning prediction, some examples of the rectum DVHs of the test set with respect to the actual rectum DVH have been plotted in Figure 4.6. Here it visually seems that the structure based dose prediction produces a better DVH prediction.

The performance can be quantified by looking at the differences in DVH performance. As the OVH model predicts the volume fraction for a fixed dose, the deep learning prediction is interpolated to do the same. The average absolute volume fraction difference of the OVH prediction is 0.072 ± 0.042, while the deep learning prediction has an average absolute volume fraction difference of 0.020 ± 0.014. Taking both the quantified results and the visual inspection of the rectum predictions for all individual patients into account, the deep learning prediction gives a better rectum DVH prediction than the OVH prediction based model.

4.2. Dose engine performance

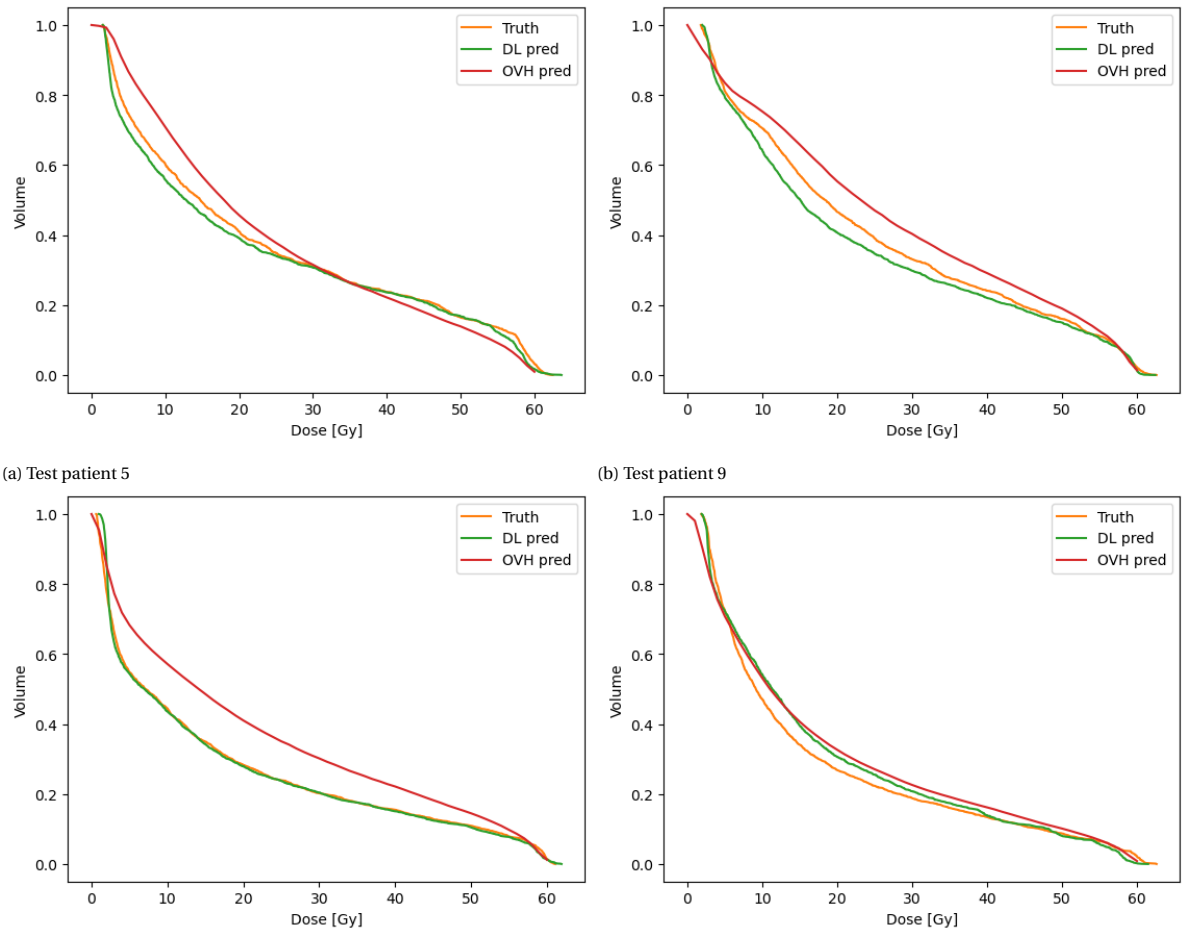
The results of the accuracy of the dose engine are based on the correct segment information retrieved from the RTPLAN. The dose engine produces two outcomes. The first part predicts the TERMA values of voxels hit by a beam from a certain angle through a MLC segment and the second part produces a collapsed cone convolution.

4.2.1. TERMA distribution

An example of a predicted TERMA per voxel for a single beam can be seen in Figure 4.7. It is clear from Figure 4.7c that the shape of the segment largely agrees with the shape of the PTV and the shape of the rectum. Also, the calculated beam is well aligned with the center of the PTV.

Combining all TERMA values from the individual beams produces a preliminary dose distribution, given in Figure 4.8. Note that the presented figures contain the dose distribution normalized on the maximum voxel value. In comparison to the actual dose distribution from the RTDOSE data, given in Figure 4.9, the beam shapes resemble the original dose distribution quite well: The low and high intensities in the dose distribution are located at the same position. This suggests that the individual segment decomposition works well.

Larger differences can be seen in the PTV area of the dose distribution where the intensity is clearly lower in the TERMA prediction compared to the original dose. Apart from that the beams in the TERMA distribu-



(a) Test patient 5

(b) Test patient 9

(c) Test patient 11

(d) Test patient 12

Figure 4.6: Rectum DVH predictions of a subset of test patients with the deep learning based prediction model and the OVH based prediction model.

tion are sharper, due to the lack of scattering dose taken into account. Calculating this TERMA distribution takes approximately 6 minutes with the current scripts.

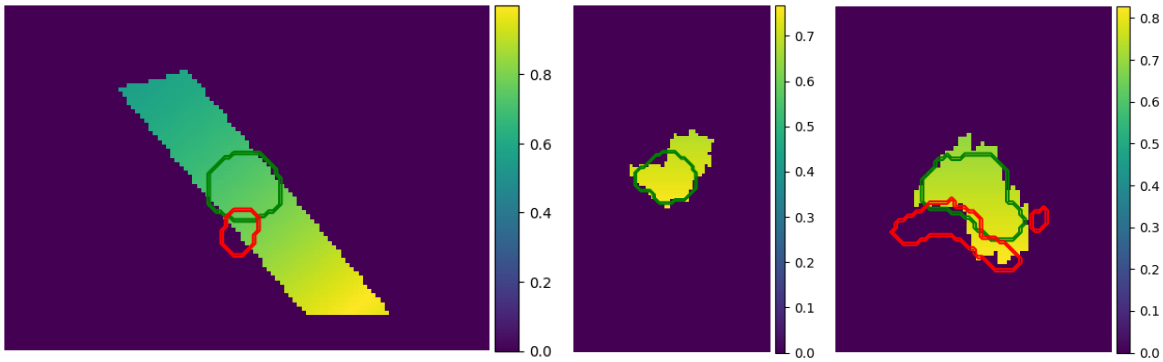
4.2.2. Collapsed cone convolution

The collapsed cone convolution is done on the individual beams to include scatter in the dose distributions. The effect of the collapsed cone convolution operation done on the predicted TERMA beam is given in Figure 4.10. It can be seen that at the sides of the beam, the dose now gradually decreases instead of directly, which is a result of the collapsed cone convolution which imitates the scatter from the beam.

Combining the beams yields again a dose distribution, which is given in Figure 4.11. Compared to Figure 4.9 the PTV intensity complies much better with the true dose distribution than the TERMA distribution. Also, the rays of the single beams are less sharp, while still visually distinguishable. However, they are also less sharp than in the original dose distribution. A disadvantage of including the collapsed cone convolution is the increase in computation time. By including the collapsed cone convolution, the calculation time increases to 30 minutes per patient.

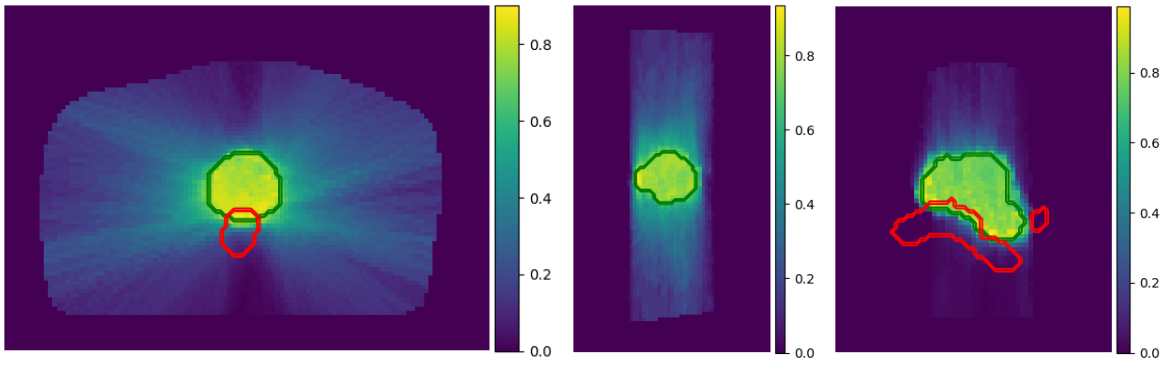
4.2.3. DVH comparisons

The accuracy of the dose engine is further investigated by comparing the DVH plots of the TERMA distribution and the collapsed cone distribution with the actual DVH. This can be seen in Figure 4.12. It stands out that in both the TERMA model and the collapsed cone model the relative dose is generally lower for all structures than the actual value. This can be an effect of the normalization. A second thing that stands out is



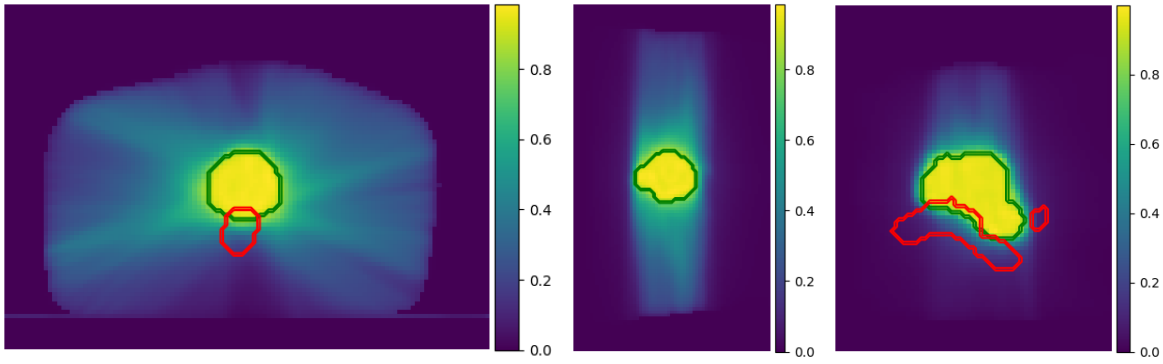
(a) Axial view (b) Coronal view (c) Sagittal view

Figure 4.7: TERMA of a single beam through a corresponding segment. The green contour is the PTV, the red contour is the rectum.



(a) Axial view (b) Coronal view (c) Sagittal view

Figure 4.8: TERMA distribution combining all predicted TERMA beams. The green contour is the PTV, the red contour is the rectum.



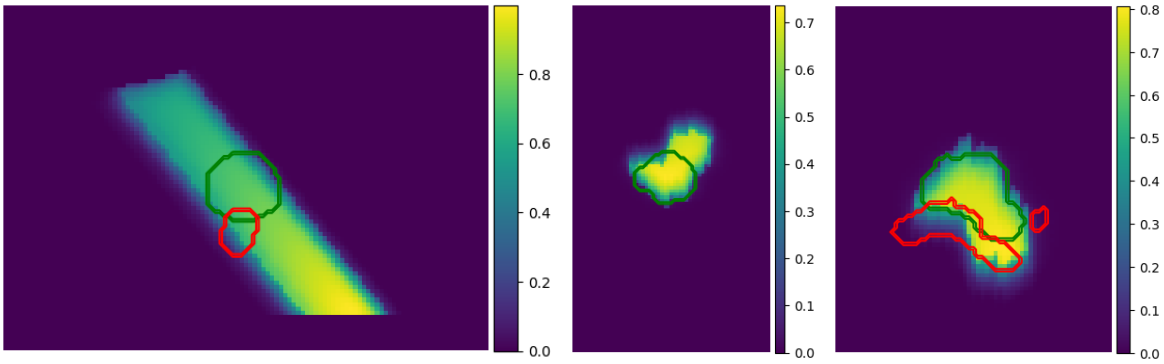
(a) Axial view (b) Coronal view (c) Sagittal view

Figure 4.9: The true dose distribution, extracted from the RTDOSE. The green contour is the PTV, the red contour is the rectum.

that the shape of the TERMA PTV DVH is similar to the actual PTV shape, while the collapsed cone DVH has a more rounded shape.

