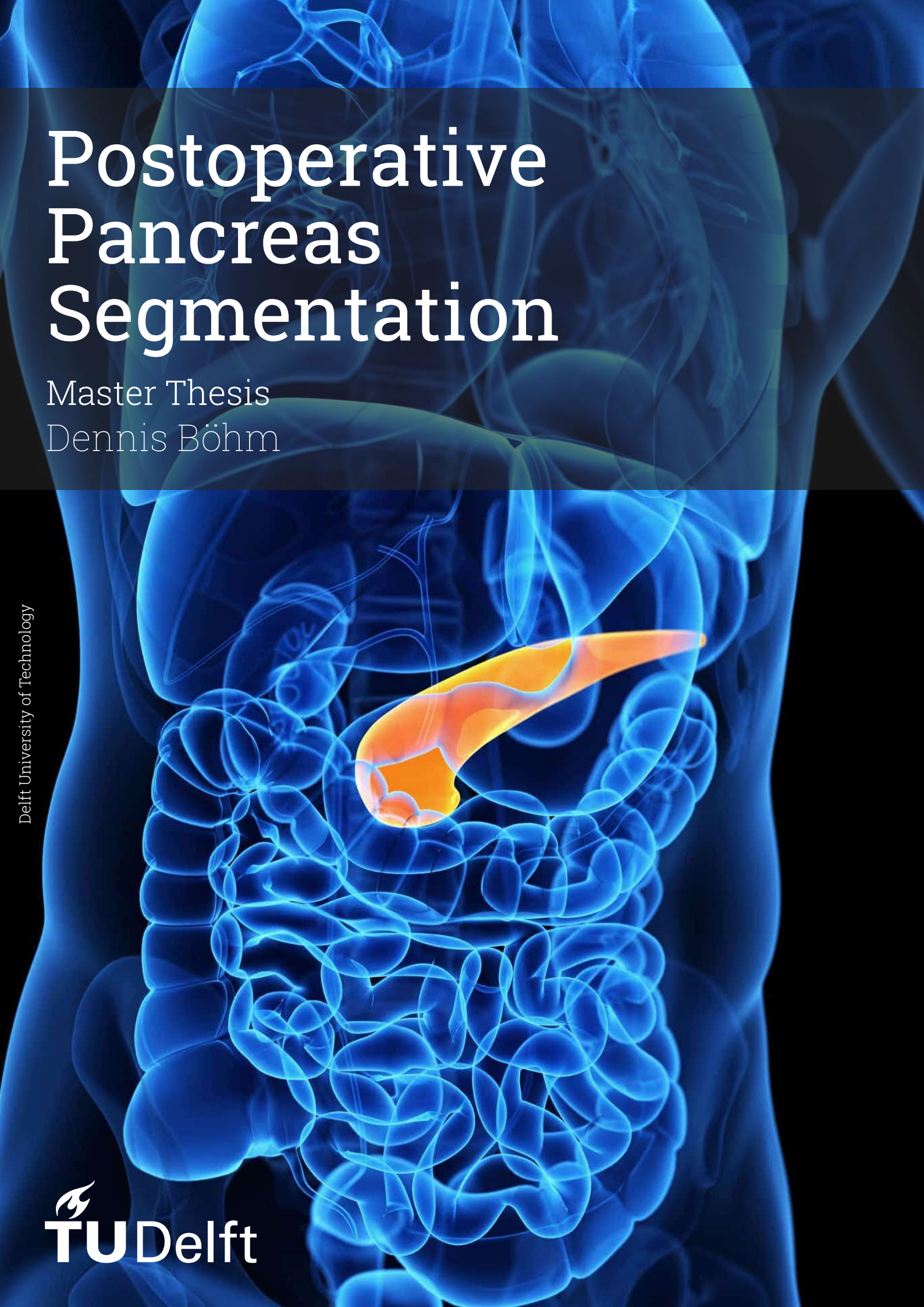
Postoperative Pancreas Segmentation

Master Thesis

Dennis Böhm



Delft University of Technology

TU Delft

Postoperative Pancreas Segmentation

Master Thesis

by

Dennis Böhm

Student Number

4689305

Thesis Advisor:	Prof.dr.ir. Boudewijn Lelieveldt
Daily Supervisor:	dr. Michael Weinmann
Daily Co-Supervisor:	dr.ir. Jouke Dijkstra
Company Supervisor:	Willem van der Geest

Cover: Kelly, Clint. "What Is an Artificial Pancreas - Who Makes It and the Cost - Prescription Hope." Accessed October 14, 2022. https://prescriptionhope.com/blog-what-is-an-artificial-pancreas/.

TU Delft

DATACATION...
DEDICATED TO DATA

With a 5-year survival rate of only 9%, pancreatic cancer is one of the deadliest types of cancer. Among other things, this is caused by the extreme difficulty of diagnosing recurrent pancreatic cancer in an early stage. One of the important next steps in discovering pancreatic cancer automatically on a CT scan is finding healthy pancreatic tissue. This research explores the feasibility of segmenting the pancreas on a CT scan using a deep learning approach, focusing on postoperative cases after pancreatic resection, by combining state-of-the-art segmentation models for the preoperative pancreas with novel techniques, such as 3D Multi-Scale Convolutional Blocks (MCBs) and KNet architectures. Using pretraining on preoperative data, a complete segmentation pipeline was designed to segment the pancreas in the postoperative state. The experimental results demonstrate that deep learning models can effectively segment the pancreas, despite the increased complexity and difficulty of the postoperative state. Notably, employing advanced techniques such as the KNet architecture and MCBs yields significant performance improvements. The newly designed architectures presented in this research, MKNet, MSKNet, and MAKNet, achieve state-of-the-art results for both preoperative and postoperative pancreas segmentation, greatly improving the Hausdorff Distance (HD) and 95th Percentile Hausdorff Distance (HD95) in particular. In the preoperative state, substantial performance improvements were observed compared to the previous state-of-the-art, with a 12.17% increase for HD and a 14.64% increase for HD95. Similarly, in the postoperative state, improvements of 11.99% for HD and 13.25% for HD95 were measured. Additionally, qualitative evaluations were conducted with seven experienced radiologists and radiotherapists to validate the algorithm's performance in a clinical context. Remarkably, in 83% of the cases, the algorithm was evaluated to accurately segment the pancreas, requiring minimal or no modifications according to the medical experts.

This study not only contributes to the state-of-the-art in pancreas segmentation but also introduces comprehensive quantitative and qualitative evaluation methodologies. Furthermore, it addresses the segmentation of the postoperative pancreas, which has not yet been touched upon in previous research. Limitations are acknowledged, such as the availability of a limited dataset, and future research directions are proposed, including generalizability and enhancements to the proposed models. Despite the limited availability of data, which impacts the performance and generalization capabilities to some degree, the research in this thesis showcases the capability of the MKNet family architectures to accurately segment the postoperative pancreas, offering potential benefits for medical applications, data annotation acceleration, and future research in this domain.

Contents

1 Introduction

With an estimated overall 5-year survival rate of only 9%, pancreatic cancer is one of the most deadly malignancies [115]. Even after significant progress in the detection and management of pancreatic cancer the death rate has not significantly changed in the last five decades [102]. Of patients with distant cancer at the time of diagnosis, 97% are expected to die within five years. Since pancreatic cancer tends to remain silent for a rather long time, the chances of being diagnosed only in a late stage are considerably high. 80% to 90% of the patients are already incurable at the time of diagnosis, making it the sixth leading cause of cancer death in the world [70]. It is even anticipated that it will become the second leading cause of cancer mortality in 2030 [88].

Treatment for pancreatic cancer is possible by, for example, chemotherapy and/or surgery. However, due to the nature of the disease, the chances of recurrence are extremely high. One study found that 80% of the patients that underwent resection developed recurrence already within ten months [27]. This is one of the reasons the mortality rate is so extremely high, treatment is rather difficult, and recurrence is likely.

After resection of the pancreas, it is important to monitor the changes in the pancreatic tissue to determine if treatment has been successful. Currently, that has to be done manually by a physician. This is, however, an extremely time-consuming task and rather error-prone due to the low soft-tissue contrasts [67]. This is where a computer-aided diagnosis (CAD) system could become helpful. Especially during the last decade, there has been an increased interest in AI for these CADs in the medical field [7]. This is not surprising, since AI models can greatly reduce the cost of analyzing images. The design and training of a model can be costly and difficult, but once a model is trained, it simply only requires a computer to run on.

Medical image segmentation in particular is an essential part of cancer care and one of the most important medical image analysis tasks [28, 52]. Deep learning has caught a lot of attention and displayed some serious improvements in the state-of-the-art in many areas, including image segmentation [62]. Essentially, deep learning is a subgroup of machine learning, where neural networks are designed with one or more hidden layers.

One of the difficulties of automated organ segmentation using deep learning, especially with regard to the pancreas, is the huge variety in size, shape, and location of the organ from patient to patient [53]. This usually can be solved by making the model more complex, but that also requires more (annotated) data, which is very difficult to obtain in the medical domain.

Nevertheless, there has been plenty of research into the automatic segmentation of the pancreas using mostly deep learning approaches [12, 28, 58, 67, 77, 108]. These models are all trained and tested on CT scans of people that do not knowingly have any pancreatic diseases or that have undergone pancreatic resection. Generally, these models tend to have a dice score of up to 88%.

However, to the best of the author's knowledge, there is no research into the segmentation of the pancreas after a pancreatic resection has been undergone. The even higher variability in the pancreas is expected to make the segmentation even more difficult, especially with little to no open data when it comes to CT scans of the postoperative state. When an accurate model can be created, however, this can make a significant difference in the diagnosis of recurrence of pancreatic cancer and with that the effective treatment of it. That is why this research will focus on exactly this stage.

1.1. Research Objectives

For this master thesis, an effort will be made toward a more accurate segmentation of the pancreas on CT scans of patients that underwent pancreatic resection. More specifically it will attempt to answer the following research question:

> *To what extent can the pancreatic remnant be accurately segmented on CT scan images of patients that underwent pancreatic resection, leveraging the potential of deep learning?*

To be able to answer this question, several aspects need to be addressed. During the segmentation of objects in images in general, there are a lot of steps involved in going from data to actual segmentation, e.g. data augmentation, data preparation, training, etcetera. This research will aim to focus more on the following subquestions:

- *What deep learning variations of segmentation models are the most promising when it comes to segmentation on medical images?*
- *How can a deep learning segmentation pipeline be designed to improve the accuracy of the segmentation of the pancreatic remnant on CT scans of patients in the postoperative state?*
- *Can pretraining methods with, for example, preoperative data help improve the accuracy and robustness of the full segmentation pipeline?*

The main objective of this study is to design an algorithm that can outperform state-of-the-art pancreas segmentation in general, but more specifically in the postoperative state. If this would be the case, and the non-cancerous pancreatic remnant can be accurately segmented, this could be an important first step in distinguishing healthy versus tumorous pancreatic tissue after pancreatic resection, contributing to early detection of recurrent pancreatic carcinoma. Furthermore, it can shed new light on organ segmentation and effective pretraining of medical images in general.

There has been a significant substance of research on organ segmentation, especially in the last decade. Since pancreatic diseases are still among the most difficult ones in the medical world, naturally there has also been research on pancreas segmentation. However, this has been mostly focused on pancreas segmentation in the preoperative state or in healthy subjects. This research will contribute by researching what difficulties surgery can bring and what impact surgery can have on the accuracy of pancreas segmentation.

1.2. Followed Approach

To be able to find answers to the research question as described in Section 1.1, an experimental and iterative approach will be taken. Research on segmentation models will be conducted in order to find state-of-the-art models to draw inspiration from and compare with. From this research, a new model will be designed that will be compared with the state-of-the-art models mentioned before to assess the performance of these models. This will be done by conducting several experiments with different (hyper-)parameters, to measure the accuracy of the models. The accuracy will be measured by the Dice Similarity Coefficient (DSC) since this is one of the most used measures for accuracy in related research [9, 58, 81, 116]. In addition, other quantitative evaluation methods are added, e.g. Hausdorff Distance, 95^{th} percentile Hausdorff Distance and Normalized Surface Distance, but these do not necessarily prove the clinical usability. Because of that, results will also be shared with four medical experts to assess the accuracy from an expert point of view in a clinical context. Even though this can be very subjective and is not easily quantifiable and comparable, it can shed more light on the practical application.

One of the more difficult challenges for the segmentation of the remaining pancreas in the postoperative state is the extremely limited amount of data that is available. To increase the robustness and accuracy of segmentation models, often a pretraining step is added. In this step, a model is trained with different conditions, e.g. unsupervised versus supervised or on different data. The learned weights are then transferred to the actual segmentation model to have more knowledge about the domain already. For this research in particular, it might be possible to make use of the more substantial availability of annotated and unannotated CT scans of patients that have not had a pancreatic resection. This additional step could greatly increase the performance of the segmentation pipeline.

The postoperative data that will be used to train and test the algorithms was provided by the Regional Academic Cancer Center Utrecht (RACU), which consists of a dataset of 81 annotated CT scans

of patients who underwent pancreatic resection between 2014-2020, mainly for a benign indication. In patients who underwent resection for pancreatic cancer, only CT scans within the first four to six weeks after surgery were selected, as the aim was to segment healthy pancreatic tissue after resection. Furthermore, the open dataset from Roth et al. [107] or the larger dataset from Ma et al. [73] can be used in the earlier stages of the research as well as in the earlier parts of the segmentation pipeline, for example, to support pretraining.

In assessing the performance of the different algorithms, the accuracy will be weighted more heavily than the runtime. Currently, it can take months before a proper diagnosis can be made of recurrent pancreatic cancer. This is due to the extreme similarity between the fibrosis that develops after the surgery and the cancer tissue on the CT scan. Only after a longer period of time, it can be seen how the tissue develops to categorize it correctly. An algorithm that has mediocre accuracy adds less value to this process, since there can only be a diagnosis when there is enough certainty about the state of the pancreas. Even if an algorithm would run a day longer to achieve significantly higher accuracy, this would be far more beneficial than having a relatively quick algorithm with lower accuracy. That being said, runtime performance is still important, especially during this research, since the time for evaluating and designing the algorithm is limited.

2 Background

In the following chapter, there will be given more background on different aspects that are deemed important to conduct research aimed at the objectives as described in Section 1.1. In particular, more background will be provided for the technology and characteristics of CT scans, different methods that can be used for training a machine learning model (e.g. supervised, unsupervised), different operations that are important in deep learning networks, types of deep learning models that are currently used in state-of-the-art segmentation pipelines, and a short review of different state-of-the-art pancreas segmentation pipelines.

2.1. Data

One of the most important things to analyze when the goal is to create a deep learning method is the data. For this particular research, the data consists of CT scans of healthy people and people that have had a pancreatic resection. These CT scans include annotations for the pancreas. In the following section, more background will be given to the specifics of a CT scanner, the characteristics of the pancreas within the CT scan, and ways to evaluate a model such that the results are meaningful.

2.1.1. CT Scans

In essence, a CT scan is not more than a more elaborate and extensive version of an X-ray. Instead of only creating an image of one point of view, multiple images are taken from different angles all around the subject as can be seen in Figure 2.1a. The X-ray tube circles around the subject and shoots beams of radiation from different angles around the subject, as can be seen in Figure 2.1b, while the subject is slowly moving through the scanner. On the other side of the beam are receptors that measure the amount of radiation that has traveled through the body.

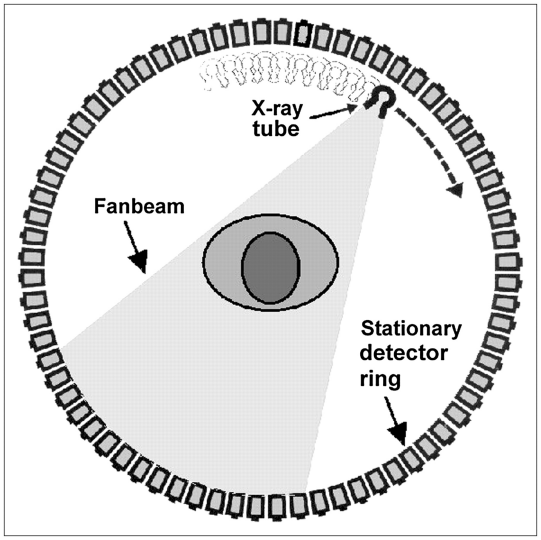
The fact that a CT scan is in essence not more than an X-ray brings similar properties. What is measured is the attenuation of the X-ray beam by different matter. Matter with higher attenuation, for example bone tissue, will attenuate or absorb more of the beam, causing less radiation to be received by the receptors. Specific tissue has specific attenuation coefficients. Because of this, bone tissue will have the same value on a CT scan image for every CT scan that is made, no matter with what device it is created [1].

The intensity of the detected radiation is described in Hounsfield units (HU). The HU is defined by the following equation:

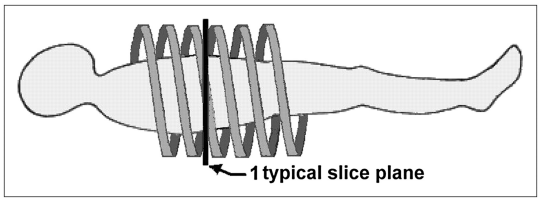
$$HU = 1000 \times \frac{\mu_{tissue} - \mu_{water}}{\mu_{water}}$$

where μ_{tissue} is the attenuation coefficient of the tissue and μ_{water} the attenuation coefficient of water. This definition has as a consequence that all types of tissue have a specific range of HU values that characterize the tissue type and is equivalent no matter what kind of CT scanner one uses. The scale goes from -1000 HU for air to more than 3000 HU for metals like steel or silver [22].

Important characteristics of CT scans, in particular when used in machine learning, are the slice thickness and pitch. The slice thickness is specified by the beam width of the X-ray tube and represents the length in the z-axis of every voxel. The pitch is defined as the table movement per rotation divided by



(a) Geometry of a CT scanner



(b) Spiral path of the X-ray tube

Figure 2.1: CT scanner technology with images from Goldman [34]

the slice thickness. If the table moves 10 millimeters per rotation and the slice thickness is 10 millimeters as well, the pitch would be 1.0. A pitch greater than 1.0 means that there exists a gap between X-ray beams of consecutive rotations with reduced accuracy and coverage as a consequence. A pitch smaller than 1.0 has the opposite effect. There would then be an overlap in beams with increased accuracy as a consequence. Typically, the pitch is somewhere between 1.0 and 1.5. Pitches greater than 1.5 occur rarely and pitches greater than 2.0 usually yield unacceptable results [1].

2.1.2. Data Characteristics

The data for this particular problem will be CT scans from different scanners with different parameters. The resolution of the slices is 512 by 512 pixels, but the amount of slices varies as well. As discussed before, the CT scans will be in Hounsfield units, possibly covering the full spectrum from -1000 to over 3000, but this makes processing rather difficult. An often-taken approach [72, 99, 16] is to cap the values to [-100, 240], which can then be normalized to [0, 1]. The capping of the values is done to magnify the contrast between soft tissue, e.g. pancreatic tissue and its surrounding tissue, to be able to better visualize the borders between different types of tissue. Normalizing to [0, 1] eases the training process, e.g. by preventing gradient overflow.

The annotations are created manually and per slice by an experienced radiologist. One of the difficulties with the annotated data that is used for training, is the fact that there can be quite some inter- and intra-rater variability as will also be discussed in the next subsection.

Another issue was also identified by Ma et al. [74], namely the huge data imbalance. There are many more background voxels than there are pancreatic voxels, on average even 503 times more in the NIH dataset. Some slices do not even include a single pancreatic pixel, making it more difficult to find a balance in training an algorithm, but also in evaluating an algorithm.

In order to overcome these issues, various approaches focused on the underlying model architecture as discussed in Section 2.5, where more insights are given on how the general architecture of the model can help in this. In Section 2.3, deeper insights are given on how specific operations within an architecture can be adjusted to address the data difficulties as well, e.g. different loss functions.

2.1.3. Evaluation Metrics

AI models can be evaluated through many different metrics. In fact, the application determines reasonable metrics. Think of a classifier that always predicts one class. If there is a huge imbalance in the data where 99% of the data points are to be classified to that class, the naive classifier will predict the correct class 99% of the time. This sounds really good, but if one's interest is in classifying the 1% class more accurately, the naive classifier is terrible. Because of this reason, one has to be cautious

about what metrics to choose in order to evaluate one's model. Choosing less meaningful metrics can give the impression that a model is performing well, even though it is useless in practice or vice versa.

Since the data consists of medical images and the annotation, i.e. the ground truth, is done by hand, all evaluation metrics should be handled with care. The images are generally not annotated pixel by pixel, if that could even be done reliably, and there can be inter- and intra-rater variability. The used ground truth, therefore, is not likely to be 100% accurate. Achieving a high score on any evaluation metric is therefore not automatically a truly well-performing model. The annotation process is an important step that should be given a lot of care. Some evaluation metrics can take a certain variability into account to better address this as well. This subsection will go into more depth on different evaluation metrics that are relevant for the segmentation of the pancreas on CT scans, the specific things they measure, and the advantages and disadvantages of the different methods.

Basic metrics

Some of the most simple metrics that can be used are simply the pixels that are correctly or wrongly classified. Four categories are important for this: *True Positives* (TP), *False Positives* (FP), *True Negatives* (TN), and *False Negatives* (FN). *Positive* in this context means that the pixel is part of the pancreas according to the prediction, whereas *Negative* means it is not part of the pancreas. *True* and *False* correspond to a correct and wrong classificiation respectively. In other words, the TP region consists of all the pixels that are part of the pancreas and classified as such, whereas the FN region consists of the pixels that are part of the pancreas and not classified as such.

In line with these basic measures, there are four often-used, generic measures: sensitivity, specificity, precision, and accuracy. Sensitivity (or recall) is defined as $TP/(TP+FN)$ and measures the ratio of correctly classified pancreatic pixels over the entire region of pancreatic pixels. This measure is therefore designed to indicate how few pancreatic pixels the model misses in its prediction. Specificity is defined as $TN/(TN+FP)$ and measures somewhat the opposite ratio, namely the one of the correctly classified non-pancreatic pixels over the whole region of non-pancreatic pixels. It gives more insight into what extent the model is capable of classifying all background pixels as such. Precision can be defined as $TP/(TP+FP)$ and results in the ratio of correctly classified pancreatic pixels over the total number of predicted pancreatic pixels. It gives more insight into how many of the predicted pancreatic pixels are really pancreatic according to the ground truth. Accuracy is defined as $(TN+TP)/(TN+TP+FN+FP)$ and measures the ratio of all correctly classified pixels over the total number of pixels. Accuracy is essentially a naive approach to measuring how many pixels are correctly classified. The problem with accuracy for this research specifically is the fact that it does not take into account the imbalance of the pancreatic and background pixels. With very few pancreatic pixels, this metric can not give meaningful insights into the performance of a model. Contrarily, it can give rather wrong impressions.

Dice Similarity Coefficient (DSC)

The DSC metric is one of the, if not the, most used metric for pancreas segmentation models as will be seen in Section 2.5. It is defined as follows by [60]:

$$\frac{2 \times TP}{2TP + FP + FN} = \frac{2\sum_i^N p_i g_i}{\sum_i^N p_i + \sum_i^N g_i}$$

where N is the total number of classified pixels and p_i and g_i the predicted and ground-truth class of pixel i respectively. This metric is more invariant toward data imbalance since it only measures the performance on the class of interest. The DSC can range from 0 to 1, where a DSC of 1 means a perfect prediction. One of the disadvantages of the DSC is the fact that it does not account for the spatial distribution. This means that the shape of the prediction can be completely different from the ground truth. Furthermore, the DSC is not boundary-aware, i.e. a prediction with a good DSC can easily spread across multiple organs, which can make, for example, therapy planning rather difficult.

Hausdorff Distance (HD)

Another often-used metric is the Hausdorff Distance. It measures the maximum distance between two finite point sets. Formally, it was defined by Huttenlocher, Klanderman, and Rucklidge [46] as follows. Given two finite point sets $A = \{a_1, ..., a_p\}$ and $B = \{b_1, ..., b_q\}$ the HD is defined as

$$H(A, B) = \mathsf{max}(h(A, B), h(B, A))$$

where

$$h(A, B) = \max_{a \in A} \min_{b \in B} ||a - b||$$

and $|| \cdot ||$ is some underlying norm on the points of A and B. In this specific application, the Euclidean distance is one of the more sensible norms. HD can range from 0 to, theoretically, infinity, where an HD of 0 would be perfect.

What the HD essentially measures is the maximal distance between any point in A and B. The benefit of this metric is that it can more accurately measure to what extent the prediction is differently shaped than the ground truth since it is more boundary-aware. One disadvantage of HD is the fact that it does not represent how many pixels or what type of pixels are well-classified. A low HD indicates a well-predicted shape but does not give any information on what pixels are wrongly classified as, for example, the basic metrics would do.

Another aspect of the HD is that it greatly penalizes outliers. A single wrongly predicted pixel can increase the HD by magnitudes, even if all other pixels are correctly classified. To overcome this issue, the X^{th} Percentile HD was designed, where X can be any percentage from 0% to 100%. The 100% Percentile HD (HD100) is equal to the standard HD. An often-used value for X is 95%, resulting in HD95. Whereas HD calculates the *maximum symmetric surface distance*, HD95 only calculates the 95% percentile symmetric surface distance. In Figure 2.2 one can see the effect of a spatial outlier on the different variants of HD.

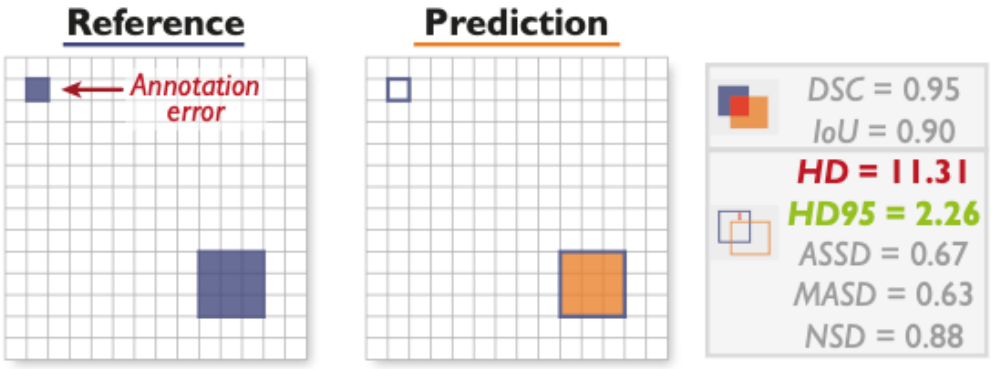


Figure 2.2: Comparison of the HD, HD95, and other evaluation metrics on an example with a spatial outlier. [76]

Normalized Surface Distance (NSD)

As mentioned earlier, one of the difficulties of working with annotated medical images is the human factor in the annotations. in Figure 2.3, one can see a 3D visualization of a 2D annotated pancreas. It can clearly be seen that the 3D shape consists of multiple stacked slices that most certainly do not always align at the borders. This is due to the fact that the CT scans are annotated slice per slice. This is partly due to the low resolution in the Z-axis of a CT scan, but also because of the intra-rater variability as earlier discussed.

The Normalized Surface Distance [90] is designed to account for this variability at the border of the region of interest. It essentially measures the DSC at the surface voxels. It is formally defined as follows:

$$R_{i,j}^{(\tau)} = \frac{|S_i \cap B_j^{(\tau)}| + |S_j \cap B_i^{(\tau)}|}{|S_i| + |S_j|}$$

where i and j denote the prediction and ground truth respectively, S is the surface, i.e. the border of a mask, B is the border region, and τ is a hyperparameter that represents the tolerance of the border region. A larger τ results in a larger or thicker border region, in which the true surface should be in order to be considered correctly classified. The different definitions are visually represented in Figure 2.4. As can be seen, NSD is boundary-aware and takes into account the variability in the annotation or ground truth. Just as with the HD, NSD can not represent the prediction performance of all pixels as DSC can, but it can evaluate the border without the noise from human-made annotations.

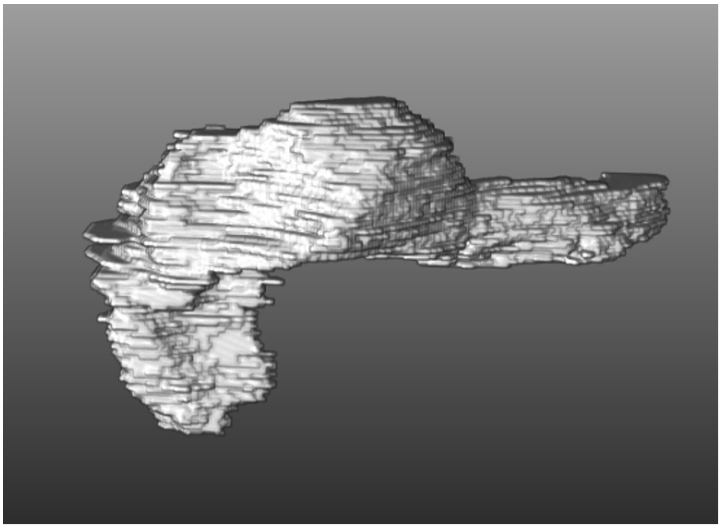


Figure 2.3: 3D visualization of the mask of one of the NIH CT scans

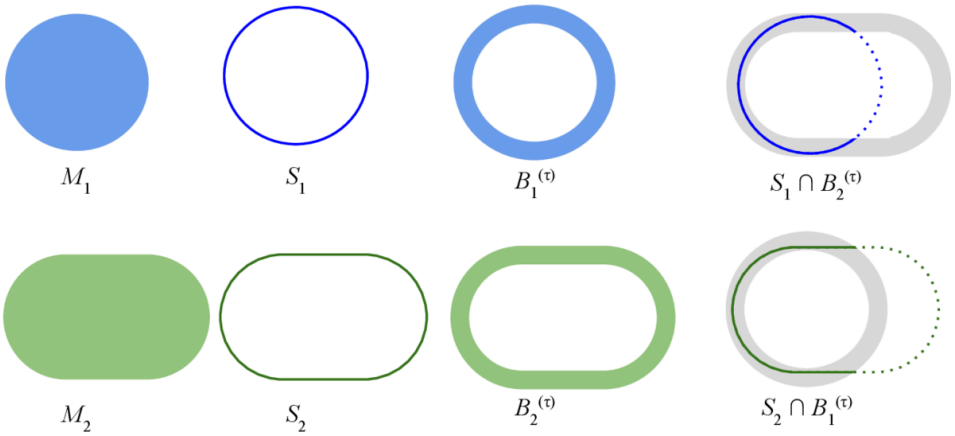


Figure 2.4: Visual representation of two masks, surfaces, border regions and their overlapping [90]

2.2. Machine Learning Methods

There are many different applications to which machine learning can be applied. The vast variety consequently also brings a wide variety of types of data and problems. Most noticeably there is the distinction between supervised, unsupervised, and reinforcement learning. Combinations of the three also occur very often. In this section, the different types are explained.

2.2.1. Supervised Learning

Supervised learning requires data that is annotated. This means that there is a dataset available that, apart from the raw data, also has labels that are essentially the correct answer the algorithm should learn to give. This method is often used in many different scenarios, for example, handwriting recognition. A great dataset of images of letters and/or numbers is given to the model, including a label for every image with the correct classification of the letter/number. The model can use these as examples and can learn what the specifics of an “a” are so that the next time it encounters an image of an “a” it can correctly classify it as such.

The benefit of this approach is that machine learning can be applied to many different domains, simply because it will learn from the labels that you will feed it. A disadvantage of this approach, especially in the medical domain, is that a model requires one to annotate the dataset to be able to learn all characteristics and achieve a satisfiable accuracy and robustness. This can make it difficult, time-consuming, and/or costly to acquire enough data.

2.2.2. Unsupervised Learning

The unsupervised approach is different from the supervised one since it does not need any labels or annotations. It simply needs data from which it can derive patterns. This is widely applied in clustering problems. The assumption is that data from the same category are similar in features or values. The model can discover patterns within the data and separate different groups consequently.

One big advantage of an unsupervised approach is the fact that data does not need to be labeled. Therefore, the data can be gathered more easily and does not need to be reviewed by an expert in the domain. A disadvantage is that this can only be applied if the underlying assumption that there is a clear pattern is true. If there is no clear distinction between different outcomes or no distinction that can be learned from the data at all, an unsupervised approach will not be able to give satisfactory results.

2.2.3. Reinforcement Learning

Furthermore, there is reinforcement learning which is a method for online learning from experience. It is not needed to have specific labels for data. Models based on reinforcement learning adjust their results based on the reward that action brings. An application where reinforcement learning is widely used is in multi-agent systems. Imagine a reinforcement learning model that needs to find its way through a labyrinth. If it comes out of the labyrinth, there is a big reward. The more steps it needs to take to get out of the labyrinth, the lower the reward. The model or agent learns from many different tries if it is best to go left or right at particular junctions and will assign rewards for the intermediate steps. Eventually, the agent will learn what is the best way to walk through the labyrinth, while maximizing the value.

An advantage of reinforcement learning is that it only requires a reward function to determine the reward for its actions or results. Apart from that it simply requires an environment and time to learn. One of the challenges is that a reinforcement model needs a good exploration mechanism to have a trade-off between the exploitation of its learned knowledge and exploration to prevent local minima. It can be very tricky to find such a mechanism. Furthermore, it does not necessarily scale well if the number of states increases.

2.2.4. Self-Supervised Learning

Self-supervised learning generally consists of two parts. Firstly, a surrogate task is performed using an unsupervised approach. Secondly, a supervised approach is used to fine-tune the previously trained model for the task that actually needs to be solved. The previous project within Datacation [24] as earlier described also made use of a self-supervised approach. At first, distorted non-labeled images were reconstructed. Here the model could learn properties that are specific to CT scans, without the need for annotated data. It could simply learn what a CT scan looks like and what the structure of a body is. This information is transferred to the second step, where a supervised model could fine-tune this knowledge and make it specific for the segmentation of the pancreas.

The great advantage of this approach is that less annotated data is required, while still coming closer to the results of a supervised approach when compared to an unsupervised approach. One of the difficulties is finding surrogate tasks that indeed bring useful information to the model, which can be transferred to the actual task.

2.2.5. Semi-Supervised Learning

Another variant is semi-supervised learning. It relates to self-supervised since this is also an approach that makes use of both labeled and unlabeled data. However, in semi-supervised learning, both the labeled and the unlabeled data are given to the same model, instead of having separate tasks. There are two approaches to semi-supervised learning. One can either use transductive learning or inductive learning, where the goal is to infer the correct labels for the given unlabeled data or to infer the correct mapping from X to Y respectively.

Semi-supervised approaches are usually implemented as follows. Firstly, a classifier is trained based on the labeled data. This classifier will then classify all unlabeled data resulting in so-called pseudo-labels. The unlabeled data with the most confident pseudo-labels are then combined with the labeled data into a new dataset. This new dataset is then used to train the actual classifier. In a way, a semi-supervised model creates its own labels for the unlabeled data.

Just like with self-supervised learning, semi-supervised approaches are useful for problems where the annotation of data is costly, but unlabeled data is relatively cheaply available. However, the labeled

data does need to be representative of the unlabeled data. If there is too much difference or too little labeled data, the semi-supervised approach will not yield good results.

2.3. Neural Network Operations

Artificial Neural Networks (ANN) architectures consist of a variety of different operations and functions, such as activation and loss functions. More specifically, a Convolutional Neural Network (CNN) is an ANN that incorporates convolutional operations.

These essentially are a re-estimation of a value, based on the surrounding input values. One simply takes a small matrix, called a kernel or filter, and applies that to every input value. One of the most simple operations arguably could be a blur operation. Imagine a kernel of the size 3x3, which looks as follows:

$$\begin{bmatrix} \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \end{bmatrix}$$

The value in the middle would represent the pixel of interest. It will be assigned a value according to all weights in the matrix, i.e. it will get $\frac{1}{9}^{th}$ times the value of the pixel to the top right, $\frac{1}{9}^{th}$ the value of the pixel above, etcetera. This kernel will move over all pixels, changing the values accordingly.

CNNs can be further extended with other operations, such as regularizations or attention mechanisms. All these different operations, from the basic functions to the CNN extended functions, can have a different impact on the performance and behavior of a network. To truly understand the impact on the model and in order to create a model that is fully specialized on the problem at hand, a deeper understanding is to be created about the individual operations. This section specifically aims to do that, by going into depth on the different operations and variants that have been presented in prior research.

2.3.1. Activation Function

An ANN essentially consists of many layers consisting of multiple neurons. Each (hidden) neuron receives input from other neurons that it weighs accordingly, e.g. as a weighted linear combination of the input. After this, there is an activation function determining whether or not the neuron should be "activated", i.e. what weight should be given to the output of this neuron. As can be seen in Figure 2.5 the result of the activation function is the output of the neuron. This activation function tends to be a nonlinear function, since this will make the ANN nonlinear as well, making the ANN capable of learning more complex distributions [98].

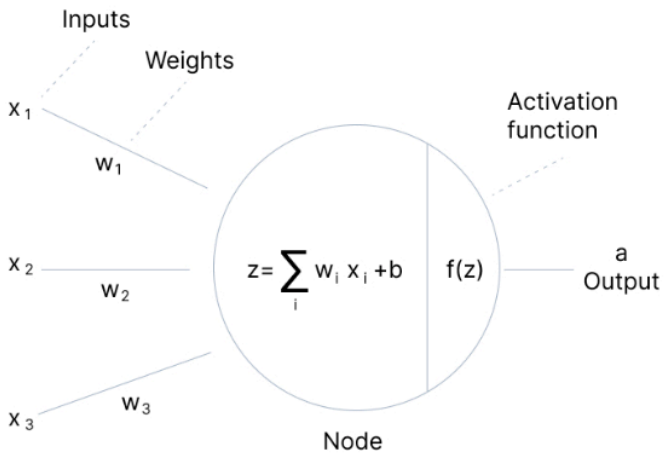


Figure 2.5: The internal working of a node [98]

One of the most often used activation functions within ANNs is ReLU [87] or a variation of it. Mathematically, this function can be represented as follows:

$$ReLU(x) = max(0, x)$$

In other words, this function returns the original value if it is positive, and 0 when it is negative. ReLU is an incredibly simple function to compute, reducing the computational costs when compared to logistic sigmoid or tanh functions that were used before. It, furthermore, has no *vanishing gradient problem* for positive values, i.e. the gradient does not become close to zero for larger values preventing a slow convergence. However, for negative values, it does have this problem since they are simply ignored. This can lead to "dead" neurons, that will never be activated and can impact the convergence negatively.