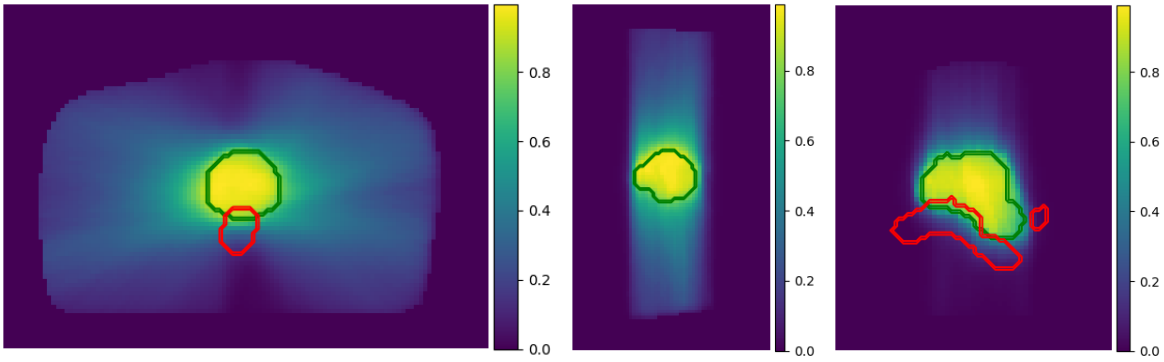
Both models are far from perfect, but do contain physical information of the segments, beam directions and intensities. They can therefore be used as input for the dose prediction model to try to incorporate physical information about the dose distribution in the prediction. Errors made by the dose engine can possibly be compensated by the deep learning model.

Apart from the accuracy, the calculation time is an important parameter in the use of a predictive dose al-



(a) Axial view (b) Coronal view (c) Sagittal view

Figure 4.10: Dose distribution of a single beam with collapsed cone convolution. The green contour is the PTV, the red contour is the rectum.



(a) Axial view (b) Coronal view (c) Sagittal view

Figure 4.11: Dose distribution combining values of all beams, corresponding segments and relative weights of the beams. The green contour is the PTV, the red contour is the rectum.

gorithm. The collapsed cone convolution method takes about five times longer than the TERMA calculation, increasing the time from 6 to 30 minutes. In this specific application, this time increase proved to be hard to work with as the generation of the collapsed cone dose distributions take more time than the neural network training. Considering that the collapsed cone method still achieves results that are not perfect, the TERMA distribution is used as input for the segment prediction model.

4.3. Segment prediction

A segment prediction needs to be done to provide input for the dose engine to create a dose distribution. As discussed in Section 3.6, the goal is not to predict the shapes perfectly, but to produce segments that give the possibility to calculate an approximate dose distribution using the dose engine.

4.3.1. Shape prediction

The three different models, also described in Section 3.6, have been trained to predict segment shapes of all 140 control points in the treatment plan of the patient. All three models produced very comparable results, both in individual shape prediction as well as in average DICE scores over all test patients. Examples of shape predictions can be found in Figure 4.13. From these examples can be seen that the predictions can be quite accurate and can resemble the actual shape very well, especially for the larger segments such as the first row. For more complex and smaller segments, the accuracy of the predictions is often more off, such as can be seen in the last two rows.

The average DICE scores for the test patients can be found in Figure 4.14. As all different models have approximately the same average DICE score for all different control points, and the inspected segments often

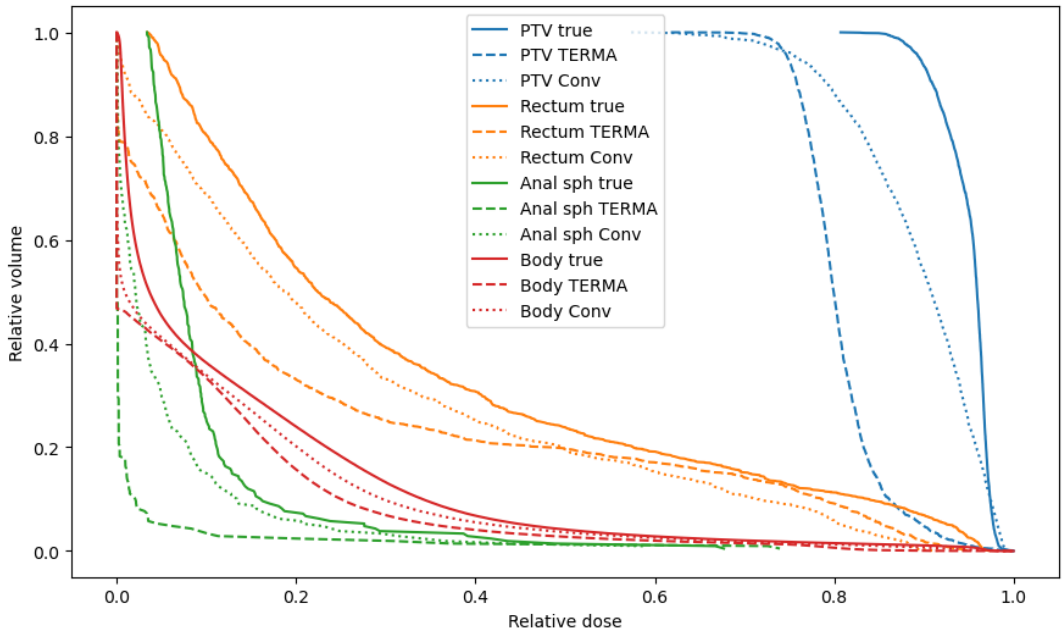


Figure 4.12: DVH plot of the dose engine models and the actual DVH. The solid line is the original dose, the dashed line is the DVH calculated from the TERMA distribution and the dotted line is the DVH of the collapsed cone approximation. It should be noted that the relative volume and relative dose is on the axis.

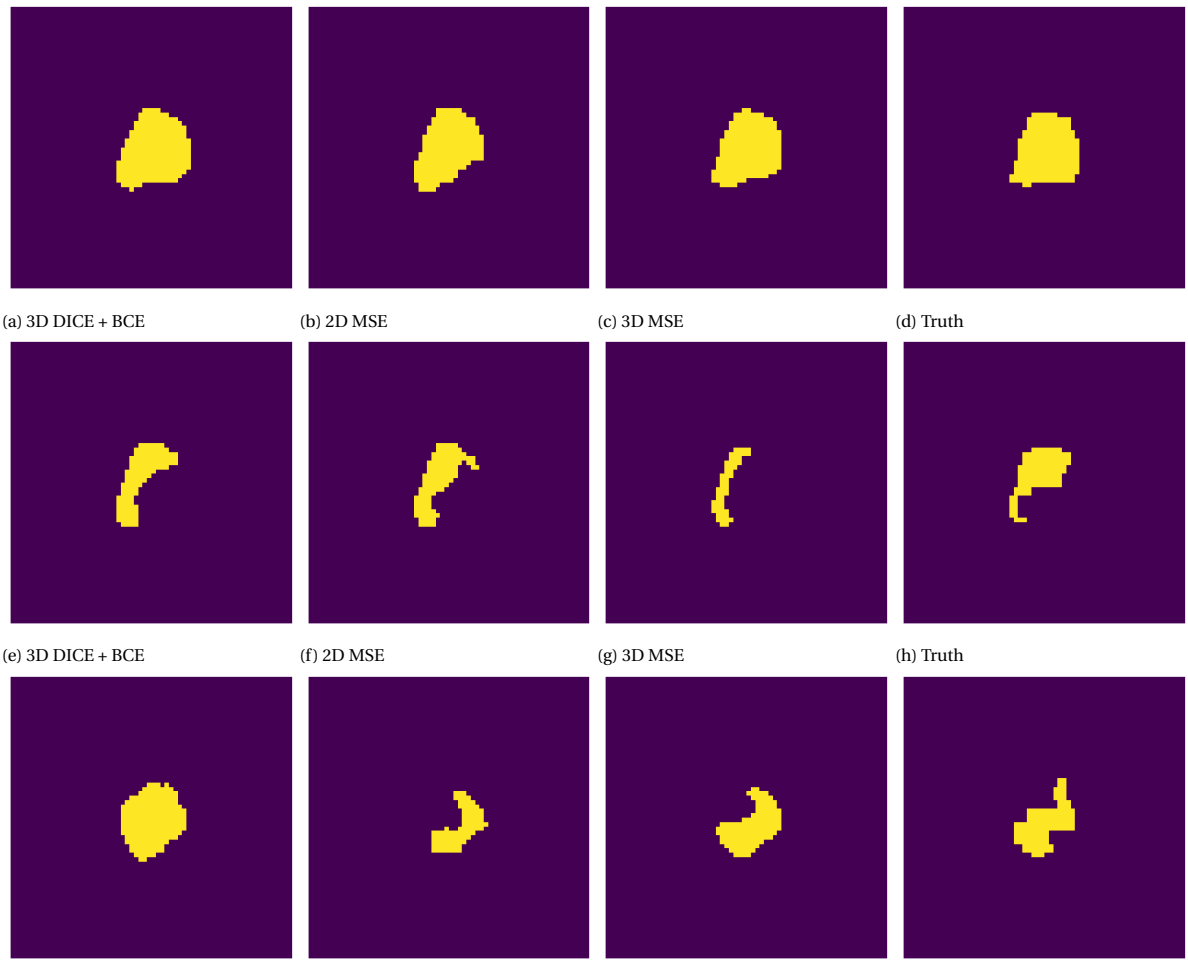
do not differ significantly visually, the prediction is not drastically different by using either of the three models. The 3D convolution model is finally used within the segment prediction, to keep all models of the same kind of architecture.

4.3.2. Weight prediction

The weight prediction was simultaneously done with the segment prediction. The weights of the segments are quite volatile, as can be seen in the example of the weights in Figure 4.15a. In Figure 4.15b an example of the predicted weights of the same patient is displayed. Although both seem to randomly go up and down, at least part of the prediction matches with the correct weight prediction: For example the weights around the 60th control point form a plateau, same as with the valley before the 120th control. As such, including weights is at least an improvement over uniformly weighted segments.

Using the shape prediction, the weight prediction and a MLC reconstruction algorithm, the dose distribution for a specific patient is calculated. An example of the prediction of the same patient as in Figure 4.2 can be found in Figure 4.16. It can be seen that the resulting dose distribution is far from perfect with high dose gradient and hot spots within the tumor. However, the ray effects are still visible as well as a lower dose at the top and bottom of the patient, outside the tumor. Therefore the prediction might still contain useful information for the U-Net in the dose prediction step.

From the DVH, displayed in Figure 4.17, the same can be seen. The dose within the PTV is clearly far from uniform, which results in parts of the PTV being hotspots. The same is true for all other structures. In general, the prediction seems a lot worse when compared to the prediction with the correct segments.



(a) 3D DICE + BCE (b) 2D MSE (c) 3D MSE (d) Truth

(e) 3D DICE + BCE (f) 2D MSE (g) 3D MSE (h) Truth

(i) 3D DICE + BCE (j) 2D MSE (k) 3D MSE (l) Truth

Figure 4.13: Examples of three different segment predictions with different contour complexity and size.

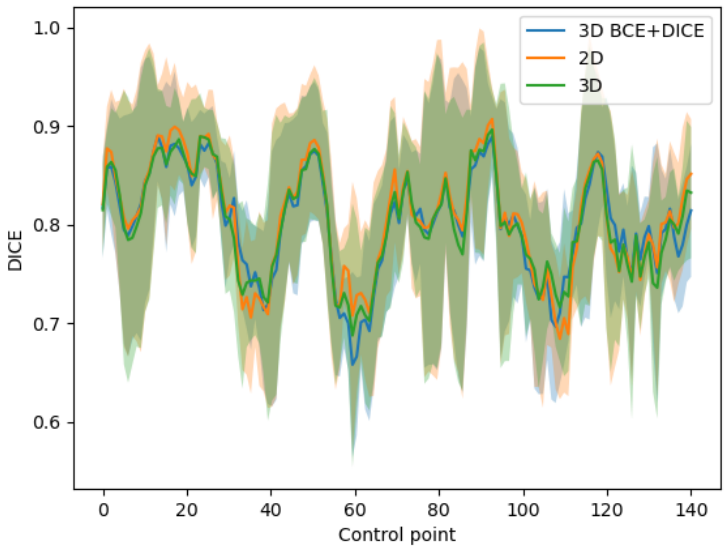
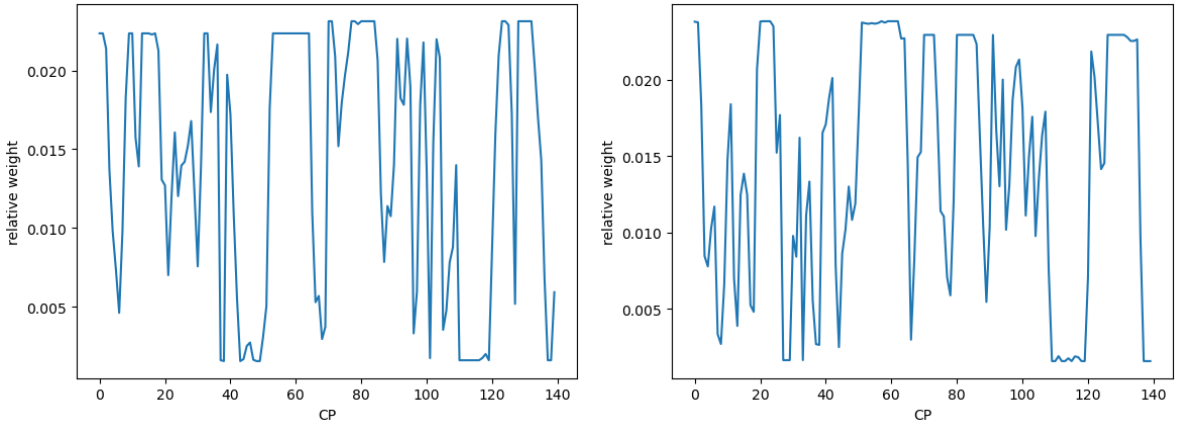