Many different methods have tried to overcome this by changing the function slightly to have a nonzero gradient in the negative space, e.g. LReLU [75], PReLU [39], RReLU [39], BReLU [66], MTLU [35]. However, these variations do not necessarily bring better performance but might bring higher complexity and therefore higher runtime [4]. Recently, more attention has been given to learnable activation functions, in order to learn the right parameters for the underlying nonlinearity [4]. Again, however, there does not seem to be a single activation function that is Pareto-optimal compared to others in terms of accuracy and runtime, as the well-known no-free-lunch paradigm indicates.

It does seem that Swish [101] or, a variant on Swish, E-Swish [2] does slightly outperform ReLU for many different problems, while keeping the simplicity, according to their initial papers. However, the Swish is not consistently outperforming ReLU in other research [86, 26] which can be due to a different network architecture that suits the activation functions differently. Swish and E-Swish are defined as $f(x) = x \times sigmoid(x)$ and $f(x) = \beta x \times sigmoid(x)$ respectively. Both can easily replace ReLU functions, making it rather easy to implement them and due to the learning parameter they can optimize a little bit better, while not being bound in the negative space.

Mish [82], being in the same family as Swish, was found to perform slightly better than Swish and many other activation functions. Its definition is $f(x) = x \tanh softplus(x)$. The computational costs are, however, significantly higher, but the variation of Mish-CUDA does seem to perform relatively similarly to ReLU when it comes to computation time.

The creators of the Padé Activation Unit (PAU) [86] tried to overcome the reliability in performance on the hand-picking of an activation function. With their PAU activation function, they designed an activation function that can learn the most optimal activation function. The big advantage of this method is that it can automatically learn its parameters to very accurately approximate many different activation functions, e.g. ReLU, Swish, and LReLU. Therefore PAU can easily replace other activation functions while having an equally good or better performance. The authors did not indicate to what extent the training time is affected, but from [26] it seems it does come at a computational cost.

The Parametric Deformable Exponential Linear Units (PDELU) [17] is yet another activation function. As the name suggests, PDELA is part of the exponential activation functions. It is aimed to push the mean value of activation closer to zero to ensure the steepest descent in the training of deep neural networks. Essentially, PDELU is a variant of ELU [19] which was designed with an exponential term in the negative space in order to push the mean closer to zero and thus speed up learning. In the positive range, ELU is equal to ReLU. With PDELU, the mean is aimed to be pushed even closer to zero and thus improving the convergence speed. It is defined as follows:

$$f(x_i) = \begin{cases} x_i & \text{if } x_i > 0 \\ \alpha_i \times ([1 + (1-t)x_i]^{\frac{1}{1-t}} - 1) & \text{if } x_i \leq 0 \end{cases}$$

The parameter α_i is learned by the loss function and can differ per feature channel. In their experiments, they were able to achieve better performance than ReLU and many different variations of it, which was also found by Dubey, Singh, and Chaudhuri [26]. However, Dubey, Singh, and Chaudhuri [26] also found that the training time is extremely longer than other activation functions. This is probably due to the more expensive exponential calculations and the learning on them.

The previously mentioned activation functions have in common that they are not data dependent, i.e. they behave exactly the same independent of how the data is distributed. Average Biased ReLU (ABReLU) [25] aims to make the activation function dependent on the input volume. It can be formulated as $f(x) = max(0, x - \beta)$, where β is the average of the input volume. It essentially is a horizontally shifted version of ReLU. This ensures that if negative values are more important in the input volume, the important negative values can actually be used. Contrarily, if there are many positive values, the less important positive values can be ignored. It was found that this method can achieve state-of-the-art performance, while also keeping the computational cost somewhat limited.

2.3.2. Loss Function

During the training of an ANN, it is important for the ANN to know when it is doing well. To facilitate this, the ANN makes use of a loss function to determine its performance during the training phase. This loss function is then used to optimize all its parameters. Loss functions come in many different forms, designed for all kinds of problems. They can be divided into four different categories based on what they are derived from: distribution-based loss, region-based loss, boundary-based loss, and compound loss, where compound loss is a combination of loss functions from the other three categories [74]. Within medical image segmentation, for example, the most often used options are the Dice loss and cross-entropy loss which are region-based and distribution-based loss functions respectively [79]. One can not simply take an evaluation metric as described in Section 2.1.3, since a loss function should be differentiable for the model to know which way to go. Some evaluation metrics are, however, differentiable, sometimes after small modifications. That is why some will be discussed in this subsection and some will not.

Distribution-based loss

The main distribution-based loss function is cross entropy (CE). CE is used in many different medical segmentation tasks and is formulated as follows:

$$\mathcal{L}_{CE} = -\frac{1}{N}\sum_{c=1}^{C}\sum_{i=1}^{N} g_i^c \log s_i^c,$$

where g_i^c and s_i^c are the ground truth and predicted class c of voxel i, N the number of voxels, and C the number of classes, which is two in the case of the pancreas segmentation, i.e. either 0 or 1. The CE loss computes the difference in the distribution of the ground truth and the prediction. In the experiments ran in Ma et al. [74], CE performed moderately well on the NIH dataset [107]. One of the biggest challenges for pancreas segmentation is the huge imbalance of foreground and background pixels, as already mentioned in Section 2.1.2. This is also the reason that CE had a difficult time. Due to the formulation of CE, the ANN learns equally from every individual pixel which is not desirable since it learns more from the background pixels than the foreground ones [123]. To better address data imbalance, weighted cross entropy (WCE) is often used. Essentially, WCE is equal to CE with the difference being that every pixel is now weighted by a factor w_c based on its class in order to overcome the disadvantages of the low amount of foreground pixels.

A different approach was taken in Top-K loss [133]. Top-K loss focuses on the difficult voxels and ignores the pixels with high-certainty predictions. There are two approaches. One only calculates the CE on the voxels with a probability value higher than a threshold t. The other only calculates the CE of the k% worst voxels. The latter was found to be the best performing and most robust on highly imbalanced data in [74].

Instead of ignoring the easily classified pixels completely, focal loss (FL) [68] reweights the voxels based on the prediction probability. It is formulated as follows:

$$\mathcal{L}_{FL} = -\frac{1}{N}\sum_{c=1}^{C}\sum_{i=1}^{N}(1 - s_i^c)^{\gamma} g_i^c \log s_i^c,$$

where γ controls the strength of the reweighting. The higher the value of γ, the more the loss of easy predictions is discounted. The idea is that this helps in the class imbalance in the dataset. However, in the experiments run in Ma et al. [74], FL does not perform very well.

Region-based loss

Undoubtedly, the most well-known and well-used region-based loss in the field is the Dice loss (DL). There are multiple definitions and implementations for the DL. The terms in the denominator can namely be either squared or not. Even though they do not differ too much, it was found that for pancreas segmentation the squared variant performs slightly better than the non-squared variant [74]. This can be explained by the data imbalance. The squared variant tends to focus more on bad predictions when compared to the non-squared variant. It is defined as follows:

$$\mathcal{L}_{DL-squared} = 1 - \frac{2\frac{1}{N}\sum_{c=1}^{C}\sum_{i=1}^{N} g_i^c s_i^c}{\sum_{c=1}^{C}\sum_{i=1}^{N}(g_i^c)^2 + \sum_{c=1}^{C}\sum_{i=1}^{N}(s_i^c)^2}$$

Implementation-wise DL is either computed by sample or by batch. Sample DL computes the DL for every sample in the batch and then averages across the batch. Batch DL on the other hand treats the whole batch as a pseudo-volume, calculating the DL as if it were one sample. In combination with the squared definition, the batch approach was found to be outperforming the other combinations on the pancreas dataset [74].

Another well-known region-based loss is the intersection over union (IoU) or Jaccard index. In essence, the IoU is very similar to the DL. They are even always positively correlated, i.e. if a classifier A outperforms another classifier B in one of the metrics, A will also beat B for the other metric. The difference is in the way they penalize instances of bad classification. IoU penalizes the worst cases more than DL, meaning the bad predictions are weighted more heavily. In the experiments of Ma et al. [74], the DL does slightly outperform IoU, in particular the squared DL variant, on the pancreas dataset.

Boundary-based loss

Even though there are strategies to overcome the data imbalance, which highly influences the results for distribution-based and region-based losses, such as taking an equal amount of samples, in general, distribution-based and region-based losses will be harmed by data imbalance. In boundary-based losses, the integrals are computed over the interface between regions, mitigating the data imbalance issue [57].

The first approach is the boundary loss (BL) of Kervadec et al. [57]. One of the difficulties of a boundary loss is the fact that it is not trivial to represent boundary points of a prediction as a differentiable function, something that is necessary for a network to use it to train. In BL, a level-set approach was taken to overcome this challenge. As can be seen in Figure 2.6, the BL essentially sums linear functions, i.e. the distance from the boundary of the ground truth to the boundary of the predicted region. It is defined as follows:

$$\mathcal{L}_{BL} = \sum_{\Omega} \phi_G(p) s_\theta(q)$$

where ϕ_G is the level set function and $s\theta$ is the softmax probability output of the network. One of the problems of BL is that an empty foreground prediction will lead to very low gradients, meaning that this loss function might get stuck into local minima. In particular, this is a problem at the beginning of training. The authors acknowledge this and state that BL can best be used in a dynamically weighted combination with a region-based loss in order to overcome this.

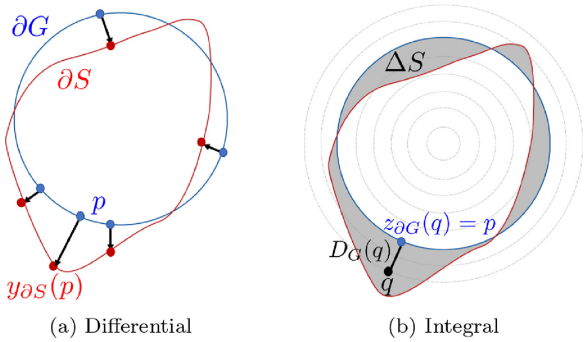


Figure 2.6: Representation of boundary loss [57]

One often-used evaluation metric is the Hausdorff Distance (HD), i.e. the maximum distance from one boundary to the other. Even though the HD is not differentiable, and therefore cannot be used directly as a loss function, Karimi and Salcudean [54] adjusted it slightly to make it suitable as a loss function. Distance transforms were used in order to find the maximum distance from one shape to the other. This can be done either one-sided, similarly as in BL, from the ground truth to the prediction, or two-sided, i.e. both from the ground truth to the prediction as well as vice versa. One way of reducing the computational cost is to use the one-sided variant since the distance transform of the ground truth only has to be computed once whereas the distance transform of the prediction needs to be computed for every prediction. The one-sided distance transform HD loss can be formulated as follows:

$$\mathcal{L}_{HD} = \frac{1}{N} \sum_{c=1}^{C} \sum_{i=1}^{N} \left[(s_i^c - g_i^c)^2 \circ d_{s_i}^{\alpha} \right],$$

where α is the parameter that determines how strongly larger errors should be penalized. Again, however, training on just the HD loss can be very unstable, which is why it can best be combined with other losses.

Compound loss

Since every loss function has its advantages and disadvantages, there has also been research into combinations of more than one loss function in order to get the best of all. Generally, these loss functions are not fundamentally different, but just a (weighted) sum of the two or more functions. Combo loss [122] combined DL and CE loss in the following way: $\mathcal{L}_{DiceCE} = \mathcal{L}_{Dice} + \mathcal{L}_{CE}$. The reasoning behind it is that the DL aspect can prevent the algorithm from getting trapped in local minima, whereas the CE can simulate better model parameters by penalizing false positives and negatives. With similar reasoning, [146] and [10] designed a combination of DL and FL, and DL and Top-K respectively. Both losses were defined by simply adding the two components equally weighted.

To focus more on less accurately predicted structures and overcome the data imbalance at the same time, [130] designed the exponential logarithmic loss (ELL). It is defined as:

$$\mathcal{L}_{ELL} = w_{Dice}E[(-\ln(Dice_c))^{\gamma Dice}] + w_{CE}E[w_l(-\ln(s_i^c))^{\gamma CE}]$$

where $Dice_c = \frac{2\sum_{i=1}^{N} g_i^c s_c^c + \epsilon}{\sum_{i=1}^{N}(g_i^c + s_c^c) + \epsilon}$, w_{Dice}, w_{CE} are the weights for DL and CE respectively, $\gamma Dice, \gamma CE$ control the nonlinearities, $E[\cdot]$ represents the mean value with respect to the class and voxel and w_l a weight for reducing the influence of more frequently seen classes. In particular, this loss function allows for greater control of the nonlinearities, while accounting for the disadvantages of both a region-based as well as a distribution-based approach.

As was already mentioned in the original papers of BL and HD loss, both loss functions are not well-suited for individual usage. Both were combined with a regional loss as DL. In Kervadec et al. [57] they rebalanced the combination of the losses during the training phase. The total loss was defined as $\mathcal{L} = (1-\alpha)\mathcal{L}_{DL} + \alpha\mathcal{L}_{BL}$ where α starts close to 0 and then gradually becomes larger over time. This helps in using the DL as a stable factor at the beginning of training, while the model will still finetune more on the BL later in the training. This was found to be performing significantly better than using a constant α or an increasing α while keeping the weight of $\mathcal{L}_{\mathcal{DL}}$ constant at all times.

In Karimi and Salcudean [54] a slightly different approach was taken. The weight was still rebalanced during training, but they changed it as follows. After every epoch, the HD loss and DL were computed. The weight for the next epoch would then be the ratio of the mean of the HD loss and the mean of DL over all training samples. This resulted in both loss terms getting equally weighted. This approach was reported to work well in the experiments they ran.

2.3.3. Pooling

One of the most occurring operations within (deep) CNNs is the pooling operation. Generally, this operation is applied in every layer. Pooling tries to scan the feature map and aggregates the information within a local area, depending on the specific variables. Max-pooling, for example, takes only the maximum value of an area to the next layer. This new representation of the data tends to make a model more robust to translations and distortions to a certain extent [120].

In Mansouri et al. [78] two adjusted versions of mode pooling are designed, fully-connected mode pooling and Mode-Fischer pooling (MF). One of the significant advantages of using mode pooling is the preservation of spatial and local appearance information, while also retaining the largest number of real values in the new representation. They were able to outperform other pooling techniques ranging from 1% to 10% with both versions. The fully-connected mode pooling achieved the lowest error rates, but with the MF approach, they were able to also reduce energy savings for the computations.

Pooling operations, as applied in most CNNs, are hand-crafted, i.e. upfront the creator of the network defines the weights of the operation. For average-pooling, this would mean that it is always the average that is taken. Springenberg et al. [117] suggests that it might not necessarily be beneficial to have these hand-crafted operations. Instead, they suggested using a convolutional layer with an adjusted stride instead, which yielded similar results. The advantage of this is that the network can actually learn the best weights for the pooling-like operation, possibly increasing the accuracy. One of the disadvantages of the approach taken by Springenberg et al. [117] is the fact that there is quite an increase in learnable parameters.

Sun et al. [120] suggested a learning pooling (LEAP) operation, which yields comparable results with a fraction of the parameters. The traditional convolutional operation requires $W \times H \times M \times N$ parameters, where $W \times H$ is the kernel size and M and N are the number of input and output feature channels, respectively. They propose the LEAP operation, which firstly uses a filter of size $W \times H \times M$ for an intermediate response channel, after which a filter of $M \times N$ is applied, which results in a total of $W \times H \times M + M \times N$. Because of the size of all variables, this tends to be a much lower amount of parameters.

Another approach to solving the limitation of one hand-crafted method was taken in Guo et al. [37]. They designed an active selection pooling (ASP) method, which in essence simply learns which pooling operation to apply. It is based on the human visual system, as it adjusts its strategy based on the application. In their approach, they could achieve similar performance levels as the state-of-the-art pooling methods and for specific problems even outperform them.

2.3.4. Padding

One might now understand how to apply a kernel to most pixels in an image. However, it gets more complicated at the borders of the image. A border pixel, namely, does not have neighbors in every direction. This means it would be impossible to multiply the neighbor values with the kernel to get the right value for the border pixel. One could simply not calculate the values for the border pixels and ignore them, but this would result in a different output size, especially deeper in the convolutional networks this can be a big problem as the border pixels represent a larger area of the original image due to the downsampling operations. To prevent the image size to decrease, a technique called “padding” is used. Artificial pixels are used that extend the borders of the image to make sure the actual border pixels still can be calculated using the kernel.

As shown in Islam et al. [49], padding can have a great impact on the segmentation performance in CNNs. They showed that different padding techniques result in different levels of position information, especially at the borders of an image. It was found that networks with padding consistently outperformed networks without padding, further showing the importance of giving thought to the padding technique used.

One of the most often used padding techniques is zero padding, where zeros are added along the borders, but this can greatly alter the input distribution around the border region [89]. Other often used methods are reflection padding, where the input is reflected at the borders, replication padding, where the values of the border pixels are extended, or mirror padding, where the pixel values of the border pixels on the opposite side of the image are used. Usually, these methods all yield similar results [138].

Partial convolution-based padding was designed by Liu et al. [71] to overcome the undesired effects at the border of the input. They essentially reweighted the convolution results near the borders by treating the padded pixels differently than the actual input pixels. During their experiments, they were able to consistently outperform zero padding for a number of different configurations.

The padding problem can also be described as an image extrapolation problem, as done by Huang, Proesmans, and Van Gool [43]. As such, they modeled the padding as a local image extrapolation. They defined a small region near the border to minimize the GPU memory used and fed an input where the original region is cropped and the cropped version padded to an encoder-decoder model. This model could then be trained for every border to predict the best pixel values for the padded pixels. With this method, they were able to perform slightly better than the partial convolution-based padding as described by Liu et al. [71].

2.3.5. Subsampling

Within the encoder phase, different strategies can be applied to reduce the dimensionality. One of them is subsampling. During a convolutional operation, the kernel moves over the input data. However, it does not necessarily need to move pixel by pixel since this would make the computation of a network extremely slow. One way to reduce the dimension is to use pooling, but subsampling can also act as an operation to reduce dimensionality. Effectively, subsampling means the kernel does not move pixel by pixel, but more pixels per step. The amount of pixels the kernel moves is defined as the stride. A stride of $(1, 1)$ would move the kernel one pixel at a time in both the horizontal a well as the vertical direction. Having a stride of $(3, 3)$ would move three pixels per step both horizontally as well as vertically, skipping two input pixels and reducing the dimension by a factor of nine.

One of the big disadvantages of subsampling is the increased possibility of aliasing to occur. This

can happen when not enough samples are taken and the bandwidth of the signal does not satisfy the Nyquist-Shannon theorem. Often this effect can be prevented by applying low-pass filters, but the placement and variables for these filters can greatly influence the performance of the CNN [127].

Finding the best stride value is difficult. It usually is a hyperparameter that requires cross-validation or architecture search to find the best combination [148]. However, the number of possibilities grows exponentially with the downsampling operations, making it infeasible to find the best configuration.

Zaniolo and Marques [139] experimented with a variable stride length during the first convolution. In the center, the stride was minimized, whereas the stride length around the edges of the input was greater. They were able to achieve better performance than with a constant stride value while having equal computational complexity. It was found that this is only the case for images in which the points of interest are centered, but due to the nature of CT scans and the position of the pancreas within the human body, this could also be beneficial for the segmentation of the pancreas on CT scans.

In trying to overcome the difficulty of finding the right stride, Riad et al. [104] designed DiffStride which is a downsampling layer with learnable strides. It makes use of cropping in the Fourier domain, where it learns the box size via backpropagation. Their results show a great improvement in accuracy as well as in robustness.

2.3.6. Regularization

Within convolutional networks, especially the very “deep” ones, overfitting is a common problem. It describes the phenomenon where a great number of parameters within a network are capable of finding a very accurate approximation of the underlying function of the training data, which is not a great approximation of the underlying function of the real data. In other words, the performance on the training data is very high, whereas the performance on the test or validation data is relatively low. This is particularly problematic when the variability in the training data is rather low. To increase the performance on the test and validation data, regularization methods are often used.

Santos and Papa [113] conducted a survey on many different regularization techniques. They identified three different types of regularization: based on data augmentation, internal structure changes, and label regularization. For the scope of this research, the regularization based on internal structure changes is the most interesting, which is why it is only this group that is focused on.

One of the often-used methods for regularization is Dropout [118]. This method deactivates nodes within the network with a certain probability, which are then ignored during the training. Because of this, units are less capable of co-adapting too much and thus reduce the chances of overfitting. Furthermore, it greatly reduces the number of computations that need to be done, since all computations associated with the dropout nodes do not need to be done anymore. This method showed great improvements in the performance on unseen data.

The dropout method has inspired many others to design a variation of it, e.g. MaxDropout [112], DropBlock, [31] and TargetDrop [145], which try to better define the areas that should drop out. However, the pattern of dropout with those methods is manually defined by different parameters and design choices. In AutoDropout [96] this pattern is learned by training, which can increase the performance and robustness of the model even further.

A different approach was taken in Shake-Shake regularization [29]. In this method, no nodes or branches are fully disabled. Instead, branches are multiplied by a random variable following a uniform distribution between 0 and 1. Whilst the branches are not fully deactivated, their influence is adjusted during every training step.

A variation of Shake-Shake is given by ShakeDrop [137]. The formulation is changed when compared to Shake-Shake, resulting in more tools not being trapped in local minima. According to the results in [113], ShakeDrop is one of the most promising regularizers in different situations.

2.3.7. Normalization

Normalization is a mathematical operation that is widely used in data preprocessing in particular [61, 38]. Although the precise definition of normalization can vary in different domains, the definition given in Huang et al. [44] describes the concept well. They defined it as *“a general transformation, which ensures that the transformed data has certain statistical properties”*. In their paper, five different levels of normalization are described: centering, scaling, decorrelating, standardizing, and whitening. The consequences and results of the different methods are visualized in Figure 2.7.

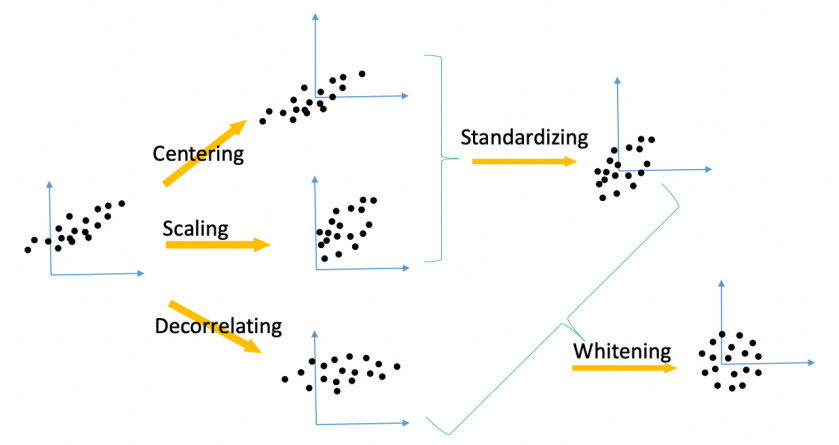


Figure 2.7: Visualization of different normalization operations [44].

The normalization transformations can be applied in different parts of a segmentation pipeline, improving a model's learning rate, quality, and accuracy [23]. One of the more obvious and often-used ones is directly in the input. Learning weights is more difficult if large input values, i.e. the intensity values for CT scan voxels, are multiplied by the, usually, small weights within the network. Because of the scale-invariant property, normalization can deal with this, irrelevant of how the input is scaled.

It can also help in decreasing distortions caused by different factors in the environment and generalizing the results. The normalized output and population statistics in the normalization process, in particular when it is a form of batched normalization, have stochasticity in them. The inputs are essentially drawn from an underlying distribution of the dataset or full input, which inherently introduces stochasticity [23].

Apart from applying normalization on just the input, it can, furthermore, be applied within the training layers for every layer. One of the more notable methods for this is Batch Normalization (BN) [47]. Since the whitening based on a complete layer's input is too costly, BN normalizes each scalar factor independently, making it have zero mean and unit variance. Furthermore, it is applied to mini-batches instead of the full layer, thus approximating the real mean and variance. BN normalizes on all samples in the batch for each feature channel individually.

Another approach is Layer Normalization (LN) [6], which essentially transposes BN. Within LN the mean and variance are calculated along each individual sample using all summed inputs within each layer. This is particularly useful for sequential data, for example in recurrent neural networks or Natural Language Processing (NLP) tasks, since the normalization statistics can be computed at each time step individually.

In case the batch size of BN would be one, Instance Normalization (IN) [125] occurs. This approach computes the normalization statistics for each data point individually for every single feature channel. In particular, IN is very suitable for style transfer applications.

Lastly, Wu and He [132] introduced Group Normalization (GN). This is essentially a batched form of LN, where the normalization statistics are computed over a group of feature channels for every individual sample separately. Every feature channel belongs to a group, with which it shares the statistics. In case the number of groups would be one, it is exactly the same as LN. The authors found that GN works particularly well on semantic segmentation tasks, greatly improving the runtime especially.

All four different normalization areas can be seen in Figure 2.8. One can imagine that combinations of the different methods can be made, e.g. Batch Group Normalization (BGN) [119].

Instead of performing the normalization on the output of layers, it can also be computed on the weights of the nodes, resulting in Weight Normalization (WN) [110]. This, however, seems to be used less in semantic segmentation tasks.

2.3.8. Attention Mechanism

As ANNs are derived from the human brain, it makes sense that there is research in implementing more human brain mechanisms into the ANNs. Mechanisms that play a crucial role in human perception are attention mechanisms [103]. The mechanism of selective attention accounts for the fact that the

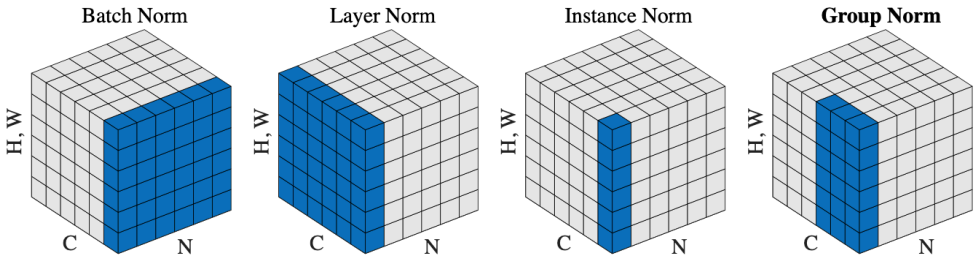


Figure 2.8: Different normalization methods and the data they use to compute the normalization statistics, where C represents the channel dimensions, N the sample dimension and H, W the height and width dimensions respectively [132].

capacity of the human brain to process all stimuli is limited. Instead, the brain decides to focus on only a particular part of stimuli, which gets priority in processing, i.e. the stimuli that are in one's attentional field [92].

Similarly, attention mechanisms are applied in ANNs. It essentially reweights specific values in the convolutional layers in order to put the ANN's attention on specific features of the input. This can result in acceleration of the learning process, more extraction of critical and discriminative features, improvement of the robustness, and the capability of being more adaptable to a smaller training dataset [124].

Trainable attention mechanisms can generally be divided into two groups, soft and hard attention. Hard attention essentially describes the mechanism where the model iteratively finds regions that it will process in high resolution, as it ignores (to some extent) the values outside of the region. It essentially tries to find the best cropping of an image [85]. This process, however, is non-differentiable and is therefore often combined with reinforcement learning, making this type of attention more difficult to train [93].

Soft attention, on the other hand, is probabilistic and can use the standard backpropagation to learn without the need for Monte Carlo sampling [93]. Within a layer, it receives the feature map as input and outputs highlights on the more important features. This can be done for specific channels, as well as for spatial position, resulting in either channel or spatial attention respectively. Wang et al. [128] defined a Residual Attention Network, combining both forms of attention in a mixed attention approach. They stacked Attention Modules that can generate attention-aware features, which greatly improved performance.

Jetley et al. [51] added spatial attention from feature vectors in later layers to the end classifier. Instead of feeding the input of the final classification layer directly to the final layer, they combined this input with the feature vectors from earlier layers. These combined feature maps were then fed into the final layer, making sure both global as well as local feature maps were used in the final classification.

Oktay et al. [93] applied attention gates to all the skip connections in the U-Net. In their research on the segmentation of the pancreas, a small organ with high variability, they found additive attention can improve performance. The Attention Gate (AG) was fed the gating signal from a coarser layer and the input features. Linear transformations are computed using 1x1x1 convolutions and activated by a sigmoid activation function. This results in attention coefficients that are then multiplied with the input features to form the true attention-based input features for the next layer. With their Attention U-Net, they could consistently outperform a U-Net without attention.

Channel attention was addressed more in the Squeeze-and-Excitation networks (SENet) [42]. A SENet consists of SE blocks, that compute channel weights and dependencies. In the squeeze phase, global average-pooling is used to find channel-specific statistics, which is aimed to overcome the lack of contextual information outside of the local receptive field, particularly in the lower layers. This channel-wise information is then used in the excitation phase, where a ReLU activation function is used together with a sigmoid activation function to compute the final channel weights. The authors made use of two fully-connected layers around the ReLU operation to decrease the dimensionality and make the block more efficient. They were capable of winning the ILSVRC 2017 challenge [109] by significantly beating the competition and previous state-of-the-art.

In an attempt to introduce both channel as well as spatial attention in a CNN, Woo et al. [131] designed the Convolutional Block Attention Module (CBAM). CBAM consists of a channel attention

module and a spatial attention module in sequence. The channel attention module is inspired by the SE block of [42] but extended by making use of both max-pooling as well as average-pooling. They showed that both pooling methods add additional information and using both yields better results than using only one. The spatial module operates along the channel axis by also applying both average and max-pooling and concatenating the result. On this result, a convolution is applied with a 7×7 filter size. The final weights are then computed by a sigmoid activation function over the convolution result. One disadvantage of this method is the fact that it adopts a convolution to produce the spatial attention map, which in turn leads to a limited receptive field [37].

The CBAM module and AG were combined in the triple attention mechanism in Tong et al. [124]. They designed the ASCU-Net, where every decoder layer was preceded by an AG module as earlier described and succeeded by a variant of the CBAM module. The CBAM module as used in the ASCU-Net, however, does not do the channel and spatial attention blocks in sequence, but in parallel, and only the average-pooling is used. ASCU-Net did perform better than Oktay et al. [93] and their results show that adding AG modules to the network is beneficial when comparing it with only the channel and spatial attention modules. However, since the channel and spatial modules used in ASCU-Net are slightly different than the ones used in Woo et al. [131] it is difficult to compare them directly in terms of performance.

In the previously mentioned mechanisms, the contextual attention information per pixel is limited to a smaller region around the pixel, based on the filter size. Recurrent Criss-Cross Attention (RCAA) [45] aims to overcome this. Feature maps are processed with multiple sequential Criss-Cross Attention modules in order to retrieve attention from all pixels in the spatial domain. In their experiments, RCAA is capable of outperforming all state-of-the-art methods for the datasets Cityscapes [21], ADE20K [142], and COCO [69]. Although the content of the images of these datasets is quite different when compared to the ones that, for example, ASCU-Net is tested on, the RCAA module itself is promising in other contexts as well.

Another attempt to improve the receptive field of the attention mechanism, while combining both the channel and spatial attention, was made in Qu, Xia, and Zhang [100]. In their SP_CSANet, a U-Net-like structured network as can be seen in Figure 2.9, strip pooling techniques were used to achieve a greater receptive field. In every encoder layer, a strip pooling attention module (SPAM) was added. In this SPAM block, the result of two strip poolings with kernel size CxHx1 and Cx1xW are concatenated and result in the input for the next layer after some 1x1 convolutions and a sigmoid activation function. Every skip connection is taken through a strip convolution model, which concatenates the results of two different convolutions, sequentially applying a filter of 7×1 and 1×7 or 1×7 and 7×1 respectively. This result is then fed into a channel spatial attention module (CSAM) together with the result of the previous decoder layer. The CSAM concatenates both the deep information from the previous layer and the shallow information from the skip connection to apply a channel and spatial attention module in a similar fashion as in CBAM. This results in a large receptive field while also allowing for attention between different feature maps from different layers of the network. With this network, they were able to consistently beat the state-of-the-art setting a new state of the art.

Lastly, Misra et al. [83] aimed to design a lightweight but efficient attention mechanism, which resulted in the Triplet Attention Module. Essentially, this consists of three parallel pipelines that are concatenated again. One builds spatial attention similarly to CBAM. The other two capture cross-dimension interaction between the channel and the spatial dimensions H and W respectively. This is done by applying a rotation operation of 90 degrees on either the H- or W-axis. This rotated data is then passed through their Z-pool, which concatenates max- and average-pooling operation results. After this, the three branches are simply averaged, which will bring the result. One big advantage of this mechanism is the limited additional parameters of only $6k^2$ where k is the height and width of the kernel. For comparison, CBAM adds $2C^2/r + 2k^2$ parameters. Triplet Attention, furthermore, comes close to the performance of CBAM and can sometimes even outperform CBAM depending on the task and chosen underlying networks.

2.4. Segmentation Models

The field of deep learning methods has been researched extensively, especially in the last decade. In this section, a deeper look will be given at different deep learning methods for semantic segmentation. Deep learning is a subfield of artificial intelligence and in particular a subfield of machine learning. It

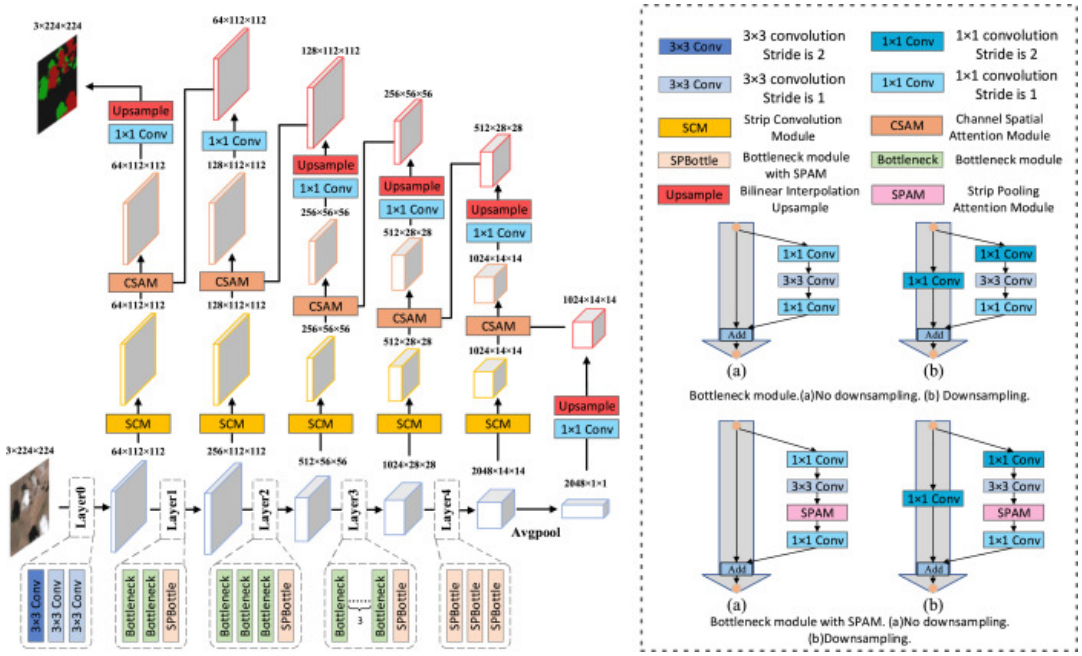


Figure 2.9: The SP_CSANet as designed in [100]

has been majorly contributing to the solving of many problems that have been present in the artificial intelligence domain for years [62]. The idea of deep learning is very similar to the way a human brain works. Many different neurons are connected to each other, forming an artificial neural network (ANN) [8]. It is these large ANNs that are very capable of finding structures in a large amount of data and using that to make accurate data-driven decisions. In particular, this is the case for contexts that are complex but have large datasets readily available [55]. In image processing, deep learning has proven to bring efficient and accurate models [32], making it a great help in the medical domain as well, by analyzing and processing, for example, CT scans.

There are many different architectures and approaches, but one of the most popular approaches in image segmentation is an encoder-decoder model [80]. Such an architecture consists of two parts: the encoder, which tries to encode the original input into a lower dimensional image, and the decoder, which tries to reconstruct the image again by adding dimensions layer by layer. The idea is that features are extracted during the encoder stage such that higher semantic information is captured, whereas the decoder gradually recovers that information again [15]. Usually, an encoder-decoder makes use of convolutional operations and many other operations as described in Section 2.3. There are different techniques associated with encoder-decoder architectures and different ways to implement such an architecture.

U-Net One of the most popular encoder-decoder models for image segmentation is the U-Net, introduced by Ronneberger, Fischer, and Brox [106]. This network was designed to be able to be trained faster and with smaller data sets, while still being able to see many features. The architecture is u-shaped as the encoder and decoder stages consist of an equal amount of operations. In the encoder stage, a series of layers consisting of two 3×3 convolutions followed by a ReLU are applied, alternated with 2×2 max-pooling operations with a stride of 2 [106]. In the decoder stage, the reverse counterparts of the operations in the encoder stage are applied. In addition to this, feature maps are copied from the encoder stage to their corresponding decoder stage counterparts to mitigate the loss of pattern information [80]. These are the so-called skip connections. The full architecture can be seen in Figure 2.10.

V-Net The V-Net architecture is yet a different approach of the U-Net and was mainly introduced to account for 3D volumes in medical data [79]. The V-Net, therefore, also has 3D kernels of $5 \times 5 \times 5$ voxels. The V-Net architecture does not make use of max-pooling operations as is the case with U-Net.