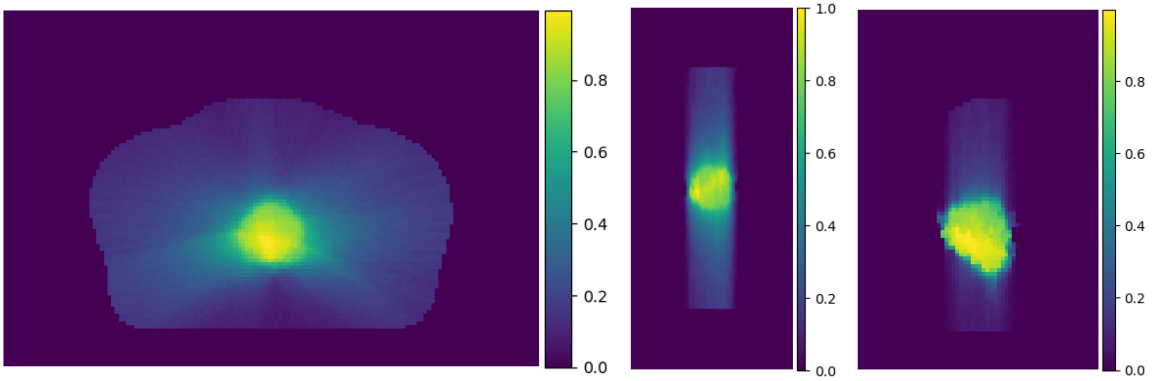
Figure 4.14: DICE scores per control point over the test set for the three models.



(a) Correct weights of segments of example patient

(b) Predicted weights of example patient

Figure 4.15: Example of the correct and predicted relative weights.



(a) Axial view

(b) Coronal view

(c) Sagittal view

Figure 4.16: Dose distribution by using the dose engine in combination with segment prediction.

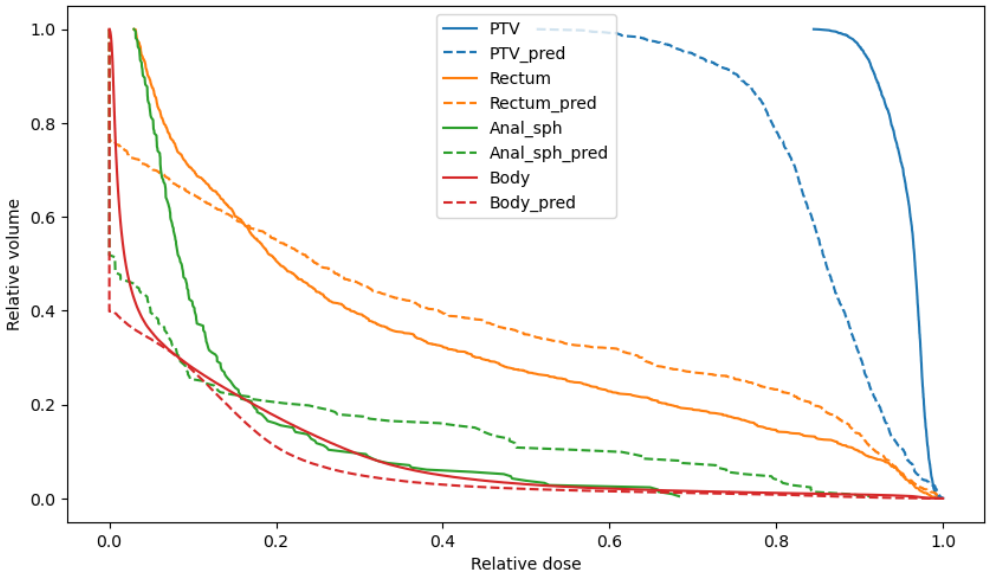


Figure 4.17: Relative DVH plot of a patient, by using the segment prediction and the dose engine. Note that the relative dose and relative volume are used.

4.4. Multi stage dose prediction

After calculating all segment dose predictions to be used as input, training the final neural network for dose prediction costs a similar amount of time as training the structure based dose prediction network. The training characteristics of the models can be found in Table 4.5. An example of the decrease of the loss functions during training are included in Figure 4.18. The other loss graphs are again displayed in Appendix A

Table 4.5: Training characteristics of the models, with the amount of epochs both training phases take and the final validation loss.

Model	**Epochs, first phase**	**Epochs, second phase**	**Validation loss**
Combined segment	6	39	0.864
Segment	5	49	1.853
Correct segments	13	114	0.433

From this table can be seen that the validation loss of the model with only correct segments as input has the lowest validation loss. Both other models have validation losses which are higher than the MSE model validation loss from the structure based dose prediction model, given in Table 4.1. The correct segment prediction also outperforms the earlier models in terms of final validation loss. Apart from that, it can be seen that using the correct segments takes a lot more epochs to train than when using the predicted segments. Examples of the predictions for a random test patient can be found in Figure 4.20.

Several things directly stand out from the dose distribution examples. The combined segment model prediction results in a visually uniform prediction, as was also the case in the simple dose prediction algorithm. The segment model gives a prediction with does not resemble the correct dose distribution at all and the maximum dose is predicted over 10 Gy higher compared to the true dose distribution. The correct segment model predicts a dose distribution which not uniform. It seems to approximate the ray effects seen from the original dose distribution. This effect is even more clear when looking at the difference plots given in Figure 4.19. In Figure 4.19a, the difference varies mostly within different angles to the isocenter following mistakes in intensities of rays from a certain angle. In Figure 4.19b, the difference is somewhat more random and the differences are not in ray like shapes anymore. Something that is worse in the correct segment predictions are the multiple artifacts visible in the prediction in the coronal and sagittal slices.

4.4.1. Coverage statistics and DVH comparisons

The performance of the models can again be further investigated by using dose characteristics. The average absolute percentage dose difference of the PTV coverage statistics are given in Table 4.6. For the rectum, together with the CI and HI the statistics are given in Table 4.7.

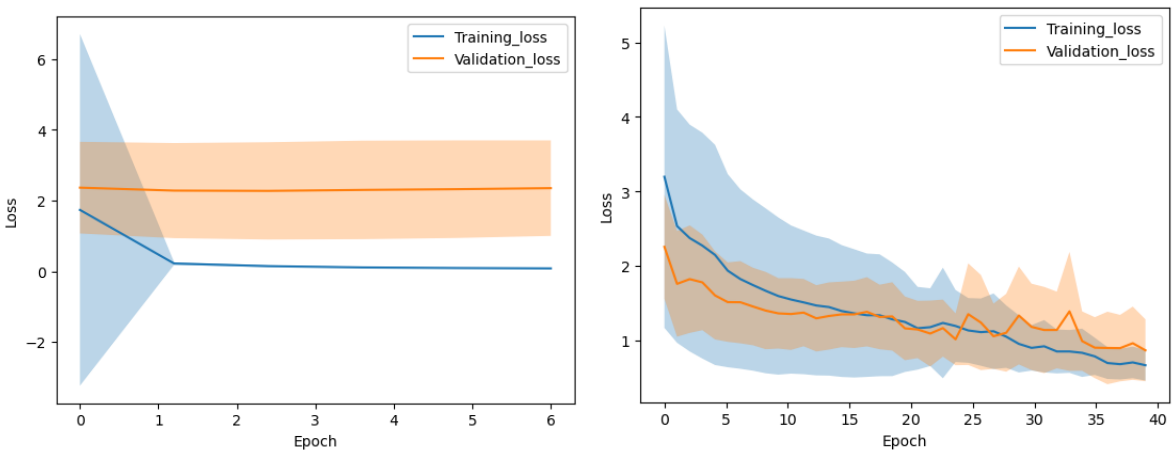


Figure 4.18: Example of the loss during training in two different phases of the combined model. The training loss and the validation loss are included. The shaded area gives the standard deviation of the loss.

Table 4.6: Average absolute % dose difference of several PTV coverage statistics

	Average absolute dose difference: **100** $\frac{D_{true}-D_{pred}}{D_{pres}}$			
	PTV coverage statistics			
	D95	**D98**	D_{max}	D_{mean}
Model	Mean ± SD	Mean ± SD	Mean ± SD	Mean ± SD
Combined segment	4.31 ± 3.26	4.37 ± 3.84	9.05 ± 5.00	4.69 ± 3.28
Segments	20.25 ± 14.46	26.28 ± 16.33	15.64 ± 6.52	5.75 ± 4.72
Correct segment	5.71 ± 6.57	6.92 ± 8.19	4.63 ± 4.09	2.01 ± 1.76

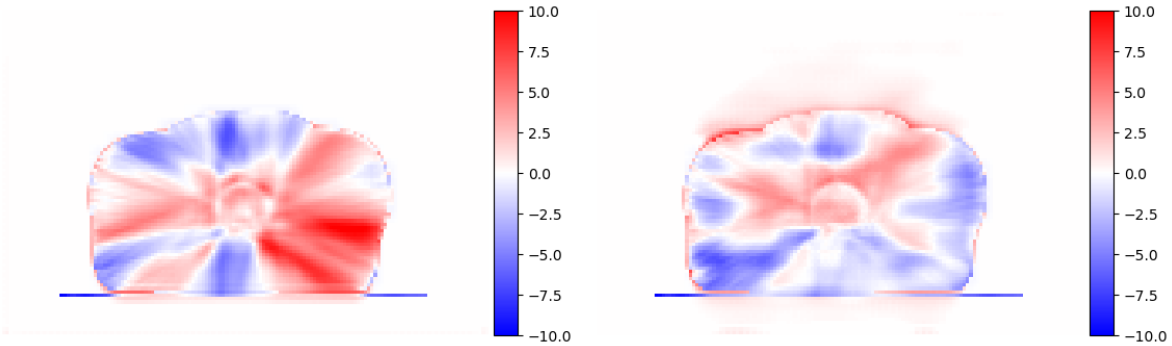
Table 4.7: Average absolute % dose difference of several rectum coverage statistics, Conformation index, Homogeneity.

	Average absolute dose difference: **100** $\frac{D_{true}-D_{pred}}{D_{true}}$				
	Rectum coverage statistics, CI, HI				
	D_{max}	D_{mean}	**V45**	**CI**	**HI**
Model	Mean ± SD	Mean ± SD	Mean ± SD	Mean ± SD	Mean ± SD
Combined segment	5.52 ± 3.94	4.50 ± 3.56	5.80 ± 4.81	24.29 ± 12.69	10.84 ± 4.72
Segment	10.01 ± 6.35	21.31 ± 14.13	39.61 ± 25.21	14.00 ± 11.34	40.72 ± 18.45
Correct segment	4.00 ± 4.68	8.43 ± 6.39	14.90 ± 15.00	16.29 ± 8.11	10.58 ± 11.88

From the values it directly stands out that all models are underperforming in coverage statistics accuracy compared to the structure based prediction algorithm. Especially the segment model has very high inaccuracies. Apart from that, the combined model seems to have better performance of the dose coverage statistics than the correct segment model. Only the maximum prediction value is better within the correct segment model.

Again the DVH of the prediction can be investigated to see more complete information of the prediction accuracy. For clarity, the prediction of the segment model has been excluded in the DVH plot as it can already be concluded that the prediction is not accurate. The DVH can be seen in Figure 4.21. From this figure it can be seen that the prediction models for this single patient do perform reasonable in the OARs with comparable accuracy as in Figure 4.4. The prediction within the PTV is more off and shows a large over prediction compared to the actual DVH. Also, the difference between the two models does not seem to be very large from this DVH

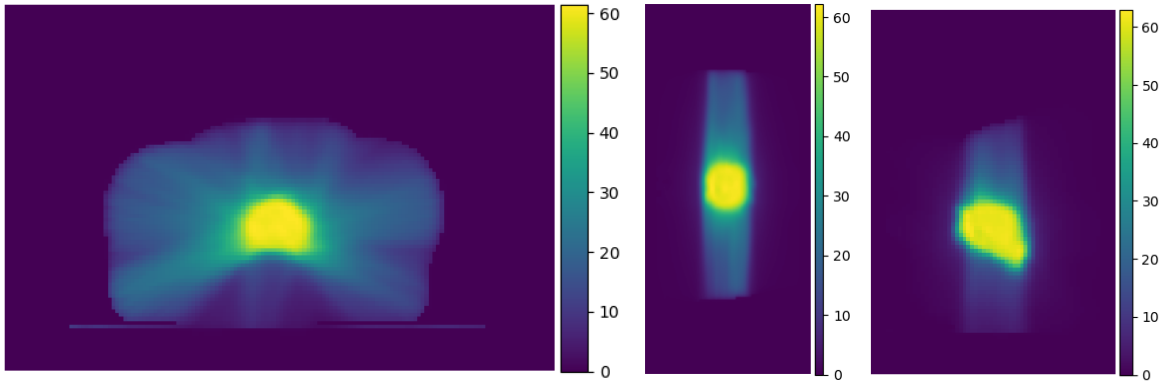
A more quantifiable approach can be done by looking at the individual average dose difference per patient. The combined prediction model performs a lot better in the DVH prediction than the correct model as all average predictions of the DVH are better. Also the correct segment model has a much more variable prediction and there are individual patients that are predicted very bad, such as test patient 5 and 10.