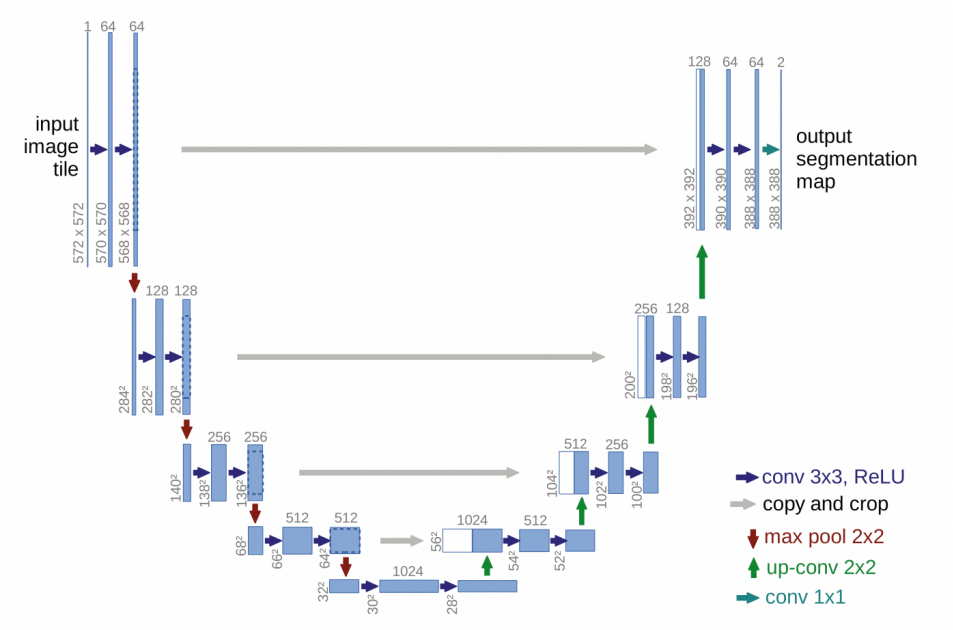


Figure 2.10: U-Net architecture [106]

Discouraged by Springenberg et al. [117], they instead used new convolutional layers with voxels of $2 \times 2 \times 2$. Throughout the network, they also made use of Parametric Rectified Linear Units (PReLU). The model introduced a slightly adjusted objective function based on DSC to account for the 3D space, which helped in the strong imbalance between foreground and background pixels.

U-Net++ Many different variations have been created, based on the U-Net architecture. One of them is U-Net++ [144]. Their hypothesis is that it can more effectively capture fine-grained details of foreground objects. They introduce a new architecture where the encoder and decoder are not just connected with skip connections. In between, there is a series of nested dense convolutional blocks, with the idea to bridge the semantic gap between the feature maps of the encoder and decoder before they are fused. Furthermore, they introduced deep supervision [63] in the architecture. The loss function is defined as a combination of the DSC and binary cross-entropy, enabling model pruning and improving the performance. U-Net++ was capable of increasing the IoU score by an average of 3.9 percentage points on different medical image sets when compared to U-Net.

R2U-Net Another approach was taken in [3], combining the U-Net architecture with concepts of residual recurrent networks in a Recurrent Residual U-Net (R2U-Net). The feature accumulation of the residual recurrent blocks in the network allows for a better feature representation for segmentation tasks when compared to a classical U-Net. In the layers of the R2U-Net, the convolutional operations have a recurrent connection to themselves. The output of the two convolutions is then summed with the residual connection from the input of the layer. This allows the network to have the same number of parameters as U-Net while increasing its performance.

TransUNet One of the difficulties of convolutional neural networks (CNN) is learning larger-scale spatial dependencies since a convolution is a local operation. TransUNet [13] tries to overcome this by combining the strengths of a transformer and U-Net. Transformers are often used in sequence-to-sequence applications, e.g. NLP. It is hypothesized that using a transformer as an encoder can bring benefits since it can extract global contexts. However, a naive approach where a transformer is simply used as an encoder and a CNN as a decoder was not found to outperform the state-of-the-art. On the other hand, using a CNN-Transformer hybrid as an encoder did significantly improve the performance. As can be seen in Figure 2.11, the CNN is used to project image patches as linear sequences as input for the transformer. In this linear representation, the spatial position is embedded to retain positional information. Skip connections are used from the CNN encoder to the decoder in order to leverage the

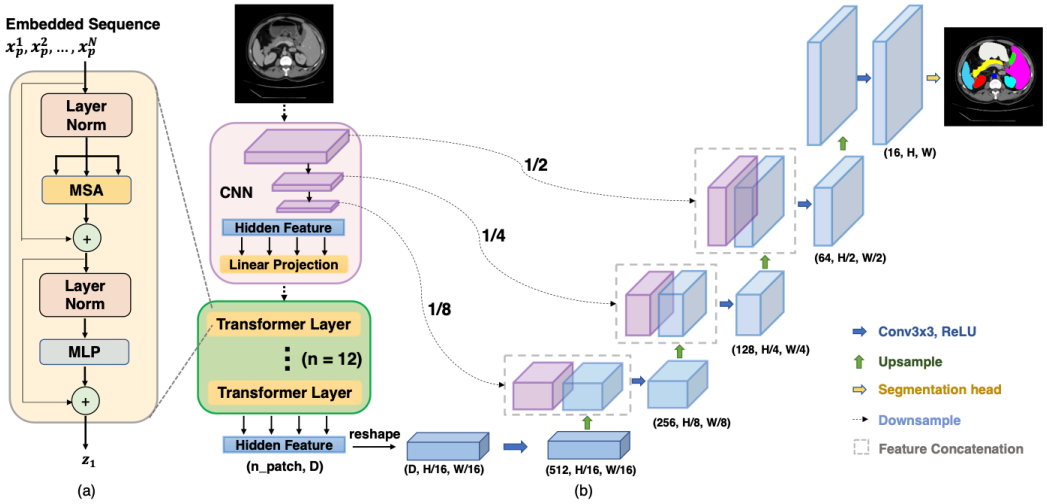
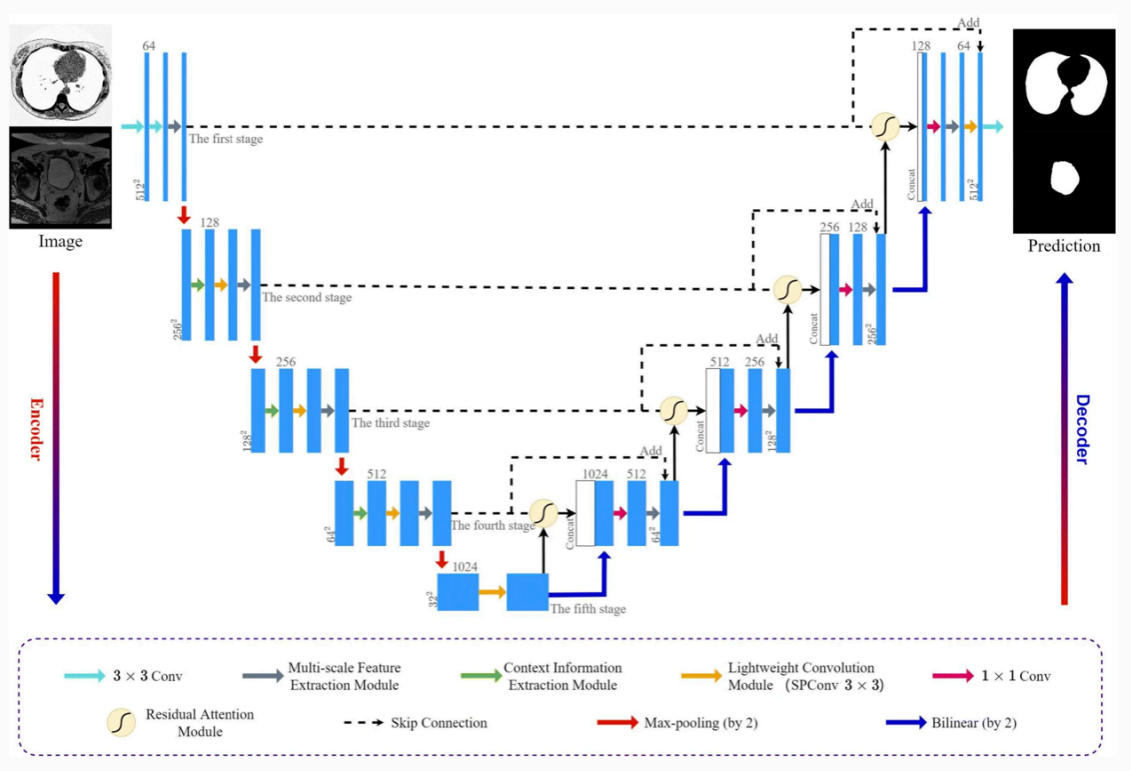


Figure 2.11: Overview of the TransUNet architecture. (a) Scheme of the Transformer layer; (b) architecture of the proposed TransUNet. [13]

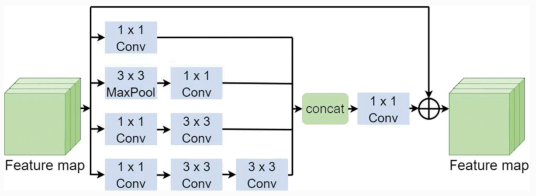
high-resolution feature maps from the CNN in the decoder. In the multi-organ segmentation task, the TransUnet outperformed U-Net significantly, in particular on the pancreas the new architecture showed a great performance improvement.

DQN U-Net In Man et al. [77] the problem of not making use of a larger context was also tackled. The Deep Q Learning (DQN) driven and deformable U-Net approach consists of two steps. First, a DQN-driven localization agent finds the region of interest (RoI) containing the pancreas. this stage is driven by a context-adaptive deep reinforcement learning approach. The agent has ten possible actions, namely five zoom actions, four shift actions, and one trigger action, i.e. to stop the search if the bounding box is good enough. The reward function of the agent is a slightly modified IoU, i.e. the area of the intersection of the bounding box and true mask divided by the union of both. The second stage is the actual segmentation within the previously found bounding box. The model in Man et al. [77] uses a deformable U-Net architecture for this. This is essentially a "standard" U-Net, however, it makes use of deformable convolutions. This changes the receptive field by adding an offset weight for every position in the kernel. In this way, the model itself can learn the best shape of the kernel. These two steps are done for three orientations of the CT scan, namely the axial, coronal, and sagittal slices. The results of every individual pipeline are then combined by a majority vote to determine the final segmentation. This architecture turned out to be very promising, achieving a state-of-the-art DSC of 86.93% on the NIH dataset.

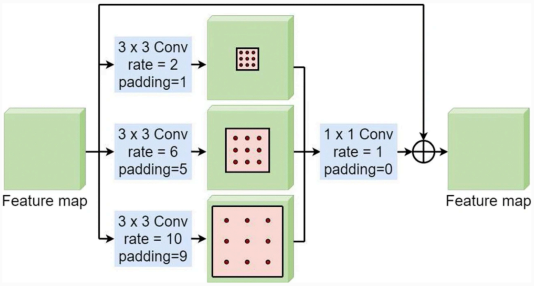
MC-Net The need to preserve details within the encoder stage was also one of the main motivations in Xia et al. [134] when designing the Multi-Scale Context-Attention Network (MC-Net, Figure 2.12a). The model follows a similar architecture as the regular U-Net, however, instead of two convolutions per layer, it applies a Context Information Extraction Module (CIE), a Split-based Convolution Module (SPConv [140]) and Multi-scale Feature Extraction Module (MFE) in the encoder layers and a 1×1 convolution followed by an MFE in the decoder layers. The MFE module finds correlations between multiple scales by four parallel convolution branches that result in a different amount of channels and represent different features (Figure 2.12b). The CIE module is inspired by atrous spatial pyramid pooling [14] and multi-scale dilated convolution [36, 135]. It has three dilated branches as can be seen in Figure 2.12c to extract features of different detail levels to ensure both high-level details as well as very fine details are in the feature maps. Furthermore, residual attention modules are added to the skip connections to bring the more important features to the decoder stage. The full MC-Net is capable of obtaining clearer segmented boundaries by exploiting correlated information across multiple scales and context information. As a result, it could outperform the state of the art in four different medical datasets.



(a) The MC-Net architecture.



(b) Illustration of the MFE module.



(c) Illustration of the CIE module.

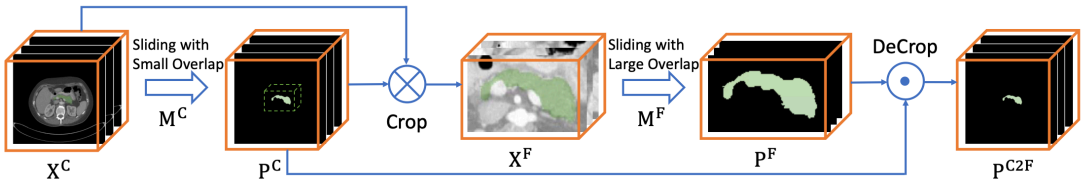
Figure 2.12: Architecture and modules of MC-Net. [134]

2.5. State-Of-The-Art Pancreas Segmentation Pipelines

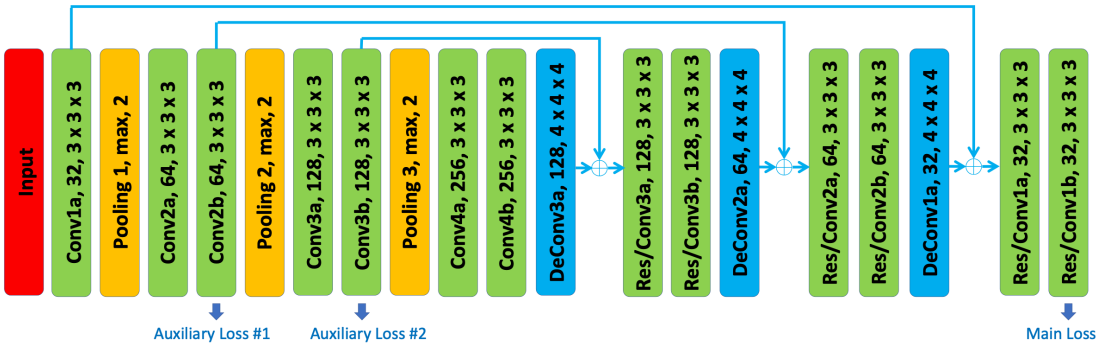
There are several studies into the semantic segmentation of the pancreas. The most often-used dataset on which the models are trained and evaluated is the already mentioned NIH dataset [107]. In Table 2.1, the performance of all researched state-of-the-art algorithms is compared. All papers reported the DSC values of the respectively presented techniques, which is why that is the main evaluation metric to compare the algorithms. For the ones that did mention the precision, sensitivity, and Hausdorff distance, Table 2.1 also includes these values.

ResDSN C2F In 2018, Zhu et al. [147] presented the 3D Course-to-Fine Framework. This framework is designed to first find a course segmentation result on a sampled representation of the input data to retrieve a bounding box. In this bounding box, a more precise 3D segmentation is performed to find the fine segmentation result. The flow of the model can be seen in Figure 2.13a. The volumetric segmentation is done by the 3D ResDSN, which can be seen in Figure 2.13b. The architecture is rather close to the original U-Net in Ronneberger, Fischer, and Brox [106], having multiple encoder layers and an equal amount of decoder layers. In each layer, two convolutions are applied with a $3 \times 3 \times 3$ kernel, followed by BN and ReLU. One of the differences is the fact that the residual or skip connections are not concatenated but summed elementwise. This was done to elaborate the fine-scaled segmentation.

Furthermore, the authors included deep supervision in the network by adding two auxiliary loss layers in the encoder. The output of the auxiliary loss layers is upsampled by a deconvolution to match the size of the input. The three loss functions are then combined with a weighted average to compute the final loss. This deep supervision *"provides robustness to hyper-parameters choice, in that the low-level layers are guided by the direct segmentation loss, leading to faster convergence rate"* [147]. The loss that was used for all the functions is the cross entropy or negative log-likelihood loss. The mean DSC on the NIH dataset of this method was $84.59\% \pm 4.86\%$. Interestingly, the authors also trained and tested their method on the JHMI dataset [143], which also includes some CT scans of a pancreas with pathological cysts. This can cause huge variations in the different data points, making the dataset much more challenging. ResDSN C2F was still able to achieve a mean DSC of $80.56\% \pm 13.36\%$



(a) The flow of the 3D C2F model



(b) The CNN for volumetric segmentation

Figure 2.13: Graphical representation of the 3D Coarse-to-Fine Framework [147]

DQN U-Net One of the other SOTA algorithms has already been mentioned in Section 2.4: the DQN-driven and deformable U-Net of Man et al. [77]. One of the most important features of this approach is the two-step-based strategy for approximating the location of the pancreas, before actually segmenting it, just like in Zhu et al. [147]. The DQN agent first defines a boundary box in which the pancreas should be present. Then the U-Net segments it. The U-Net itself has a classic architecture but does bring deformable convolutions with it. A deformable convolution essentially redefines the receptive field, i.e. it adds an offset to the kernel locations [77]. This is all done on the three different axes, in order to create a 2.5D model, as can be seen in Figure 2.14. The mean DSC they achieved on the NIH dataset [107] was $86.93\% \pm 4.92\%$. The loss function they used to train their model is the Dice loss function as already defined in Section 2.3, using the non-squared variant.

Two-stage 3D CNN A slightly different approach was taken in Zhao et al. [141], with a Two-stage 3D CNN. It also uses two steps to first identify a Region of Interest (RoI) after which a more refined segmentation will be done. In the first, coarse, stage, the CT scan is downsampled by a factor of four. Then the downsampled data is fed into a 3D U-Net that is trained to segment the pancreas. In the second stage, there is a difference in the process for the training and testing phase. In the training phase, the ground truth is used to define a bounding box with a margin of ten pixels around the true mask along the three axes. This bounding box is then used to train another 3D U-Net to segment the pancreas within the bounding box. Note that this can be done in parallel since the second stage does not need the output of the first stage.

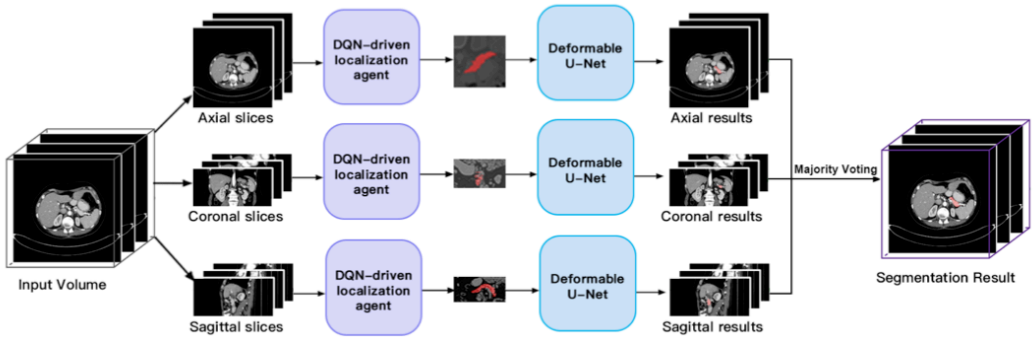


Figure 2.14: Architecture of the DQN with deformable U-Net approach [77]

The process during the testing phase can be seen in Figure 2.15. The first stage is equal to the one described for the training phase. However, in the second stage, the output of the first stage is used to define the bounding box. The second stage has two different flows, that can be described as follows:

1. The (downsampled) result of the first stage is directly used to formulate a bounding box with a margin of two pixels from the predicted mask along different axes. This bounding box, which is still downsampled, is then fed into the 3D U-Net to find a segmentation result in the downsampled space. The segmentation result in the original space is then obtained by rescaling this bounding box with the downsampling factor along different axes.
2. The (downsampled) result of the first stage is first upsampled to the original resolution. From that, a bounding box is defined with a margin of ten pixels around the predicted mask. This bounding box is then used as input for the 3D U-Net to get a fine segmentation result.

Note that the difference between the two parallel paths originates from the moment on which the results are upsampled and they thus differ in the errors made during upsampling. The result is then computed by using a majority voting of the output of the two paths of the second stage and the directly upsampled prediction mask of the first phase.

The loss function that was used in Zhao et al. [141] is a combination of the dice loss and the center loss, defined as follows: $\mathcal{L} = \mathcal{L}_{Dice} + \gamma\mathcal{L}_{Center}$, where γ is a penalty parameter. They defined γ to be 10^{-3} for the first 50 epochs, after which it is decreased to 0 in the following epochs. In their experiments, they were able to achieve a mean DSC of $85.99\% \pm 4.51\%$.

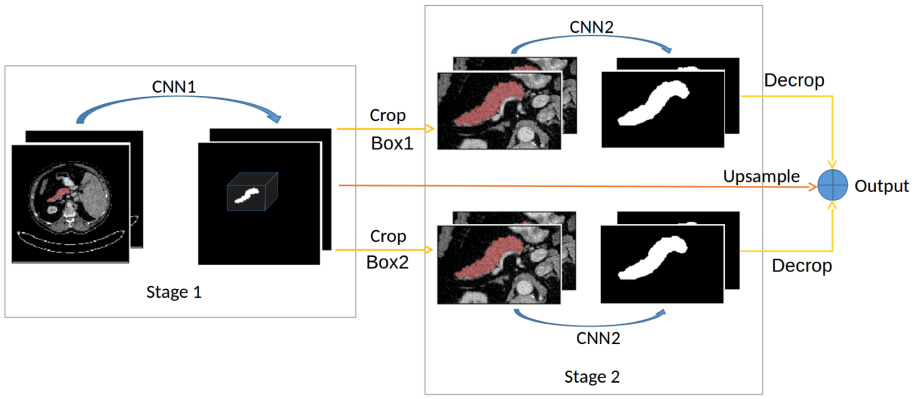
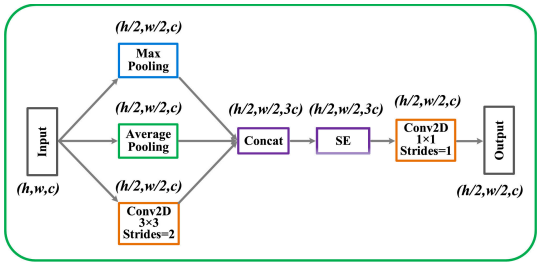
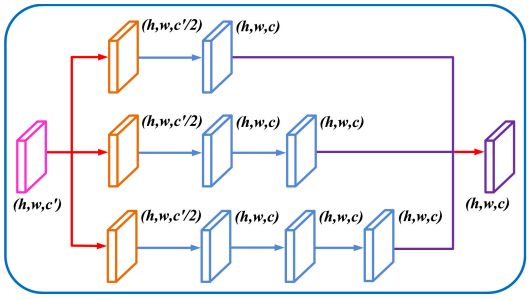


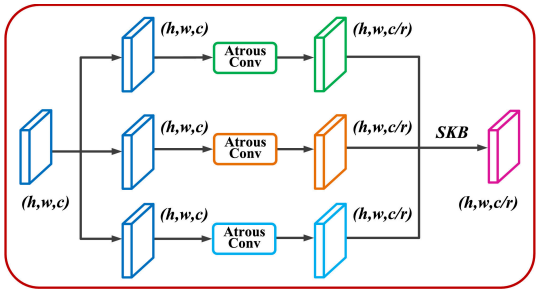
Figure 2.15: Two-stage 3D CNN architecture in the testing phase [141]



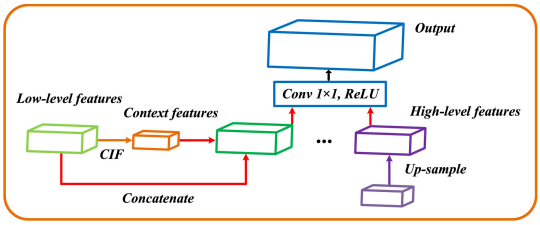
(a) The hybrid down-sampling block (HDSB)



(b) The multiscale convolutional block (MCB). The red lines represent 1x1 convolution, the blue lines represent 3x3 convolution, and the purple lines represent concatenate operation.



(c) The context module (CIF)



(d) The Skip Connection+

Figure 2.16: The different blocks as described in Ma, Zou, and Liu [72]

MHSU-Net The new architecture presented by Ma, Zou, and Liu [72] remains relatively close to the original U-Net architecture. Instead, the authors decided to optimize the blocks within the architecture. They describe three different new blocks that can be incorporated into the U-Net architecture: hybrid down-sampling block (HDSB), multiscale convolutional block (MCB), and Skip Connection+.

The HDSB is proposed to replace the max pooling operation in the encoder. The structure of the block can be seen in Figure 2.16a. The block is designed to retain richer features during downsampling by the combination of three downsampling methods: max pooling for texture features, average pooling for background information, and 3×3 convolutions with stride 2 as a learnable pooling operation. The results of the different pooling methods are then concatenated and used as input for an SE block as described in Section 2.3.8 to incorporate channel attention.

The MCB replaces the convolutions in the encoder of the U-Net and is designed to extract abundant multiscale semantic features. As can be seen in Figure 2.16b, the block consists of three different branches that apply one, two, and three sequential 3×3 convolutions respectively. The result of the three branches is then concatenated and convoluted with a 1×1 convolution to compute the final result. Because of the different amount of convolutions, the block is capable of extracting features from different scales and thus producing a richer feature map into the next layer. The MCB is also one of the most important building blocks in the algorithm designed by Li et al. [64].

Skip Connection+, as the name suggests, replaces the regular skip connection of the U-Net. One of the main building blocks of the Skip Connection+ is a context module (CIF) as shown in Figure 2.16c. The CIF module aims to solve the problem of the limited receptive fields of convolutions, by making use of three different atrous convolutions with varying atrous rates. This CIF module is applied to the low-level features of the skip connection and the output of it is then concatenated again with the low-level and high-level features. The three types of features are then fused along the channel dimension with a 1×1 convolution.

The MHSU-Net was evaluated on different datasets including the NIH pancreas dataset [107]. In their experiments on pancreas segmentation, they used the Dice loss function as an optimization parameter. They were capable of achieving the best DSC at this time, to the best of the knowledge of the author, with a mean of 88.48%. Unfortunately, the standard deviation was not mentioned. Interestingly enough, in the ablation experiments, they found that an MU-Net, i.e. a U-Net with only the MCB applied to it, could already achieve state-of-the-art performance with a mean DSC of 88.08%.

PanKNet The creators of PanKNet [99] did an attempt to overcome the limited receptive field of convolutions as well. They chose a hierarchical approach. PanKNet is designed as an encoder-decoder architecture as well. However, instead of a single decoder, it has multiple decoders. As can be seen in Figure 2.17, the output of every layer of the encoder is used as input for its own decoder. The intermediate segmentation masks are then concatenated along the channel dimension and merged through a convolution to output the final predicted mask. Note how the different decoders are different than having ordinary skip connections. With the multiple decoders, different predicted masks can be computed having different local and global dependencies. The skip connections simply aim to ease the gradient flow and forward the low-level features to the decoder.

The decoders process the output of one of the encoder layers and upsample it to have an equal resolution as the input, by cascading a number of upsampling blocks depending on the size of the input feature maps. Each upsampling block is made up of a 3D convolutional block with BN and ReLU, one or two separable convolutional blocks, and a trilinear upsample layer. The last layer is a pointwise 3D convolution in order to output a single mask prediction.

For the encoder, two different approaches were taken. The first approach is to use an architecture based on S3D [136], which is pretrained and is named PanKNet. The second approach is an encoder based on MobileNetV2 [111], which is significantly more lightweight with a factor of ten fewer parameters than the S3D encoder. Due to the limited number of parameters, the architecture with the MobileNetV2 encoder is called PanKNet$_{Light}$. Both architectures are trained on a hierarchical variant of the Dice loss function, achieving a mean DSC of 88.01% and 87.13% for PanKNet and PanKNet$_{Light}$ respectively.

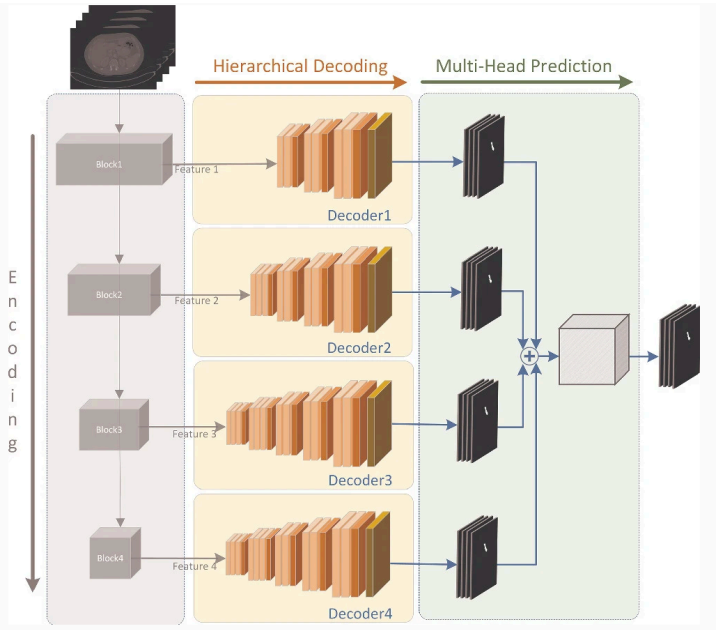


Figure 2.17: The PanKNet architecture [99]

Dual-input V-mesh Interesting contributions were made in the Dual-input V-mesh of Wang et al. [129] as well. Most notably, as the name suggests, the network takes two images as input as can also be seen in Figure 2.18. One is the original CT scan image, the other is an image processed by a contrast-specific graph-based visual saliency (GBVS) algorithm. It essentially is a saliency map, to extract more boundary information. The GBVS applies Gaussian pyramids [50], followed by a bank of Gabor filters [40] to obtain direction features. Then weights are computed between any two nodes and the initial saliency map is then obtained by the stable state of a Markov chain. To enhance the tissue contrast, the values of the initial saliency map are multiplied by the intensity values of the original CT images.

These two inputs are then fed into a V-mesh FCN, which shows similar horizontal dense connections and vertical cascaded connections as U-Net++ as discussed in Section 2.4. This would extract more detailed feature information. In the skip connections, an Attention Gate (AG) is applied to incorporate attention in the network. In the last layer, a spatial transformation and fusion module (SF) is applied. The

SF module contains a deformable convolutional layer, followed by a multi-branch residual convolutional block, similar to the MCB. The deformable convolution is, furthermore, applied to the convolutions in the encoder stage in combination with a residual connection.

During the training of the Dual-input V-mesh network, the binary cross-entropy loss function is used to update the network weights. With their model, they were able to achieve a mean DSC of $87.4\% \pm 6.8\%$

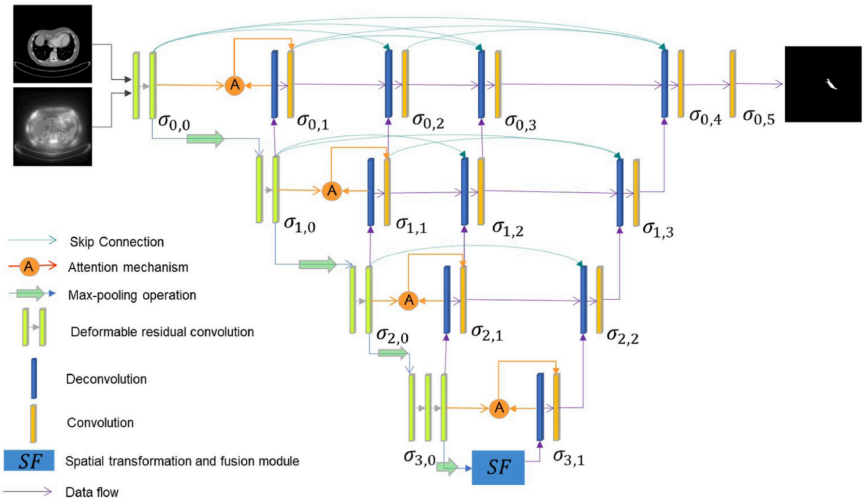


Figure 2.18: Dual-input V-mesh architecture [129]

CTUNet Similarly to TransUNet as earlier discussed in Section 2.4, Chen and Wan [16] researched the possibility of combining a transformer with a U-Net architecture. Their proposed CTUNet can be seen in Figure 2.19. The most noticeable difference here is the fact that the transformer uses the output of all layers in a multi-head cross-attention module and thus fully replaces the skip connections of a typical U-Net architecture. The advantage of this is that global features can more easily be used from different layer levels.

In addition to this, Pancreas attention modules (PA) are added to incorporate attention into the model. The PA module is based on the Project & Excite module (PE) as proposed in Rickmann et al. [105] but preceded by another attention module. Since there is only a small portion of foreground pixels. Essentially, the PA module applies an AG to bring attention to the pancreatic region. Then the PE module is applied in order to find the global pooling in three different dimensions and add additional attention to the images.

In training the model, the authors recognized the data imbalance and tried to (partially) overcome this by using a combination of dice loss and focal loss. With their model, they were able to achieve a mean DSC of $86.8\% \pm 4.1\%$ on the NIH dataset [107] with 4-fold cross-validation.

Method	DSC $\pm$ std	Precision	Sensitivity	HD (mm)
ResDSN C2F [147]	$84.59\% \pm 4.86$	–	–	–
DQN U-Net [77]	$86.93\% \pm 4.92$	–	–	–
Two-stage 3D CNN [141]	$85.99\% \pm 4.51$	–	–	–
MU-Net [72]	$88.08\% \pm ???$	–	–	–
MHSU-Net [72]	**88.48% $\pm$???**	–	–	–
PanKNet$_{Light}$ [99]	$87.13\% \pm 4.58$	$86.85\% \pm 6.52$	$88.48\% \pm 5.12$	–
PanKNet [99]	$88.01\% \pm 4.74$	$88.25\% \pm 5.45$	**88.69% $\pm$ 5.99**	–
Dual-input V-mesh [129]	$87.4\% \pm 6.8$	**89.50% $\pm$ 5.80**	$87.70\% \pm 7.90$	**18.41 $\pm$ 28.10**
CTUNet [16]	$86.8\% \pm 4.1$	$86.2\% \pm 6.5$	$88.0\% \pm 6.0$	–

Table 2.1: Comparison of the performance of the current state-of-the-art on the NIH dataset

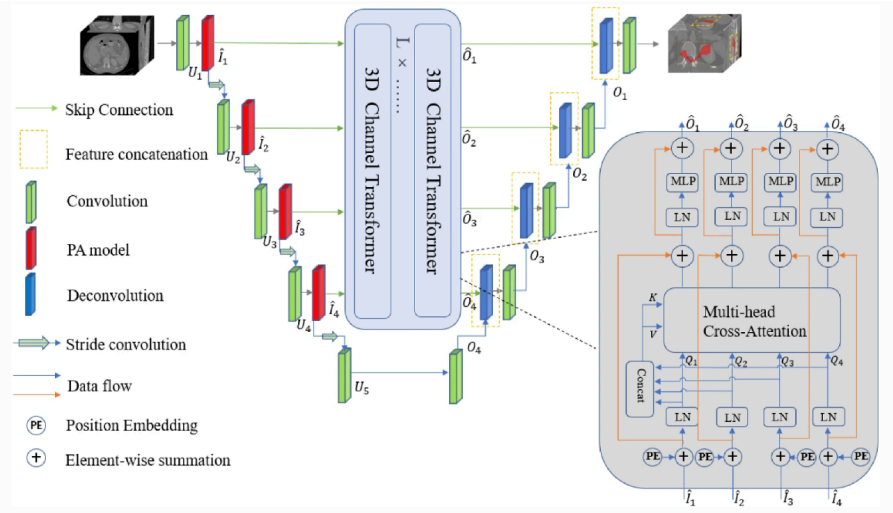


Figure 2.19: The CTUNet model. Q, K, and V denote query, key, and value, respectively, participating in multi-head self-attention calculation [16]

3 Methodology

Since research into the segmentation of the pancreas is still at a relative infant stage, the number of different possibilities to extend upon current research is high. This thesis is an exploratory study into the segmentation of the postoperative pancreas. Even though the segmentation problem is extremely similar to the one with CT scans from healthy patients, the postoperative state characterizes as one where the pancreas is even more variable in both size as well as shape, not to mention the completely different anatomical situation in the abdomen. To be able to successfully segment the postoperative pancreas, a model would need to be improved in such a way that it is capable of finding difficult-to-find patterns in an environment that is differently shaped for every sample in the dataset. Something that can currently only be done by experienced radiologists, who can still have difficulties finding the correct tissue. This chapter will go more into depth on the specifics of the data, but also on the researched architecture and the way in which it is trained, validated, and evaluated. Lastly, some more details on the implementation are described.

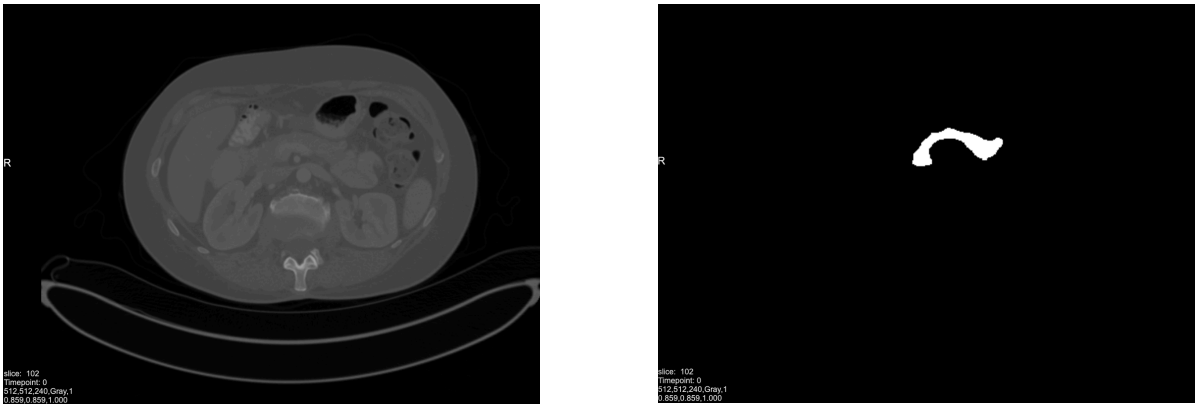
3.1. Data Collection and Preprocessing

In this study, two different datasets were used in order to train and validate a model. Since most research on pancreas segmentation makes use of one specific dataset, namely the one from the National Institutes of Health (NIH) [107], this dataset was used in order to make this research comparable to prior research and to have additional data for potential pretraining steps. Apart from the NIH dataset, a private dataset from the RACU was used with postoperative scans, which was used to finetune the models for postoperative scans. Both datasets are not optimal to feed to a model immediately, e.g. the voxel dimensions are different among samples and the orientation can differ. Preprocessing steps can overcome this, which is why various steps were applied. This section will describe both datasets as well as the preprocessing steps that were taken before the data was fed into the model.