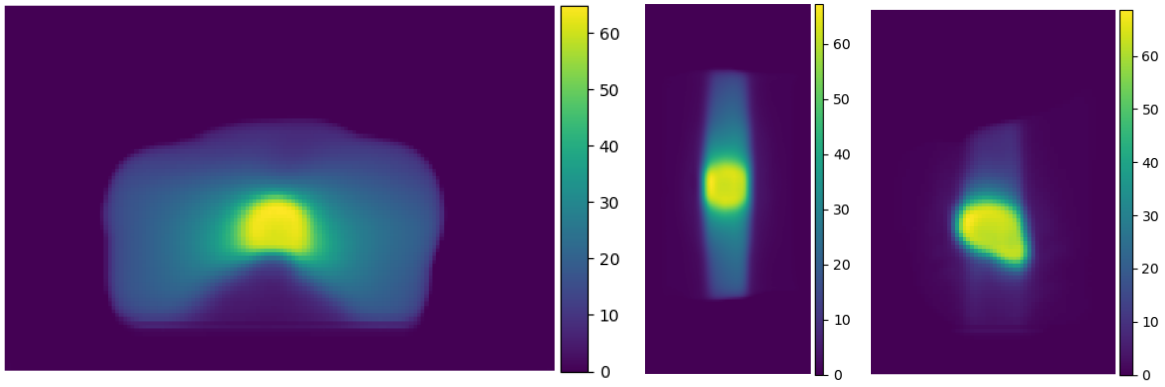
(a) Combined model

(b) Correct segment model

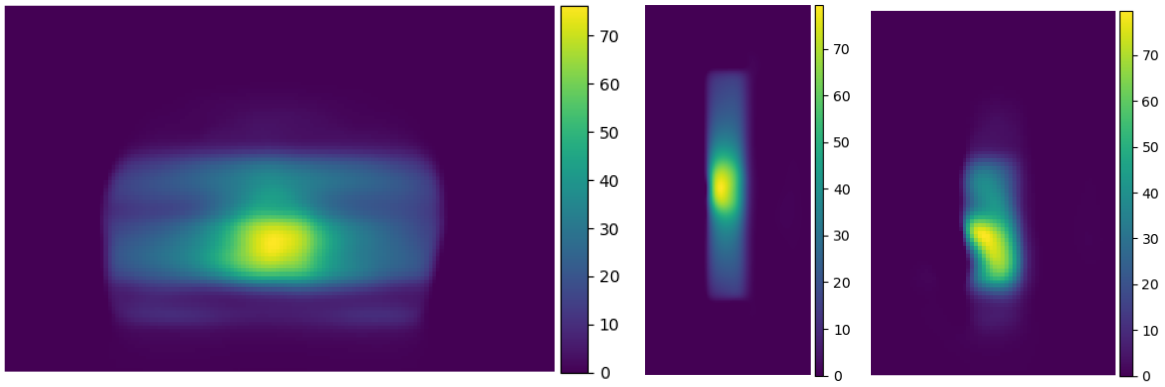
Figure 4.19: Difference plots of the combined and correct segment prediction model in axial view



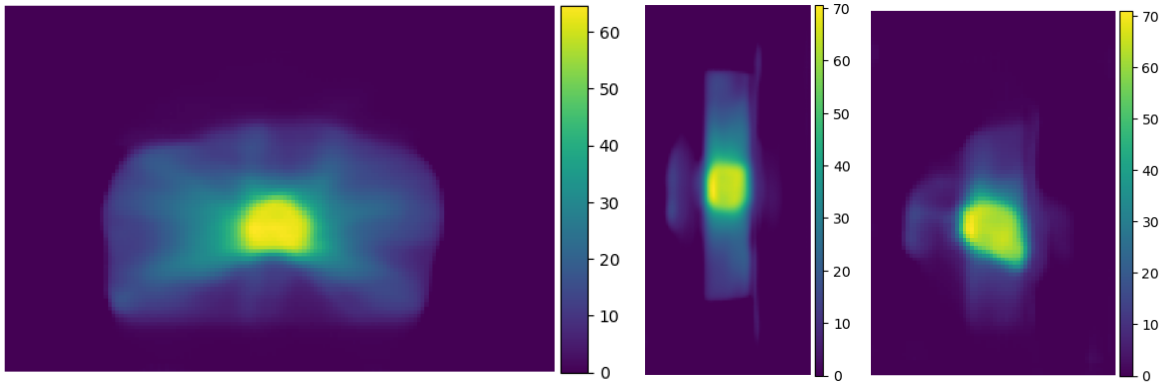
(a) Original dose, axial view (b) Coronal view (c) Sagittal view



(d) Combined model, axial view. (e) Coronal view (f) Sagittal view



(g) Segment model, axial view (h) Coronal view (i) Sagittal view



(j) Correct segment model, axial view. (k) Coronal view (l) Sagittal view

Figure 4.20: Summary of dose prediction model using segment dose approximations. The axial, rotated coronal and sagittal view are respectively displayed. First row contains the true dose distribution, the second row contains the prediction of the combined model, the third row contains the segment model and the last row contains the correct segment model. Doses are give in Gy.

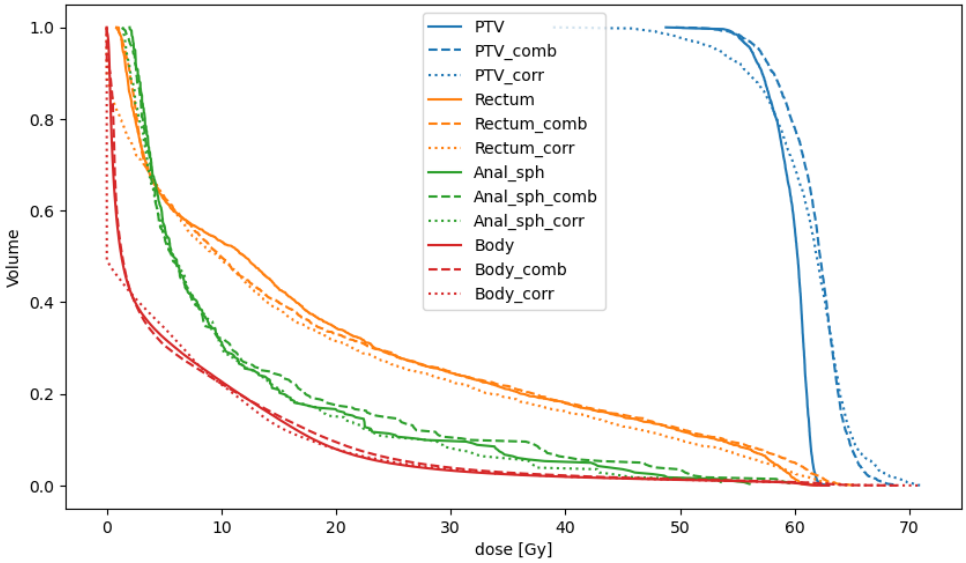


Figure 4.21: DVH of the combined prediction model and the correct prediction model.

Table 4.8: Individual average absolute DVH dose difference per patient for the PTV, the rectum, the anal sphincter and the entire body. The values are calculated for the combined segment model (comb) and the correct segment model (corr). The values represent the average difference per DVH point p. The standard deviation indicates the variation in DVH point difference in the patient. An average calculation of these values is included at the bottom of the table.

		Average Absolute dose difference: $\frac{1}{N}\sum_{p=0}^{N}\lvert D_{true,p} - D_{pred,p}\rvert$			
		Quantified DVH difference (Gy)			
		PTV	**Rectum**	**Anal Sph.**	**Body**
Patient	Model	Mean ± SD	Mean ± SD	Mean ± SD	Mean ± SD
1	Comb	4.67 ± 1.76	1.72 ± 1.19	0.49 ± 0.47	0.56 ± 0.76
	Corr	5.09 ± 5.41	13.22 ± 10.30	5.32 ± 5.07	0.76 ± 0.55
2	Comb	1.78 ± 1.05	0.69 ± 0.45	0.41 ± 0.32	0.86 ± 0.91
	Corr	8.78 ± 8.93	7.98 ± 5.84	1.20 ± 1.42	2.70 ± 2.12
3	Comb	1.48 ± 1.03	1.12 ± 0.86	1.46 ± 1.28	0.23 ± 0.35
	Corr	17.79 ± 13.32	25.89 ± 15.99	30.36 ± 12.92	1.21 ± 1.08
4	Comb	3.36 ± 1.22	1.75 ± 1.62	0.61 ± 1.41	0.54 ± 0.54
	Corr	5.47 ± 6.10	7.93 ± 7.01	9.50 ± 10.87	0.83 ± 0.60
5	Comb	1.67 ± 0.83	0.95 ± 0.74	0.94 ± 1.77	0.59 ± 0.62
	Corr	26.49 ± 11.66	15.95 ± 11.69	10.01 ± 4.31	1.24 ± 0.90
6	Comb	7.29 ± 1.59	2.32 ± 2.74	1.22 ± 1.49	0.71 ± 1.23
	Corr	6.03 ± 6.71	19.76 ± 12.85	31.71 ± 15.93	1.17 ± 1.05
7	Comb	3.04 ± 1.49	1.99 ± 2.00	1.03 ± 0.57	0.25 ± 0.46
	Corr	5.09 ± 6.29	5.53 ± 6.15	0.89 ± 0.68	0.78 ± 0.52
8	Comb	1.47 ± 1.03	0.95 ± 0.96	0.99 ± 1.36	0.33 ± 0.50
	Corr	2.23 ± 1.65	1.73 ± 0.96	0.62 ± 0.95	0.59 ± 0.38
9	Comb	1.35 ± 1.43	1.42 ± 1.08	0.63 ± 0.51	0.16 ± 0.22
	Corr	17.12 ± 14.33	13.35 ± 9.06	12.70 ± 5.02	1.38 ± 0.94
10	Comb	5.04 ± 2.38	1.78 ± 1.31	1.04 ± 0.79	0.90 ± 1.18
	Corr	26.26 ± 15.65	11.49 ± 6.17	4.59 ± 0.69	3.25 ± 3.18
11	Comb	1.18 ± 1.05	1.30 ± 1.23	0.80 ± 0.33	0.15 ± 0.18
	Corr	4.73 ± 5.07	5.15 ± 5.68	2.13 ± 3.09	1.07 ± 0.73
12	Comb	1.16 ± 0.83	0.84 ± 0.76	0.92 ± 0.49	0.18 ± 0.25
	Corr	5.07 ± 6.32	1.59 ± 1.68	2.84 ± 4.18	0.57 ± 0.39
Average	Comb	2.79 ± 1.31	1.40 ± 1.25	0.88 ± 0.90	0.46 ± 0.60
	Corr	10.85 ± 8.45	10.80 ± 7.78	9.32 ± 5.43	1.30 ± 1.04

5
Discussion

In this study, dose prediction using neural networks has been researched. In the following part the results of the structure based dose prediction, the dose engine, the segment prediction and the multi stage prediction are discussed and interpreted individually.

5.1. Structure based dose prediction

Training the different structure based dose prediction models generally worked out as expected. In the first phase, the validation error quickly stopped decreasing in most models, while the training loss shrank fast. As the uncertainty of the training loss quickly decreased, the network learned to handle the augmentations quickly. However, the generalizability of the single patient with augmentation to the validation set ran into its limits. This lack of generalizability could mean that the augmented dataset was too simple to have a significant effect on the training. A relatively small first training phase was also seen in the DoseNet article of Kearney et. al. [33]. In the second phase more epochs were needed. The training loss generally decreased steadily, but the decrease slowed down as the loss became lower, which is again as expected. Approximately the same behavior is seen in the validation loss.

Compared to several instances in literature it stands out that there is the low amount of epochs trained before a network stops early. The relatively quick convergence could mean that the used dataset is more homogeneous than the patient sets used in literature, especially since it is a dataset that contains autoplanned patients. For example in the article of Nguyen et. al. it is specifically noted that there are patients in the dataset with vastly different structure geometries [10]. Also the Kandalan et. al. paper contains different patient groups [43]. Apart from that, a variation in the results can originate from the early stopping mechanism as small differences in validation loss dictate the amount of epochs trained. For more accurate results it would therefore be beneficial to implement k-fold cross validation to be able to make an estimate of the variation within model training. When implementing this, it would also be beneficial to increase the computational power used for training.