3.1.1. NIH Dataset

The publicly available dataset of pancreatic CT scans from The Cancer Imaging Archive (TCIA) created by the NIH [107] is the first one that was used. This dataset is widely used in pancreas segmentation research [12, 28, 58, 67, 77, 108] and consists of 82 contrast-enhanced CT scans of people that do not knowingly have any pancreatic diseases or that have undergone pancreatic resection. This means that the pancreas should be largely intact. There were 53 male and 27 female subjects, with ages ranging from 18 to 76, with a mean age of 46.8 and a standard deviation of 16.7 years. 17 of the subjects were healthy kidney donors and the remaining 65 were selected by a radiologist for lacking major abdominal pathologies or pancreatic cancer lesions. The scans were made approximately 70 seconds after intravenous contrast injection in portal-venous. The CT scans have a resolution of 512x512 pixels with different numbers of slices, varying pixel sizes, and slice thicknesses between 1.5mm and 2.5mm, acquired on Philips and Siemens MDCT scanners with a tube voltage of 120 kVp. The pancreas was manually segmented slice-by-slice by a medical student and verified/modified by an experienced radiologist to serve as ground-truth data. The original images were acquired in DICOM format, after which they were converted to NIFTI format for easier processing. The scans can be downloaded from

the cancer imaging archive website. An example of one of the slices of one CT scan can be found in Figure 3.1 [107].



(a) CT scan **(b)** Mask

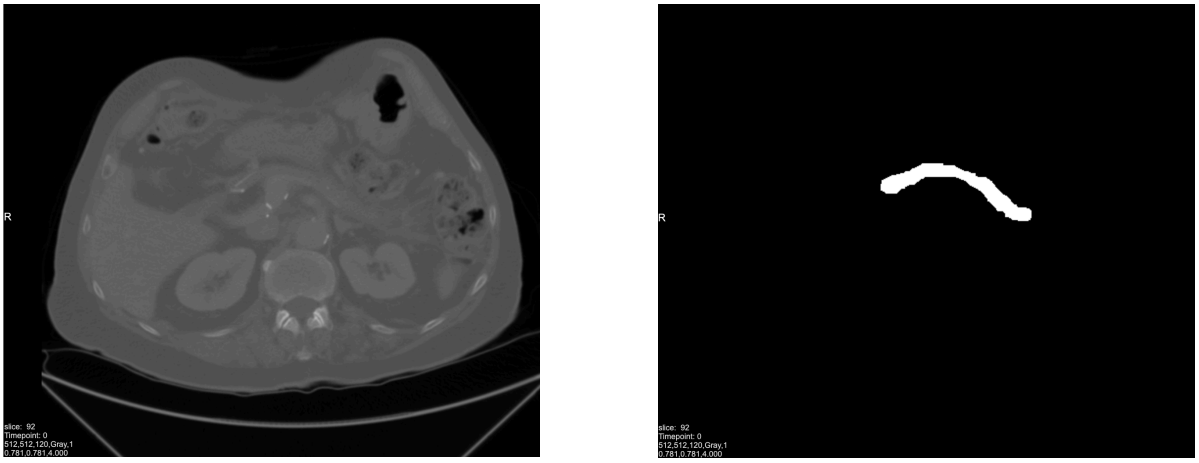
Figure 3.1: An example CT scan of the NIH dataset [107]. This specific image is slice 102 from PANCREAS_0001.

3.1.2. RACU Dataset

In addition to the publicly available dataset of NIH, a dataset from the Regional Academic Cancer Center Utrecht (RACU), University Medical Center Utrecht (UMCU) and St. Antonius Hospital Nieuwegein has been used to finetune the algorithm on postoperative pancreases. This dataset contains 81 contrast-enhanced abdominal CT scans of individual patients who underwent pancreatic resection, either a pylorus-preserving pancreatoduodenectomy (PPPD), Whipple procedure, or distal pancreatectomy. These were all contrast-enhanced CT scans according to CT pancreas protocol, taken during the portal venous phase. Fifteen of the CT scans were performed after pancreatic resection for a benign indication, while the remaining 66 CT scans were performed after pancreatic resection for either malignant or benign disease. In case the pancreatic resection was performed for malignant diseases, the CT scans were performed less than 1 month after surgery to minimize the risk of annotating disease recurrence. Resection types included pancreaticoduodenectomy, in which the head of the pancreas is removed, and distal pancreatectomy, in which the body and the tail of the pancreas are removed. The scans have an equal resolution compared to the NIH dataset of 512x512 pixels and varying numbers of slices, with a slice thickness between 1.5mm and 4mm. On all CT scans, the pancreas was annotated by one postdoctoral researcher of the divisions imaging and oncology, and surgery, a Ph.D. candidate in surgery, or a radiologist in training, with all scans being verified by an experienced radiotherapist with extensive expertise in this topic. The scans were acquired in DICOM format and then transformed into NIFTI files by the author, using the PlatiPY Python library [97]. An example of a slice of one of the scans can be seen in Figure 3.2. This specific CT scan was taken from a patient six days after a pancreaticoduodenectomy. Since this research was done in collaboration with the RACU, the data could be securely transferred from the servers from the RACU to the ones of Datacation in order to train and evaluate a model.

3.1.3. Preprocessing

As customary in the pancreas segmentation on the NIH dataset [107], all intensity values are clipped to the range $[-100, 240]$ [72, 16]. Furthermore, the input is scaled between 0 and 1 to speed up the convergence and prevent gradient overflow. The voxels are then spaced such that all voxels have a size of $1mm \times 1mm \times 1mm$, which helps make the algorithm agnostic for CT scans of different machines. For the CT scans this is done using trilinear interpolation, whereas the labels use nearest neighbors. The nearest neighbor for labels is used, since labels have to be either 0 or 1, and thus cannot be a value in between. Since the models generally consist of many parameters, $10 - 50M$, and the data consists of large 3D volumes, $30 - 50M$ values, the possibilities of training the model on the full volume for all scans in the batch are limited. Because of that reason, the scans are first cropped in the X and Y dimensions to only include the columns that contain intensity values after which the pancreatic slices are selected with a margin of 10 slices on both sides with a minimum of 64 slices. Currently, these



(a) CT scan **(b)** Mask

Figure 3.2: An example CT scan of the NIH dataset. This specific image is slice 92 from PANCREAS_110006.

slices are selected based on the mask, which is unsuitable for a final version of the predictor since the mask would not be known. However, finding the corresponding slices can be done by a lighter and faster algorithm since high accuracy is not required. Lastly, a new random sample of $128 \times 128 \times 64$ voxels is taken from every CT scan in every epoch, with a probability of 50% that the center voxel is a pancreatic voxel. This 50% is chosen to have enough samples without pancreatic voxels, but still be able to converge fast.

3.2. Model Architecture

In reviewing the current state-of-the-art in Section 2.5, both the PanKNet [99] and MHSU-Net [72] algorithms showed significant improvements regarding the DSC, where the latter made the most significant difference with the Multi-scale Convolutional Block (MCB). PanKNet and MCB are both based on the idea that simple convolutions in a basic UNet are not enough to grasp the complex characteristics of the pancreas. Whereas PanKNet uses different decoders to create a multi-head prediction, the MCB uses multiple branches with differing amounts of consecutive convolutions in order to obtain more information. This underlying concept is particularly useful for the postoperative state, since the target, i.e. the pancreas, is even more variable. This will, therefore, be the starting point for the experiments for the postoperative state.

Since the intention of this research is to create an architecture that is capable of segmenting the pancreas with state-of-the-art performance, specifically on postoperative data, the design was chosen to incorporate elements of the current preoperative state-of-the-art that can theoretically be deemed most promising when it comes to postoperative data. Because of that, it was chosen to take a combination of PanKNet and MUNet, namely MKNet, as a starting point. The MKNet architecture is somewhat different compared to PanKNet as described by Proietto Salanitri et al. [99]. The encoder part makes use of a similar implementation as the one described for MUNet, which means that every down layer consists of two consecutive MCBs followed by an activation, normalization, and downsample module. Every encoder layer is then followed by its own decoder, which means the full network consists of a number of decoders equal to the number of layers. The decoder consists of a transpose convolution to upsample the feature maps from the previous layer, followed by two consecutive 3D convolutions, an activation, and a normalization module. This results in an output with $n \times 2$ output channels, where n is the number of layers. These predictions are then concatenated and fed as input to the final convolution with a kernel size of $1 \times 1 \times 1$, which then results in the final prediction. The full architecture can be seen in Figure 3.3.

The MCB as implemented for this research differs slightly from the approach of Ma, Zou, and Liu [72] as can be seen in Figure 2.16b. First of all, the new MCB uses 3D convolutions, since that is more suitable for the 3D network in which it will be placed. Furthermore, the first convolution, which divides the number of channels by two, is shared among all three branches. The benefit of having a shared convolution, instead of having a parallel one for every branch, is the reduction in the number

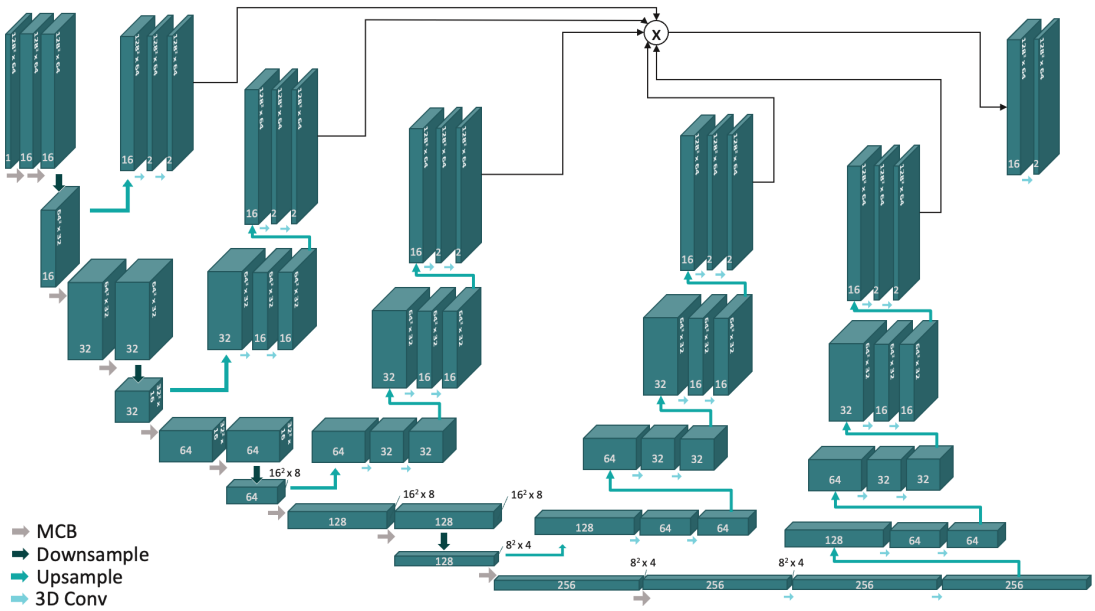


Figure 3.3: The MKNet architecture

of parameters and computations. Particularly since this 3D variant already comes with an increased number of parameters and computations, and preliminary results did not show a significant decrease in performance. Apart from that, the specifics, e.g. the kernel sizes, are equal. In Chapter 4, experiments are conducted to evaluate whether other parameters could have a positive impact on the performance.

As already discussed in Section 2.3, both the normalization as well as the activation functions bring significant benefits if included in a model. Even though Batch Normalization is very often used in prior work [147, 72, 99, 129], it was decided to use Layer Normalization (LN) instead. Using the preprocessing as described in Section 3.1.3, every batch contains random samples of the CT scan. As a consequence, normalizing over the batch dimension seems wrong since the underlying data can be very different, i.e. one of the samples could be mostly covering the lungs or the liver, whereas another one could be precisely including the pancreas. Because of that reason, it is chosen to make use of the LN method, which normalizes over the channel dimension. This way, the different data distributions of every sample in the batch will not influence the normalization, theoretically allowing for a better sample-specific prediction. One could follow a similar argumentation in favor of using Instance Normalization, which normalizes over every channel for every sample. This, however, will, theoretically, not allow for channels to train to include very different information. Every channel will be normalized over just the data of that channel. As a consequence, the different channels will have comparable distributions allowing for less divergence of information in the channel dimension.

As for the activation function, ReLU as discussed in Section 2.3 is by far the most often used activation function. As already identified before, ReLU brings the negative aspect of never-activated neurons, which can negatively impact convergence, as well as the fact that it is not dependent on the data distribution (which led to the development of ABReLU [25]). Since Mish does not have this problem, it was able to outperform a substantial amount of other activation functions, and, in the CUDA variant, does not come with a significant extra computational cost, it was chosen to use this as activation function.

In Section 2.3, it became clear that using dropout mechanisms can help generalize the model and thus improve the performance on the validation set. Despite the fact that the author acknowledges this can be extremely useful, the starting point does not include a dropout mechanism. Dropout mechanisms can hide the effects of other techniques since part of the network will be disabled. Furthermore, it brings the most added value in a scenario where the performance on the training data is relatively high compared to the performance on the validation data. Since performance on both will be measured, dropout mechanisms can be added later if there is a need for it.

One of the characteristics of a U-shaped network is the fact that the images get downsampled and

upsampled. As one could have seen in Section 2.3, there are a vast amount of different methods to do this either with pooling or subsampling operations and their opposite operations. Two of the more commonly used methods for downsampling the images are regular convolutions with a stride of 2, i.e. a subsampling operation, and max-pooling, i.e. a pooling operation, where max-pooling is often combined with average-pooling to retain more information. In combined max- average-polling, the result of both pooling operations is concatenated in the channel dimension after which a convolution ensures the output has the correct number of channels. As a baseline, it was decided to use convolutional downsampling according to the example of Springenberg et al. [117]. This convolutional downsampling layer did not necessarily lead to a significant increase in performance, but the fact that it is capable of learning a strategy is something the complex problem of the variable pancreas can benefit from.

The model will output a prediction in a one-hot format in which the tensor values will be in an undefined range. In order to come to a final prediction with every voxel value belonging either to the pancreas or to the background, some postprocessing needs to be done. Following the example of Proietto Salanitri et al. [99], softmax is used to convert the two-channel tensor values in an undefined range to values for which the sum of the two-channel values for every voxel sum to one, e.g. a voxel can have a chance of 0.25 to belong to the background and a chance of 0.75 to belong to the pancreas. The channel with the highest value is then set to 1, whereas the other channel is set to 0 to come to the final binary prediction, i.e. an argmax function is used.

Even though the network is meant to learn the precise pancreatic tissue, it can occur that some noise occurs in the prediction, i.e. very small groups of voxels that are not situated near the pancreas are falsely classified as pancreatic voxels. Since it is known that there can only be one pancreas per CT scan, an additional postprocessing step is added, namely the selection of the largest connected component. In other words, after the final prediction, the largest connected component is selected using the *KeepLargestConnectedComponent* (KLC) function in MONAI, after which all voxels that are predicted to be pancreas and do not belong to the largest connected component are set to background voxels. Due to this measure, the noise will be removed from the prediction, optimizing the predictive performance. One of the assumptions that is made by using this postprocessing step is that the prediction of the pancreas indeed is the largest connected component within the prediction. If this is not the case, the performance will immediately go to zero. Particularly for the postoperative CT scans, this assumption might not always hold true, which is why the performance of both predictions, i.e. with and without noise, are logged.

3.3. Training and Validation

For the architecture to be able to learn the weights that result in optimal performance, the training process needs to be designed carefully. During every epoch, the algorithm needs to have a way to see how close its prediction is to the ground truth and what weights need to be adjusted to come even closer. This is done by formulating a loss function. The loss function used for training the MKNet in these experiments is a combination of Dice Loss and Focal Loss. As researched by Ma et al. [74], this compound loss performs very well. Particularly for the great imbalance of the data, the possibility of weighing the different classes in the Focal Loss is beneficial. The Dice Loss is defined as in Milletari, Navab, and Ahmadi [79] with squared denominators, to even better address the imbalance. The Focal Loss is weighted with 1^{-3} for the background voxels and 1.0 for the pancreatic voxels. These weights are based on the occurrences of the background and pancreatic voxels. As found by Ma et al. [74], the background voxels occur over 500 times more often than the pancreatic voxels on average in the NIH dataset [107]. Since the algorithm specifically needs to perform well on the class of pancreatic voxels, it was chosen to make the focal weight of the pancreatic voxels not 500 times greater, but 1000 times greater. The γ parameter of the Focal Loss is set to 2, as recommended by Lin et al. [68]. The Dice Loss and Focal loss are then equally weighted in the computation of the final loss.

The chosen optimizer was the Novograd optimizer [33], with a learning rate of 0.001. Even though the Adam optimizer [59] is more often used in segmentation problems, the Novograd is not a conceptually different optimizer. Theoretically, it should not converge to a lower DSC and is found to be more stable [30]. In the specific problem at hand in this research, it is also useful because of the higher memory efficiency due to using layer-wise second moments [65]. These layer-wise second moments, compared to element-wise moments, are particularly interesting for large models with many parameters, as is the case for the models used in this research. In extensive preliminary experiments in this

research, both optimizers showed to converge to approximately the same score, whereas the Novograd optimizer converged a factor 2-8 times faster. Since the learning rate is optimizable in Novograd as well, tweaking the learning rate will theoretically not make a big difference in performance. This is also the reason no learning rate schedulers were used. A *GradScaler* is used to ensure that the weights do not overflow which could result in NaN errors.

Since the dataset is limited, the need for optimal use of the data is high. This research makes use of cross-validation (CV) [5], in order to mitigate the dependency on the fold characteristics improving the robustness. Specifically, K-fold CV is useful when the data consumes a significant amount of memory or training is computationally intensive, both are the case in this research. K-fold CV can be implemented by splitting the initial dataset $\mathcal{S}$ into K subsets $\{\mathcal{S}_1, ..., \mathcal{S}_K\}$ where:

$$|\mathcal{S}_1| \approx |\mathcal{S}_2| \approx ... \approx |\mathcal{S}_K|,$$
$$\cup_{i=1}^{K} \mathcal{S}_i = \mathcal{S},$$
$$\mathcal{S}_i \cap \mathcal{S}_j = \varnothing, \forall i, j \in 1, ...K, i \neq j$$

where $|.|$ denotes the cardinality of a set. The splitting of $\mathcal{S}$ is done by randomly shuffling the data and then splitting it into equal-sized subsets to improve the generalization of the folds. The value for K used is 4, as is also customary in much other research when using the NIH dataset [147, 77, 141, 72, 99, 129, 16].