From Figure 4.2 can be seen that the prediction of the neural network, in this case with MSE loss function, predicts a largely uniform dose distribution outside of the PTV and rectum area. Using structures as input does therefore not provide enough information to neural network to learn the physical effects of the different beams. This outcome is as expected as no effort was taken to possibly include physics information within this network. When using the uniform prediction for evaluation or quality control purposes, this does not necessarily pose a problem as long as evaluation properties such as PTV or rectum DVH's are predicted accurately.

5.1.1. Prediction accuracy

The WMSE model outperforms the models with other loss function on dose characteristics. This on itself is an interesting outcome as most state of the art dose prediction models use the MSE as loss function [23] [33] [43]. However, the increased performance can easily be explained. As bad predictions in the PTV and OAR regions are punished more severe by the loss function, the network prioritizes these regions. Therefore the prediction in these structures becomes more accurate. On the other hand prioritizing the PTV and OARs

might cause the parts outside of these structures to be predicted worse. This is indicated by the higher difference and uncertainty within the DVH point predictions in the body, as can be found in Table 4.4.

Another observation is that the models with high weights in the loss function perform worse on the same dose characteristics prediction. A reason for this could be that the network has a harder time converging to a solution with the high weights. An indication for this is that the uncertainty in the training loss is higher in the high weighted models, as can be seen in Appendix A. A solution could be to use a lower learning rate within these models.

The CI and HI statistics seem to follow the same trend as the other coverage statistics by being generally lower in the WMSE model. It should be noted however, that especially the CI gives very high error values that are often significantly larger than in literature, which would indicate that at least the conformity of the dose is not as in the true dose.

Although model performance is highly dependent on the training data, the dose characteristics of the two best preforming models can be compared to several prostate dose prediction models from literature to give an indication of the accuracy. One comparable model is the model of Norouzi Kandalan et. al [43]. Where the source model represents the most common planning style in a similar sized dataset. The PTV dose characteristics are mostly within 3%, while the rectum dose characteristics are within 2% for the D_{max} and 1% for the D_{mean}. Both the MSE model and the weighted MSE model from the structure based dose prediction have errors much lower than the 3% for the PTV and have comparable accuracy for the rectum D_{max}. Only the error in the rectum D_{mean} is much higher compared to the source model of Norouzi Kandalan et. al.

The dose characteristics can also be compared to the dose prediction model of Nguyen et. al [10]. The PTV prediction metrics are within 3%, while the max and mean values are all within 2%, a performance slightly better than the Norouzi Kandalan paper. The MSE model and the weighted MSE model both match these dose characteristics. An important difference however is that the prediction model is based on IMRT prediction with only seven beams instead of VMAT. A prediction with VMAT is more complex, which indicates a good accuracy of the models presented in this study. Concluding, the results obtained from the structure based prediction model is in good accordance with the models from literature.

5.1.2. Clinical accuracy

From the average absolute dose difference in DVH from Table 4.8 can be seen that the average absolute error of both predictions is about 1.5% in the PTV, which is in line with the values of the PTV dose characteristics. The accuracy of the two models varies within the two evaluated organs at risk, but the differences fall well in the variation of the predictions. This variation can also be seen by looking at individual cases, where both models outperform each other in DVH prediction in different patients. Based on these quantified DVH metrics, there is not one model that clearly outperformes the other.

One interesting difference between the two models is the average standard deviation. The WMSE model has a lower variation in average absolute dose difference for the PTV and OARs, but a higher variation looking at the entire body of the patient, which can also be seen from the presented box plot in Figure 4.3 This can again be explained by the focus of the loss function on the PTV and OARs. In conclusion, the loss function used for a model should depend on the goals of a prediction: When only the prediction in the PTV and OARs are important the WMSE should be chosen. When the entire prediction is needed, the MSE is a better fit.

The only other study that reported DICE scores come from the paper of Nguyen et. al. [10]. The found DICE scores of 0.91 on average is identical to the reported DICE scores of Nguyen et. al. Apart from that, the shape of the similarity curve is also comparable, with both having a dipping accuracy at 40% of the volume. The only major difference is the drop in the highest isodose volumes, which is less prominent in the Nguyen paper.

5.1.3. Performance compared to OVH prediction model

In a more practical application, the neural network prediction does produce a more accurate rectum DVH compared to the OVH prediction model. As the prediction speed is negligible once the network has been trained, implementing the neural network prediction could therefore provide a direct beneficial clinical im-

pact to evaluations in treatment planning. However, for the prediction to be part of an evaluation pipeline, the rectum prediction still needs to be integrated in the evaluation and more tests need to be done to ensure that the model generalizes for more prostate patients..

Overall, the structure based dose prediction matches the accuracy of models from literature very well and performs well in predicting dose characteristics. The usage of the WMSE can improve the prediction accuracy in the dose characteristics. The presented network works well and can be used as the basis for the physics guided multi stage neural network approach.

5.2. Dose engine

When using the correct segments, the dose engine is able to reconstruct a dose distribution that is visually very similar to the true dose distribution. The dose distribution is centered around the isocenter which is located in the PTV and the high dose area follows the tumor shape closely. The same ray effects with corresponding areas of higher and lower dose arise from the dose engine reconstruction. This indicates that reconstruction part of the dose engine works well.

However, the dose calculation shows that the acquired results are still far from perfect. From the DVH example in Figure 4.4 can be seen that the DVH of the actual dose distribution is not approximated very well, neither by the TERMA nor by the collapsed cone convolution model. Because of the five times longer computation time of the collapsed cone convolution dose engine, it was not feasible to use the convolution model for the creation of neural network input. Therefore, the TERMA distribution is used in practice, which is an imperfect approximation of the dose distribution without scatter. The main purpose is for the engine to produce a fast estimation of the dose distribution that provides useful physical information from the individual segments for the neural network, for which task the TERMA distribution is still useful.

There are more aspects of the dose engine that could have lead to errors. First there are inaccuracies present in the reconstruction part of the engine. In practice the radiation source has a virtual focus point from which the radiation originates in a simulation. This would create a cone like ray. In the reconstruction, the approximation is made that the source delivers parallel rays through the MLC to the target, which will therefore not be cone shaped and not completely realistic.

Other inaccuracies might be explained by an approximation that arises in the TERMA calculation. The distance traveled in the body is approximated using a standard entrance point and the unit vector of the beam direction, as given in Equation 3.10. Because of this approach, an inaccuracy will arise whenever the entrance plane is not perpendicular to the beam direction. This effect can for example be seen in in the right part of the beam in Figure 4.7a where TERMA value remains at a maximum well away from the entrance point. Although the absolute TERMA values in this way are calculated more accurately, than with an approach of only one entrance point, this approximation can induce a bias into either side of the beam, which can lead to error is the dose distribution. On the other hand, partially overlapping beams can counteract the effect.

Overall, taking these approximations into account, the dose engine creates a visually similar dose distribution that seems to contain extra information as opposed to the structure set.

5.3. Segment prediction

The expectation was that the network would not be able to predict perfect segments. This turned out to be correct. With the used approach, there was a high variability in the accuracy of the MLC opening contours. While there are larger segments that can be approximated quite well, smaller and more complex segments were predicted less accurately. This variability is clearly seen within the DICE coefficient plot for all the model in Figure 4.14.

One thing that stands out is that the performance of the different trained models are very similar. While the predicted segments do locally differ, overall performance, measured again with the DICE score, seems to provide similar results. This might indicate that the DICE score is not the perfect indicator for the accuracy of these predicted contours. Other metrics such as the Hausdorff distance could therefore be investigated to be used instead. Alternatively, this can be caused by also using the DICE statistic to determine the threshold

constant for contour determination. Including the binary classification within the network could be an improvement.

Using the predicted MLC positions in the dose engine, leads to a dose distribution that is far from perfect. The prediction does still seems to contain some new information as some ray effects are still present. However, an increase in the accuracy of the segment prediction would be needed to provide useful information for the dose prediction network. Increasing the segment prediction accuracy would therefore be an interesting possible focus point for further research. One of the ideas would be to use an architecture more specialized for contour predictions.

5.4. Multi stage dose prediction

For the combined prediction model and the correct prediction model the training process went as expected and similar to the training of the structure based dose prediction model. Just as in the training of the structure based dose prediction model, the training loss decreased quickly in the first training phase, while there was a more gradual decrease during the second training phase. Also the standard deviation in the loss decreased steadily. This was not the case for the segment model, where the variation in the training did not decrease at all. This indicates that the network had a hard time generalizing a correct output for the given input.

It could also be seen that it took much longer to train the model with the correct segments as opposed to the predicted segments. An explanation for this is that the predicted segments do not provide enough information for the neural network to learn something useful. On the other hand in the correct segment model, the network needs to learn to interpret the intricate dose distribution information in the correct segments, which can take more time to generalize.

5.4.1. Dose distribution examples

When looking at examples of the dose distribution such as in Figure 4.20 a couple of things stand out. First of all, the combined segment model produces a dose distribution that is visually similar to the dose distributions from the structure prediction model. The PTV has a high dose, while the rectum area receives much less dose and the rest of the body is mostly uniform. This is not what was expected beforehand. By adding the segment dose as extra input, the expectation was that the neural network would be able to learn some of the provided physics information. Therefore, the expectation was to see some of the ray like effects in the predicted dose distribution as well. Due to the lack of these effects, it looks like this segment dose information was mainly predicted using the structure information.

From the dose prediction example of the segment model, it can directly be seen that the dose prediction is far off. The high dose area does not conform the PTV shape and the total dose does not follow the body shape either. Also, the maximum dose is very high and there are horizontal low dose artifacts present within in the body. Therefore it seems that the segment dose prediction does not provide enough information to predict a useful dose distribution.

To test this hypothesis, the results from the correct segment are evaluated as well. In the correct segment model it stands out that while the general shape of the dose to the PTV is again correct, the dose in the rest of the body is not uniform anymore. Instead the model predicts ray like effects similar to the actual dose distribution. With the correct segments it therefore seems that the neural network does learn physical properties of the dose distribution with the segment dose as input. Consequently, the predicted segments are the bottleneck in the segment dose prediction pipeline, as that is the only difference between the two models.

5.4.2. Quantatative analysis

The quantative analysis further supports the claim that the segment dose prediction does not provide enough information for a useful dose distribution. Comparing the values of the validation losses from Table 4.5 with the validation loss of the MSE loss structure prediction model, it can be seen that the validation loss of both the combined and the segment prediction model loss are higher. However, the validation loss of the correct segment prediction is lower. This indicates that overall, the correct segment prediction model produces a better prediction than the structure prediction model, as both are trained on the exact same dataset and are evaluated with an equal loss function.

As the loss is an average value over the entire patient, decreasing the loss does not necessarily result in an increased clinical prediction accuracy. The accuracy is again investigated using dose statistics. From the statistics, given in Table 4.6 and Table 4.7, can be seen that both the combined segment model and the correct segment model perform worse than the original structure based dose prediction over all characteristics. The improved validation loss did therefore not increase the accuracy of the dose statistics.

The average absolute dose difference from Table 4.8 shows that the prediction accuracy of the correct segment model is very variable over all structures. Although a positive effect can visually be seen, the variability indicates that the model is far from useful. Testing the correct segment prediction with the structures as input as well would therefore be a good approach to see if the ray effects effect remain while the individual variability is decreased.

This is further supported when looking at the DVH and DVH differences, from which can be concluded that including the structures has a large impact on the accuracy of the model. Even with the predicted segments, the combined model outperforms the model with the correct segments. Structures should therefore always be included when constructing a final prediction model.

A problem with the segment dose prediction is the usage of the dose distributions normalized on the max dose per patient. Although the shape of the dose distribution is consistent through the normalization, the inter patient absolute dose information is not preserved. Especially distributions with a long PTV DVH tail in the high dose part will be normalized with higher maximum values than more homogeneous distributions. This could be one of the sources of the high variability within the correct segment model prediction. By doing the normalization in a different way that accounts for inter patient absolute differences, the results could be possibly less variable. The extent into which the this normalization problem has influenced is not known.

5.4.3. Outlook

The results show that the current segment dose prediction network does not predict MLC positions and weights accurate enough to increase the accuracy of the dose prediction model. However, when using the correct segments, the neural network does learn to include the physics partially. An interesting question that arises from this conclusion therefore is: When are the segment predictions accurate enough to be used in this multi stage learning method? If the MLC and weights need to approximate the actual values before they become useful, then this multi stage learning approach provides no benefit anymore and the focus should be shifted to possibly predicting the machine parameters directly. If that is not the case, the multi stage learning approach could still be useful. An important direction for following research would therefore be to learn more about the segment predictions.

Another possibility would be to try the other PGNN approach in which the physical properties are included within the loss function directly. By using the physics information within the loss function, the neural network directly optimizes for the physics information, making sure that the physics information is actually used. Accurate segment predictions would still be needed for this to work.

Finally a last improvement would be to include the entire pipeline within a single neural network. The neural network would predict the MLC positions and weights, calculate the corresponding distribution using a transformation from the dose engine, and predict the final dose distribution in one network. The possible advantage of this approach is that the individual parts of the network are optimized together, which makes sure that the different parts of the network are all optimized for dose prediction task.

6

Conclusion

In this study the applications of deep learning for dose prediction of autoplanned patients have been investigated. The aim was to produce a neural network that is able to predict the dose in a patient based on structure sets of the patient as input, and to produce a dose prediction pipeline that includes physics information, in an attempt to make the model more accurate and more realistic.

The neural network that has been developed for the structure based dose prediction was based on the U-Net architecture. It proved to be able to accurately predict the dose distribution in the test patients. The prediction of the PTV DVH has an error of about 1.5% averaged over all DVH points, which is in absolute value smaller than 1 Gy on average. Also the different OAR predictions had an average prediction difference of about 1 Gy. This is in accordance with performance of state-of-the-art models in literature. However, it should be noted that the result is dependent on the dataset used, which in this case was a homogeneous set of autoplans.

It could also be concluded that using a weighted MSE loss function can improve the prediction performance of the model on DVH statistics. The WMSE loss function outperforms the often used MSE loss for PTV statistics up to 0.7% and the rectum D_{mean} by more than 2%. Only the D_{max} is predicted better by the normal MSE. Therefore, applications in which DVH statistics are used, such as knowledge based planning or evaluations based on DVH predictions, benefit from using the WMSE loss. Furthermore, the structure based dose prediction showed to improve the currently used rectum DVH prediction model.

A physics guided neural network pipeline has been developed which incorporates physics information from individual segments into a dose prediction using a hybrid-physics data model. The pipeline consists of three main elements. The segment prediction model, the dose engine and the dose prediction model. From the results can be concluded that this implementation of the prediction pipeline was not able to increase the accuracy of the prediction model, nor make the dose distribution more realistic.

It has been shown that using the correct MLC contours and weights, the final neural network can predict a dose distribution that is heavily influenced by the physics information. The correct model achieves a lower loss than the structure prediction model and produces both more realistic dose distribution. It can therefore be concluded that the segment prediction is the bottleneck in the prediction pipeline. The segment prediction network was not able to reproduce MLC contours and relative weights accurately enough to provide useful physics information for the dose prediction.

Further research should be focused on two areas. First, as the prediction of the structure based model has a good accuracy, the clinical applications of this structure based model should be further investigated. An option for this is outlier detection for autoplans. Secondly, further research should focus on finding out how much the segment prediction should be improved before the network can learn useful things from it and consequently what the best way is to further improve the accuracy of the segment prediction model.

A

Loss of different prediction models

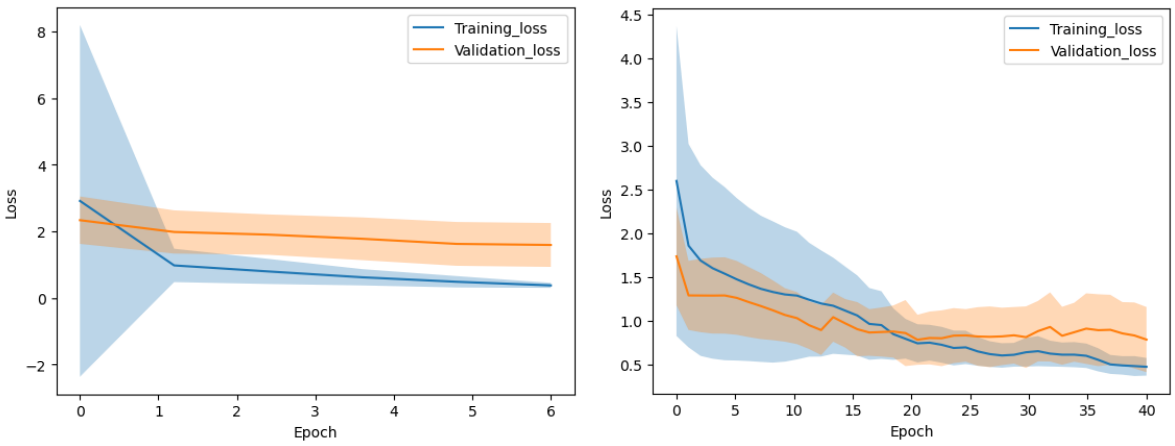


Figure A.1: Example of the loss during training in two different phases of the MSE model. The training loss and the validation loss are included. The shaded area gives the standard deviation of the loss.

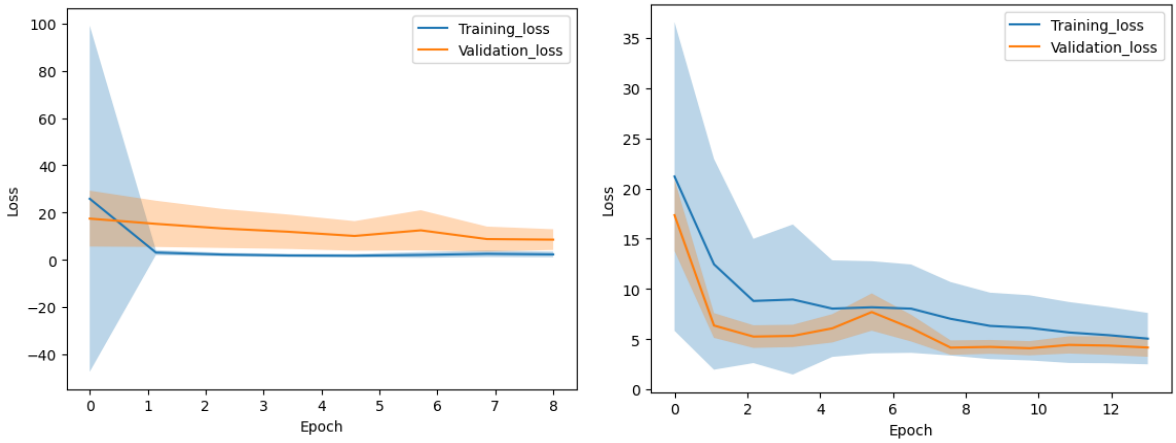


Figure A.2: Example of the loss during training in two different phases of the HWMSE model. The training loss and the validation loss are included. The shaded area gives the standard deviation of the loss.

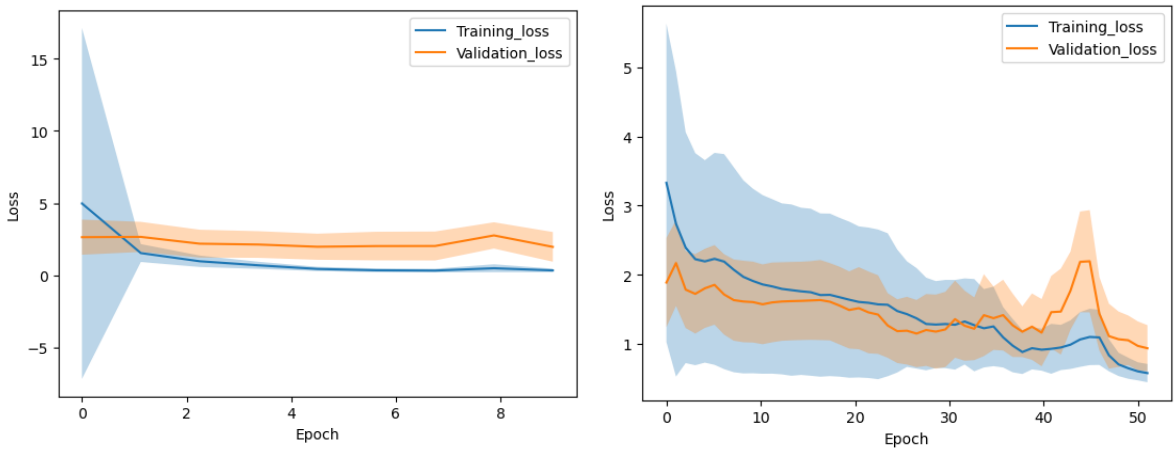


Figure A.3: Example of the loss during training in two different phases of the Heaviside model. The training loss and the validation loss are included. The shaded area gives the standard deviation of the loss.

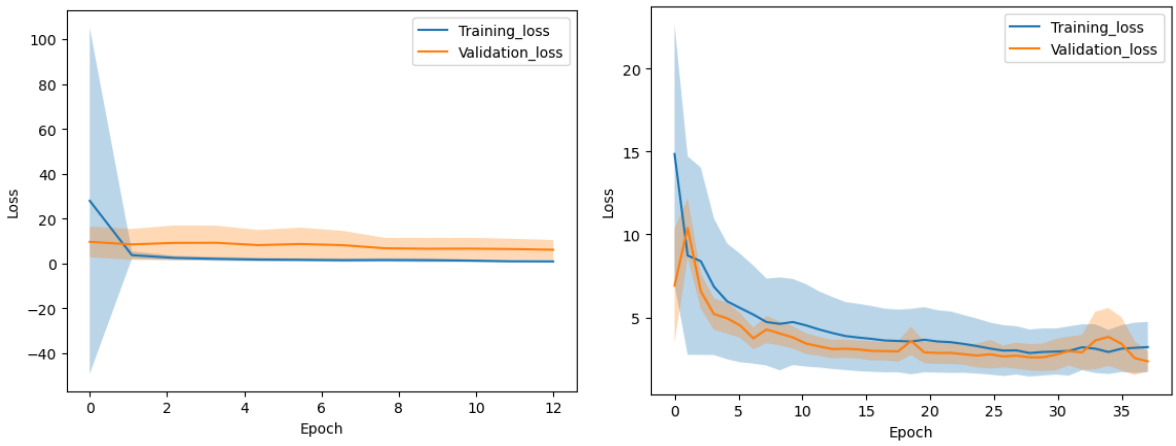


Figure A.4: Example of the loss during training in two different phases of the high weight Heaviside model. The training loss and the validation loss are included. The shaded area gives the standard deviation of the loss.

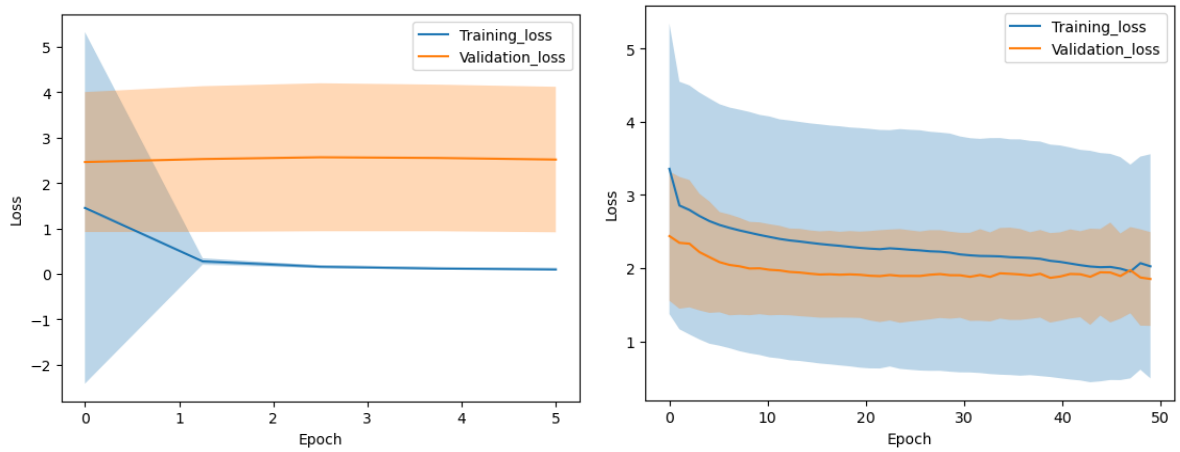


Figure A.5: Example of the loss during training in two different phases of the segment prediction model. The training loss and the validation loss are included. The shaded area gives the standard deviation of the loss.

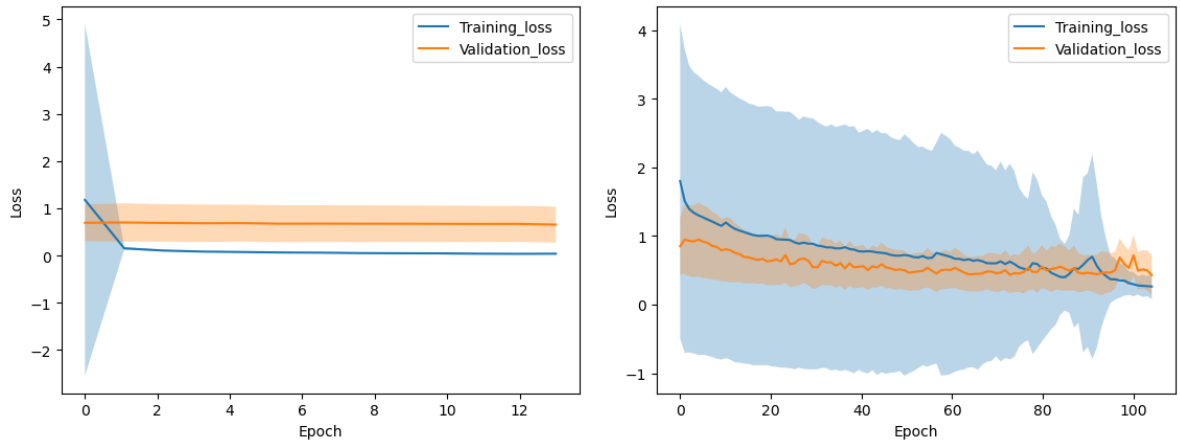


Figure A.6: Example of the loss during training in two different phases of the correct segment prediction model. The training loss and the validation loss are included. The shaded area gives the standard deviation of the loss.

B

Dose prediction examples

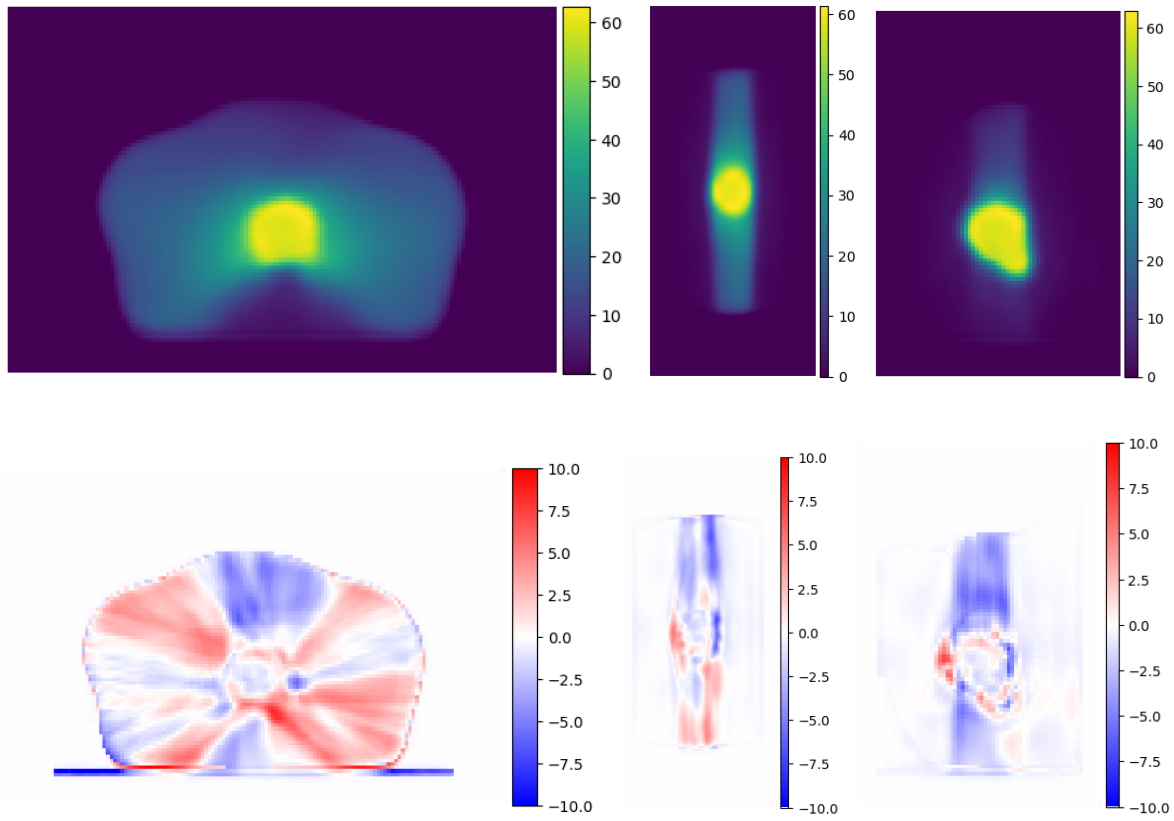


Figure B.1: Dose prediction example of the prediction dose of the WMSE and the difference with the true value. Doses and differences are plotted in Gy.

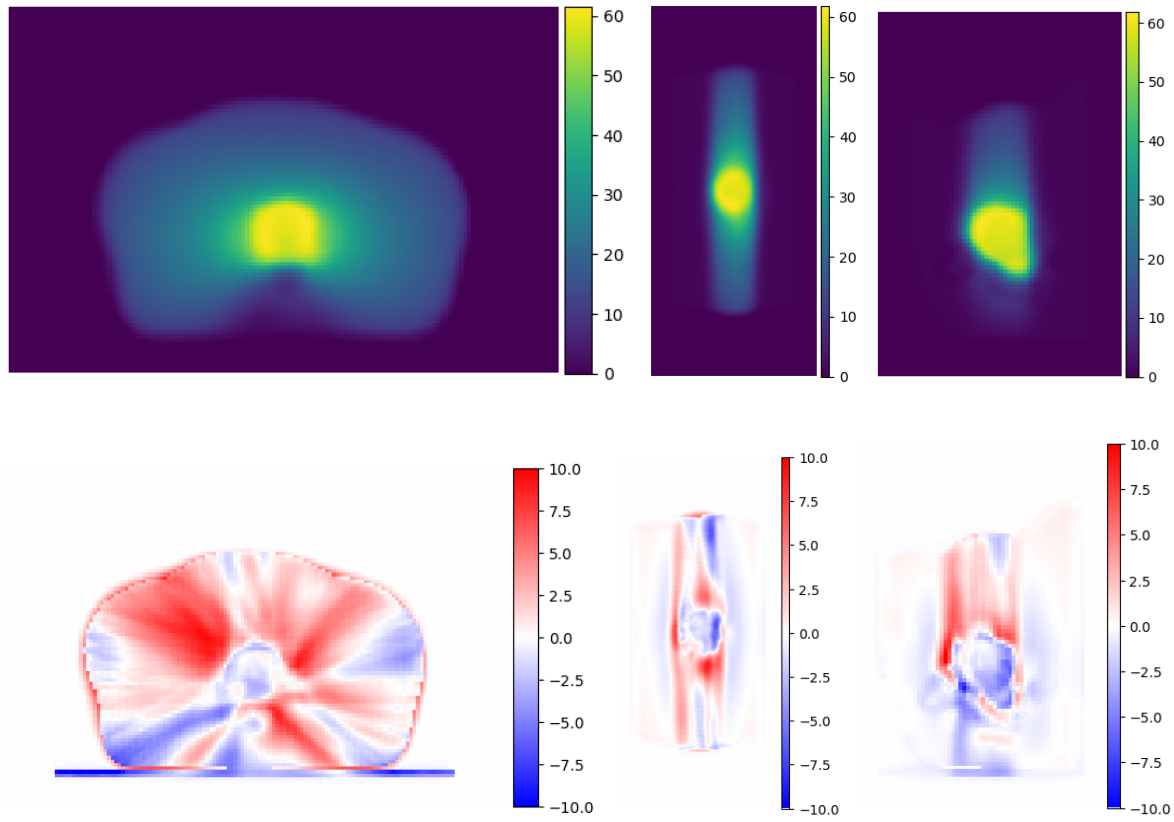


Figure B.2: Dose prediction example of the prediction dose of the HWMSE and the difference with the true value. Doses and differences are plotted in Gy.

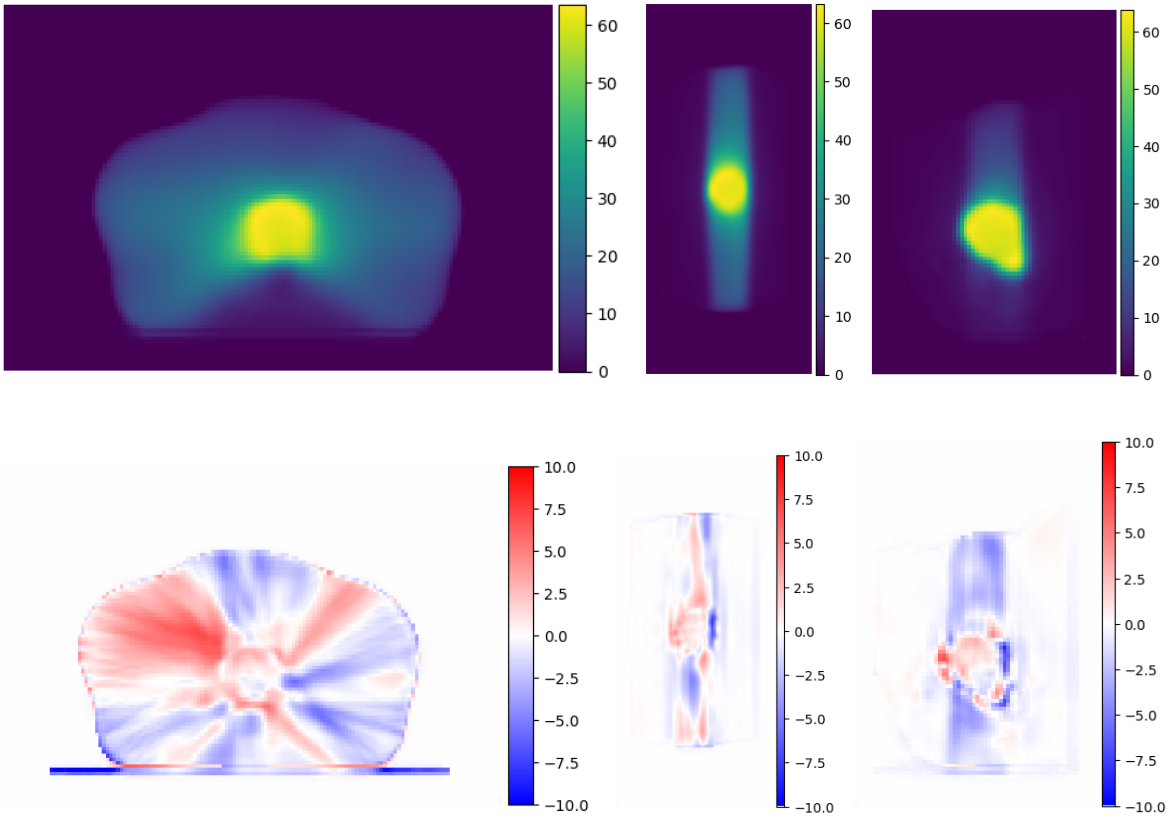


Figure B.3: Dose prediction example of the prediction dose of the Heaviside MSE and the difference with the true value. Doses and differences are plotted in Gy.

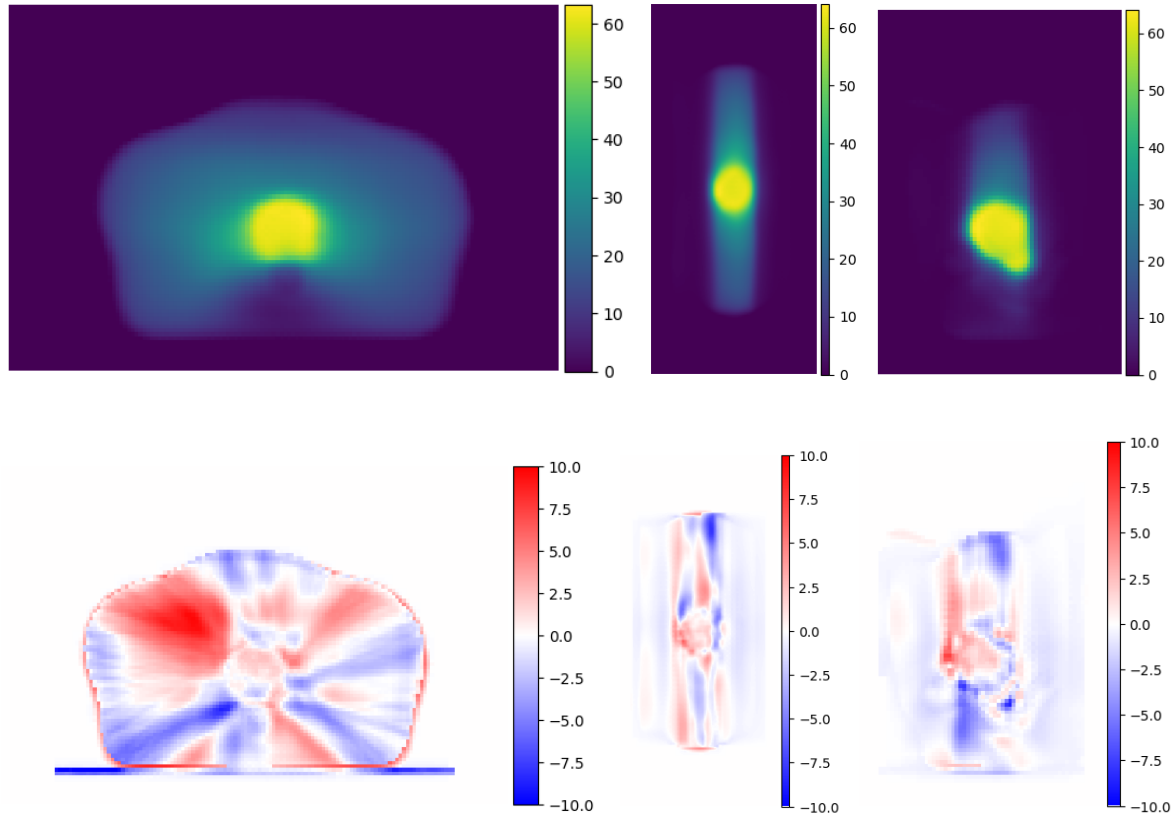


Figure B.4: Dose prediction example of the prediction dose of the high weight Heaviside MSE and the difference with the true value. Doses and differences are plotted in Gy.

Bibliography

[1] CBS Statline. *Overledenen; belangrijke doodsoorzaken*. 2020. URL: https://opendata.cbs.nl/#/CBS/nl/dataset/7052_95/table?ts=1612817420182.

[2] Obioma Nwankwo et al. "Knowledge-based radiation therapy (KBRT) treatment planning versus planning by experts: validation of a KBRT algorithm for prostate cancer treatment planning". In: *Radiation Oncology* 10.1 (Dec. 2015), p. 111. ISSN: 1748-717X. DOI: 10.1186/s13014-015-0416-6. URL: https://ro-journal.biomedcentral.com/articles/10.1186/s13014-015-0416-6 (visited on 12/07/2020).

[3] Jerome Krayenbuehl et al. "Evaluation of an automated knowledge based treatment planning system for head and neck". In: *Radiation Oncology* 10.1 (Dec. 2015), p. 226. ISSN: 1748-717X. DOI: 10.1186/s13014-015-0533-2. URL: http://www.ro-journal.com/content/10/1/226 (visited on 12/07/2020).

[4] Kanabu Nawa et al. "Evaluation of a commercial automatic treatment planning system for prostate cancers". In: *Medical Dosimetry* 42.3 (2017), pp. 203–209. ISSN: 09583947. DOI: 10.1016/j.meddos.2017.03.004. URL: https://linkinghub.elsevier.com/retrieve/pii/S0958394717300365 (visited on 12/07/2020).

[5] Tomas M Janssen. "Independent knowledge-based treatment planning QA to audit Pinnacle autoplanning". In: *Radiotherapy and Oncology* (2019), p. 7.

[6] Benjamin P Ziemer et al. "Fully automated, comprehensive knowledge-based planning for stereotactic radiosurgery: Preclinical validation through blinded physician review". In: *Practical Radiation Oncology* (2017), p. 10.

[7] Jim P. Tol et al. "Can knowledge-based DVH predictions be used for automated, individualized quality assurance of radiotherapy treatment plans?" In: *Radiation Oncology* 10.1 (Dec. 2015), p. 234. ISSN: 1748-717X. DOI: 10.1186/s13014-015-0542-1. URL: http://www.ro-journal.com/content/10/1/234 (visited on 02/02/2021).

[8] Victor Strijbis. *Machine learning for knowledge-based dose-volume histogram prediction in prostate cancer*. Master thesis, Delft University of Technology. 2018.

[9] Satomi Shiraishi and Kevin L. Moore. "Knowledge-based prediction of three-dimensional dose distributions for external beam radiotherapy: Knowledge-based prediction of 3D dose distributions". In: *Medical Physics* 43.1 (Dec. 29, 2015), pp. 378–387. ISSN: 00942405. DOI: 10.1118/1.4938583. URL: http://doi.wiley.com/10.1118/1.4938583 (visited on 12/07/2020).

[10] Dan Nguyen et al. "A feasibility study for predicting optimal radiation therapy dose distributions of prostate cancer patients from patient anatomy using deep learning". In: *Scientific Reports* 9.1 (Dec. 2019), p. 1076. ISSN: 2045-2322. DOI: 10.1038/s41598-018-37741-x. URL: http://www.nature.com/articles/s41598-018-37741-x (visited on 12/07/2020).

[11] Arif N. Ali et al. "Dosimetric comparison of volumetric modulated arc therapy and intensity-modulated radiation therapy for pancreatic malignancies". In: *Medical Dosimetry* 37.3 (Sept. 2012), pp. 271–275. ISSN: 09583947. DOI: 10.1016/j.meddos.2011.10.001. URL: https://linkinghub.elsevier.com/retrieve/pii/S0958394711002032 (visited on 01/03/2021).

[12] Flemming Kjaer-Kristoffersen et al. "RapidArc volumetric modulated therapy planning for prostate cancer patients". In: *Acta Oncologica* 48.2 (2009), pp. 227–232.

[13] Uwe Schneider, Eros Pedroni, and Antony Lomax. "The calibration of CT Hounsfield units for radiotherapy treatment planning". In: *Physics in Medicine and Biology* 41.1 (Jan. 1, 1996), pp. 111–124. ISSN: 0031-9155, 1361-6560. DOI: 10.1088/0031-9155/41/1/009. URL: https://iopscience.iop.org/article/10.1088/0031-9155/41/1/009 (visited on 01/03/2021).

[14] Tomas Pavel. "Feasibility of magnetic resonance imaging-based radiation therapy for brain tumour treatment". PhD thesis. Aug. 2017. DOI: 10.13140/RG.2.2.21791.87209.

[15] Werner Bär et al. “A comparison of forward and inverse treatment planning for intensity-modulated radiotherapy of head and neck cancer”. In: *Radiotherapy and Oncology* 69.3 (Dec. 2003), pp. 251–258. ISSN: 01678140. DOI: 10.1016/j.radonc.2003.08.002. URL: https://linkinghub.elsevier.com/retrieve/pii/S0167814003003062 (visited on 01/03/2021).

[16] Sebastiaan Breedveld et al. “Multi-criteria optimization and decision-making in radiotherapy”. In: *European Journal of Operational Research* 277.1 (Aug. 2019), pp. 1–19. ISSN: 03772217. DOI: 10.1016/j.ejor.2018.08.019. URL: https://linkinghub.elsevier.com/retrieve/pii/S0377221718307148 (visited on 12/07/2020).

[17] Herman Suit et al. “Proton Beams to Replace Photon Beams in Radical Dose Treatments”. In: *Acta oncologica (Stockholm, Sweden)* 42 (Dec. 2003), pp. 800–8. DOI: 10.1080/02841860310017676.

[18] Raphael Jumeau et al. “Stereotactic radiotherapy for the management of refractory ventricular tachycardia: Promise and Future Directions”. In: *Frontiers in cardiovascular medicine* 7 (2020).

[19] Canadian Cancer Society. *LINAC*. 2020. URL: https://www.cancer.ca/~/media/CCE/80/487f67de9ecc29aae236e675030b191f.png.

[20] Irene Hazell et al. “Automatic planning of head and neck treatment plans”. In: *Journal of Applied Clinical Medical Physics* 17.1 (Jan. 2016), pp. 272–282. ISSN: 15269914. DOI: 10.1120/jacmp.v17i1.5901. URL: http://doi.wiley.com/10.1120/jacmp.v17i1.5901 (visited on 12/07/2020).

[21] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. http://www.deeplearningbook.org. MIT Press, 2016.

[22] Kevin Jarrett et al. “What is the best multi-stage architecture for object recognition?” In: *2009 IEEE 12th international conference on computer vision*. IEEE. 2009, pp. 2146–2153.

[23] Dan Nguyen et al. “3D radiotherapy dose prediction on head and neck cancer patients with a hierarchically densely connected U-net deep learning architecture”. In: *Phys. Med. Biol.* (2019), p. 16.

[24] Léon Bottou. “Large-scale machine learning with stochastic gradient descent”. In: *Proceedings of COMPSTAT’2010*. Springer, 2010, pp. 177–186.

[25] George Cybenko. “Approximation by superpositions of a sigmoidal function”. In: *Mathematics of control, signals and systems* 2.4 (1989), pp. 303–314.

[26] Amelie Byun et. al. *Coursen notes, CS231n: Convolutional Neural Networks for Visual Recognition*. 2020. URL: http://cs231n.stanford.edu/.

[27] Jost Tobias Springenberg et al. “Striving for Simplicity: The All Convolutional Net”. In: *arXiv:1412.6806 [cs]* (Apr. 13, 2015). arXiv: 1412.6806. URL: http://arxiv.org/abs/1412.6806 (visited on 12/07/2020).

[28] Yuxin Wu and Kaiming He. “Group normalization”. In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, pp. 3–19.

[29] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. “U-Net: Convolutional Networks for Biomedical Image Segmentation”. In: *arXiv:1505.04597 [cs]* (May 18, 2015). arXiv: 1505.04597. URL: http://arxiv.org/abs/1505.04597 (visited on 12/07/2020).

[30] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. “Learning Deconvolution Network for Semantic Segmentation”. In: *2015 IEEE International Conference on Computer Vision (ICCV)*. 2015 IEEE International Conference on Computer Vision (ICCV). Santiago, Chile: IEEE, Dec. 2015, pp. 1520–1528. ISBN: 978-1-4673-8391-2. DOI: 10.1109/ICCV.2015.178. URL: http://ieeexplore.ieee.org/document/7410535/ (visited on 12/07/2020).

[31] Jonathan Long, Evan Shelhamer, and Trevor Darrell. “Fully Convolutional Networks for Semantic Segmentation”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015.

[32] Özgün Çiçek et al. “3D U-Net: learning dense volumetric segmentation from sparse annotation”. In: *International conference on medical image computing and computer-assisted intervention*. Springer. 2016, pp. 424–432.

[33] Vasant Kearney. “DoseNet: a volumetric dose prediction algorithm using 3D fully-convolutional neural networks”. In: *Phys. Med. Biol.* (2018), p. 12.

[34] Gao Huang et al. “Densely Connected Convolutional Networks”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017.

[35] Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. “Physics Informed Deep Learning (Part I): Data-driven Solutions of Nonlinear Partial Differential Equations”. In: *arXiv:1711.10561 [cs, math, stat]* (Nov. 28, 2017). arXiv: 1711.10561. URL: http://arxiv.org/abs/1711.10561 (visited on 12/07/2020).

[36] Anuj Karpatne et al. “Physics-guided Neural Networks (PGNN): An Application in Lake Temperature Modeling”. In: *arXiv:1710.11431 [physics, stat]* (Feb. 20, 2018). arXiv: 1710.11431. URL: http://arxiv.org/abs/1710.11431 (visited on 12/07/2020).

[37] Woong Cho et al. “Practical implementation of a collapsed cone convolution algorithm for a radiation treatment planning system”. In: *Journal of the Korean Physical Society* 61.12 (Dec. 2012), pp. 2073–2083. ISSN: 0374-4884, 1976-8524. DOI: 10.3938/jkps.61.2073. URL: http://link.springer.com/10.3938/jkps.61.2073 (visited on 12/07/2020).

[38] Weiguo Lu et al. “Accurate convolution/superposition for multi-resolution dose calculation using cumulative tabulated kernels”. In: *Physics in Medicine and Biology* 50.4 (Feb. 21, 2005), pp. 655–680. ISSN: 0031-9155, 1361-6560. DOI: 10.1088/0031-9155/50/4/007. URL: https://iopscience.iop.org/article/10.1088/0031-9155/50/4/007 (visited on 12/07/2020).

[39] Adam Paszke et al. “Pytorch: An imperative style, high-performance deep learning library”. In: *arXiv preprint arXiv:1912.01703* (2019).

[40] Charles R. Harris et al. “Array programming with NumPy”. In: *Nature* 585.7825 (Sept. 2020), pp. 357–362. DOI: 10.1038/s41586-020-2649-2. URL: https://doi.org/10.1038/s41586-020-2649-2.

[41] Pauli Virtanen et al. “SciPy 1.0: fundamental algorithms for scientific computing in Python”. In: *Nature methods* 17.3 (2020), pp. 261–272.

[42] G. Bradski. “The OpenCV Library”. In: *Dr. Dobb's Journal of Software Tools* (2000).

[43] Roya Norouzi Kandalan et al. “Dose Prediction with Deep Learning for Prostate Cancer Radiation Therapy: Model Adaptation to Different Treatment Planning Practices”. In: *arXiv:2006.16481 [physics]* (June 29, 2020). arXiv: 2006.16481. URL: http://arxiv.org/abs/2006.16481 (visited on 12/07/2020).

[44] Arie Van'T Riet and Leo H Elders. “A CONFORMATION NUMBER TO QUANTIFY THE DEGREE OF CONFORMALITY IN BRACHYTHERAPY AND EXTERNAL BEAM IRRADIATION: APPLICATION TO THE PROSTATE”. In: (1997), p. 6.

[45] Tejinder Kataria et al. “Homogeneity Index: An objective tool for assessment of conformal radiation treatments”. In: *Journal of medical physics/Association of Medical Physicists of India* 37.4 (2012), p. 207.

[46] Lee R. Dice. “Measures of the Amount of Ecologic Association Between Species”. In: *Ecology* 26.3 (July 1945), pp. 297–302. ISSN: 00129658. DOI: 10.2307/1932409. URL: http://doi.wiley.com/10.2307/1932409 (visited on 12/18/2020).

[47] Carole H Sudre et al. “Generalised dice overlap as a deep learning loss function for highly unbalanced segmentations”. In: *Deep learning in medical image analysis and multimodal learning for clinical decision support*. Springer, 2017, pp. 240–248.

[48] Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi. “V-Net: Fully Convolutional Neural Networks for Volumetric Medical Image Segmentation”. In: *arXiv:1606.04797 [cs]* (June 15, 2016). arXiv: 1606.04797. URL: http://arxiv.org/abs/1606.04797 (visited on 12/07/2020).

[49] Anders Ahnesjo. “Collapsed cone convolution of radiant energy for photon dose calculation in heterogeneous media”. In: (Aug. 15, 1988).

[50] Yuyin Zhou et al. “A fixed-point model for pancreas segmentation in abdominal CT scans”. In: *International conference on medical image computing and computer-assisted intervention*. Springer. 2017, pp. 693–701.