Even though the training of the algorithm could be continued indefinitely, the performance of the model showed to eventually converge and mostly stop improving in all experiments run. To ensure the computational resources are not used redundantly, a suitable termination function needed to be designed. As one will be able to see in Section 3.4, different metrics will be measured, including the DSC of the final prediction and the loss function. The termination function essentially needs to assess whether the training has converged to optimal performance on the validation set. This is why the value of the loss function will not be used as a termination criterium, since this only measures the performance on the training set. Moreover, because of the random cropping in the preprocessing, the computed loss might fluctuate based on the specific cropped regions of the data every epoch. Instead, the DSC on the validation set will be used. This score measures the actual performance on the validation set, and, since the validation data is not cropped, it is also not dependent on the randomly cropped regions.

Whereas the training data is only 128x128x64 pixels every epoch to reduce the memory load, the validation data cannot be cropped. After all, one wants to predict the full volume, not just a part of the volume. Since the model is trained on the 128x128x64 regions in the training set, one cannot simply input the full volume to the same network and expect similar performance. Instead, the network performs best when it receives equally-sized input. In order to adhere to both the requirement of the model when it comes to input size as well as the desire to predict the full volume, sliding window inference is used. Specifically, the implementation within the MONAI library [20] is used. This function essentially divides the input volume into a number of windows of a dimension (H, W, D), with a specified overlap, after which every individual window is evaluated by the model. The values for the predicted voxels for which multiple windows overlap can be computed by either taking the average of the values in the different windows or weighing them differently based on whether they are near a border or not with a Gaussian function. For this research, the dimensions H, W, and D are set to 128, 128, and 64 respectively, the overlap to 0.25, and the method for blending the output of overlapping windows to constant, i.e. all values receive equal weight.

Due to the fact that multiple windows need to be predicted by sliding window inference, the computational costs of predicting a validation sample are substantially larger than for predicting a training sample. An epoch in which the algorithm is trained on the full training set and validated on the full validation set can easily take up to 5-8 times longer than an epoch that just trains on the full training set without any validation. To speed up the experiments, it was chosen to only validate the validation set once every 25 epochs or when a new minimal loss was achieved while the previous best prediction was already higher than 80%. This greatly reduces the computational costs, particularly at the beginning of the training, while still being capable to validate the best-performing models when looking at the training set.

3.4. Evaluation Metrics

Up until now, almost all research has been solely evaluating their methods in a quantitative manner on the same dataset. The advantage of this is that it can be easily compared to other models or research and gives an impression of the performance of a model. However, when one would truly want to use the algorithm, the clinical performance, i.e. the qualitative performance, needs to be evaluated as well. In the following section, both the quantitative as well as the qualitative evaluation metrics are described that are used for this research.

3.4.1. Quantitative Evaluation

In order to evaluate the quantitative performance of the network during the training and of the final model, a couple of things were measured. First of all, the loss of the training set was registered every epoch. This gives an indication of the performance of the model on the training set. To be able to measure the performance on the validation set, the model is validated on the validation set every 25 epochs as discussed in Section 3.3. This results in a DSC based on the "raw" prediction, with the only postprocessing being a softmax function and an argmax function to make the predictions for every voxel discrete. As discussed in Section 3.2, additional postprocessing is done by applying the KLC operation. To be able to evaluate the necessity of this postprocessing step, the DSC was then computed again so that both before and after the operation the DSC can be seen.

To be able to grasp a full picture of the performance of the models, additional metrics are computed. For all scans in the validation set, after the training has converged, the Normalized Surface Distance (NSD), the 95% Percentile Hausdorff Distance (HD95), and the Hausdorff Distance (HD) will be computed as described in Section 2.1.3. For NSD, the implementation of DeepMind [121] is used. The threshold values are set to 3mm for the X-and Y-axes and 2mm for the Z-axes. This differentiation is made because of the way the scans are annotated. This is, namely, done in a slice-per-slice manner, which implies that the variability in the Z-axis would be less than in the other axes. The exact threshold values are somewhat arbitrary, and further research would need to evaluate if these values are valid. Even though the KLC prevents any outliers from being in the prediction, meaning the prediction will be a solid volume without any voxels outside of this volume, the decision was made to still compute both HD and HD95. It is known the pancreas is a solid volume as well, without any other pancreatic voxels outside of this volume, suggesting that HD95 would in this case be equal to HD. However, it can still be that outliers are attached to the largest volume, which would make HD still sensitive for outliers. Because of this, both will be measured to evaluate whether or not significant outliers remain.

3.4.2. Qualitative Evaluation

Even though quantitative results can give a lot of insights into the performance of the architecture, the objective is to be able to use this algorithm in a clinical environment. In order to assess the usability of the model, therefore, not only the quantitative but also the qualitative performance should be taken into account. Since this requires time from an experienced radiologist, it cannot be done for every scan in every model. However, for the optimal model, the results can be checked.

Similarly to the workflow as described in Section 3.1, the two-step process of annotation by a less experienced medical expert and verification by an experienced radiologist can be used to assess the performance of the model qualitatively. The model can predict the scans after which the radiologist can then check the predictions as if they were made by a master's student in medicine. The number of times the radiologist makes corrections gives an indication of the performance of the model. Every CT scan is classified as one of the following:

- No adjustments
- Minor adjustments
- Substantial adjustments
- Major adjustments

where *No adjustments* indicates the scan is clinically useful without any adjustments, *Minor adjustments* indicates only a small adjustment is necessary, *Substantial* indicates a prediction that is not completely useless but does require multiple or a single large adjustment, and *Major* indicates a prediction is pretty much useless and requires a completely new annotation. After assessing all scans, the radiologist is asked how likely they would use the model again to do the pre-annotations, with the answers being:

- Very unlikely
- Unlikely
- Indifferent
- Likely
- Very likely

which can give an indication of the usability of the model.

Since this process takes time, which can be scarce for an experienced radiologist, this cannot be asked of a large number of radiologists. Even though the results of this method would be very subjective if only done by one or two radiologists, it can still bring incredibly important insight into the clinical usability of the algorithm. Due to the collaboration with RACU, and the IMPACT consortium that RACU is part of, five radiologists have been able to evaluate all predictions.

3.5. Implementation Details

Because of the vast number of different deep learning libraries, e.g. MONAI [20] and MLFlow [84], available for Python [126] / PyTorch [95] models, the decision was made to implement the new network in Python 3.10 [126], using PyTorch 1.13.1 [95]. The experiments were mostly run on an NVIDIA RTX A6000, in addition to DelftBlue [11] for some more experiments on the NIH dataset [107], using CUDA 11.6 [91]. To be able to track all experiments, the MLFlow library [84] was used. Most graphs found in Chapter 4 will also be created using the MLFlow library [84]. For all pre- and postprocessing, the implementations of MONAI [20] were used, just like for the loss function and optimizer.

In Chapter 4, results will be shown for UNet [106], AttentionUNet [93], PanKNet$_{light}$ [99], PanKNet [99], MUNet [72], and the in this research proposed MKNet. For the experiments of UNet and AttentionUNet, again the implementation of MONAI [20] was used. For PanKNet$_{light}$ and PanKNet, the exact same implementation as published on GitHub by the authors was used [41]. Since Ma, Zou, and Liu [72] did not publish their code, a new implementation needed to be created. The UNet implementation of MONAI [20] was used as a basis. The down layers, however, were changed to implement two consecutive MCBs in every layer, followed by a downsampling convolution, as described in the paper. Differently than in the original paper, the implementation of MUNet [72] for this research was a 3D model. The implementation of MKNet was made from scratch, as a *torch.nn.Module*, following all the building blocks as already described in Section 3.2.

4

Experimental Results

The following chapter will shed more light on the experiments that were conducted, in order to come to a final segmentation model. First, the setup of the experiments will be addressed, including a description of the data (processing) and number of runs. Afterward, the results of the experiments will be displayed.

4.1. Setup

As was also experienced during this research, it can be difficult to exactly reproduce prior work with the reported performance scores. To enable future researchers to reproduce the experiments conducted in this research as well as possible, a detailed description of the setup of these experiments will be described in the following section. This also allows future researchers to evaluate the strengths and weaknesses of the experiments. Again, a short description of the data will be given, after which the exact experimental process will be described.

4.1.1. Data

There are two different datasets used for the experiments. In order to make this research comparable to other research in pancreas segmentation, the NIH dataset [107] is used to pretrain the networks and retrieve a DSC score that can be compared with previous research. This data consists of 82 contrast-enhanced CT scans of healthy patients, with an annotated pancreas. Each scan has a resolution of 512x512xD, where D is the number of slices in the range $[181 - 466]$. The other dataset consists of 81 CT scans of the RACU, which have also been contrast-enhanced. The resolution of each slice is 512x512 as well. The CT scans in this dataset are taken within one month of pancreatic resection. The resection is done according to one of the following procedures: pylorus-preserving pancreatoduodenectomy (PPPD), Whipple procedure, or distal pancreatectomy. A more elaborate description of both datasets can be found in Section 3.1.

In both pretraining as well as the final training on the RACU data, the dataset is divided and trained using 4-fold cross-validation. Again, this is done to make the metrics comparable to prior research [147, 77, 141, 72, 99, 129, 16], and to mitigate overfitting to specific fold characteristics. All pre- and postprocessing steps can be found in Section 3.1.3 and Section 3.2.

4.1.2. Experiments

To be able to properly compare the performance of the different networks and network configurations, the results reported in the following are the average over the performance of all four folds, i.e. every fold is run independently. All reported standard deviations are not the standard deviation between the final results of the different folds, but the average over the standard deviations of the distribution of all individual CT scans for every fold independently. The author is aware that there is stochasticity in the experiments, which means multiple runs for all folds would yield more reliable results. However, due to the duration of every experiment and the availability of computational resources, it was chosen to only run every fold once. This should already give a good indication, on which future research can build upon. All scores reported in this chapter are the results of solely the validation set, i.e. data that the model has never seen before.

The experiments were run with a batch size of 8, as this generally leads to a good performance and seems to be a popular choice in this field of research. An optimal batch size can improve both the space and time complexity of an algorithm. Using a batch size equal to the size of the complete dataset would not fit in memory during a training epoch, and could also rise poor generalization of the data in deep learning models [18]. On the other hand, the larger batch size is generally associated with more accurate gradient estimates [48]. In research into the effect of batch size on the performance of 3D medical semantic segmentation, Sato and Kido [114] found that, up to a certain point, a larger batch size will lead to better performance. A batch size of 64 did, however, not improve the performance when compared to a batch size of 16.

Even though all the results reported in this research are based on the exact configuration as described, many more experiments have been conducted with different parameters, e.g. image size, learning rate, optimizer, etcetera. Since these experiments were usually conducted in a less strict way, e.g. only for a single fold, they are not included in this report. However, all these experiments have contributed to the decisions made in the formal experiments as they led to new insights and argumentation for the use of specific parameters.

During the training on the NIH dataset [107], every time the DSC on the validation set reached a new best, the state of the model, as well as the state of the optimizer were saved. After the termination criteria, as described in Section 3.3, were met, the lastly saved states of the model and optimizer were used for the next step. In this next step, all parameters were set exactly the same as in the first step, except for the state of the model and optimizer which were set to the best-performing model found in step one. In the second phase, the model was then further trained on the RACU data, which, after meeting the termination criteria again, led to the final model and metrics. In between the first and second phases, the model from phase one was used to predict the validation set of the RACU data of phase two. This was done to display the difference between the preoperative segmentation problem, compared to the postoperative segmentation problem. Particularly, in performance when only trained on preoperative data.

4.2. Quantitative Results

Different experiments have been set up. Firstly, the models were trained and evaluated on the NIH dataset [107] in order to see how the models and variations performed compared to the current preoperative state-of-the-art. Afterward, a selection of variants was further trained and evaluated on the postoperative data. The models trained on the preoperative data were used as starting point in the postoperative experiments.

4.2.1. Preoperative

Many different experiments have been run on the preoperative NIH dataset, both for existing architectures as well as for the new MKNet architecture. In the following subsection, the baseline models are described including their performance on a variety of evaluation metrics. Afterward, multiple variations of the MKNet are described including accompanying results and explanations of experiments that are conducted on these variations.

Baseline

In order to compare the different architectures to the current state of the art, a baseline was set to see the performance in this specific setup for the different, already existing, networks. To be able to run these experiments, the implementation needed to be available so that it is not a changed implementation that causes a different result. These experiments were again run in exactly the same conditions as already described in Chapter 3 and Section 4.1, which is why the results can differ from the reported scores. All reported and found DSC scores can be found in Table 4.1 as well as the found HD, HD95, and NSD scores where the best score for every metric is in bold.

The scores of PanKNet, even though the exact publicly available code was used, do not come close to the original ones as reported in Proietto Salanitri et al. [99]. It could be the case that the experiments that led to the highest scores were run with a slightly different code, which is supported by the fact that there are some discrepancies in the code when compared to the parameters in the paper. However, these differences are solely in the pre- and postprocessing of the data, which should not lead to such a big difference. The most striking difference is also the fact that PanKNet performs significantly worse than PanKNet$_{Light}$, which is contrary to the results reported by Proietto Salanitri et al. [99].

In the case of UNet and Attention UNet, the implementations available in the MONAI library are used. In the case of UNet, the original architecture of Ronneberger, Fischer, and Brox [106] is extended with residual units as described in Kerfoot et al. [56]. Many different scores have been reported for UNet in the segmentation of the pancreas, varying from 71.8% in the original UNet paper [108] up until 87.6% [72]. These differences can partly be explained by different data processing methods and minor changes in the network architecture. To give a representative number that seems to be close to the median, the score reported in Oktay et al. [93] was used. For both UNet and Attention UNet, the found DSC is almost exactly equal to the reported DSC.

For MUNet [72], there is no code publicly available. This means the baseline set for these experiments is created with my own implementation based on the reported architecture. Effectively, this boils down to the implementation of MONAI's UNet with additions since the authors report that MUNet essentially is a regular UNet with added MCBs. The decoder remains equal to the one implemented by MONAI, apart from using a different normalization method, namely layer normalization instead of batch normalization, which was addressed in Section 3.2. In the down layers of the encoder of the network, the two regular convolutions have been replaced by two consecutive multi-scale convolutions. These MCBs are implemented exactly as described in Section 3.2. This also means the MUNet used for these experiments is a 3D network instead of the 2D network in Ma, Zou, and Liu [72]. Because this brings added computational complexity, the decision was made to reduce the number of channels from $[64, 128, 256, 512, 1024]$ to $[16, 32, 64, 128, 256]$ to speed up the experiments. This might come with reduced accuracy, but the delta of this reduced accuracy will be left for future research.

The results reported for MKNet are the results found for the newly designed architecture which is first proposed in this research. These results particularly are from the standard configuration, exactly as described in Section 3.2, except for the downsampling module which was the MaxAvg instead of a convolutional downsampling module.

Apart from differing results for the reported DSC compared to the found DSC, it is interesting to see the difference in performance when it comes to the other metrics. It becomes apparent that a high DSC does not automatically result in good values on the other metrics. Whereas MUNet has a top-3 found DSC, it is among the worst when it comes to the Hausdorff distances. This means that, although a great portion of the voxels is correctly classified, the wrongly classified voxels are relatively far away, i.e. the Euclidean distance is large. Intuitively, predictions with a lower Hausdorff distance have a more similar shape to the ground truth, which could suggest a better clinical performance.

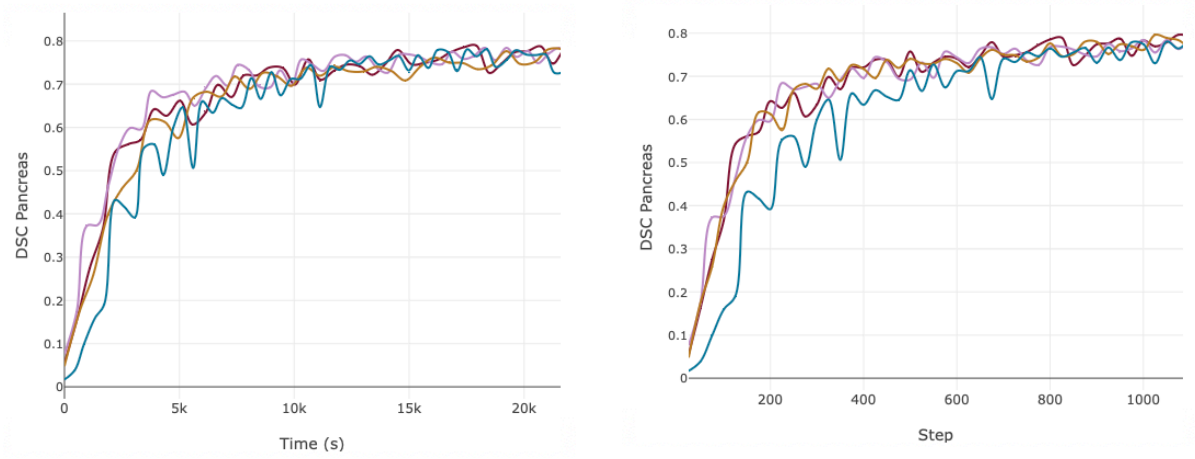
This is where the MKNet displays quite an improved performance over the current state-of-the-art. Both the HD and HD95 are the best for MKNet and the standard deviation is significantly smaller compared to the other models. This means MKNet is capable of finding the best-shaped prediction more consistently for all CT scans in the validation set. Looking back at Table 2.1, the only state-of-the-art model that reported its HD was the Dual-input V-mesh model of Wang et al. [129] with an HD of 18.41 ± 28.10, showing similar performance to PanKNet but with a significantly higher standard deviation. These results stress the importance of using multiple evaluation metrics for evaluating the model. Even though the DSC gives a good indication of the semantic segmentation results and is widely used in pancreas segmentation algorithms, it does not provide a full image of the performance of an algorithm.

Method	Reported DSC	Found DSC	HD (mm)	HD95 (mm)	NSD
UNet [106, 56]	82.0% [93]	$82.14\% \pm 5.45$	15.53 ± 08.13	6.90 ± 05.72	$40.51\% \pm 6.91$
Attention UNet [93]	83.1%	**$83.07\% \pm 4.88$**	15.65 ± 09.18	7.08 ± 07.13	**$42.67\% \pm 6.51$**
PanKNet$_{Light}$ [99]	87.13%	$79.84\% \pm 7.57$	16.61 ± 11.86	8.52 ± 10.13	$34.76\% \pm 6.21$
PanKNet [99]	88.01%	$76.85\% \pm 6.96$	18.34 ± 11.08	9.62 ± 08.93	$29.06\% \pm 5.30$
MUNet [72]	**88.08%**	$81.82\% \pm 7.13$	16.93 ± 12.53	7.92 ± 10.14	$40.91\% \pm 7.23$
MKNet (ours)	-	$81.48\% \pm 6.36$	**13.64 ± 5.40**	**5.89 ± 03.62**	$39.17\% \pm 7.24$

Table 4.1: Comparison of the performance of the current state-of-the-art methods on the NIH dataset [107]

MKNet Variations

Layers In most research, UNet-based models all have five layers. One could conclude that five layers allow for enough parameters to find the underlying distribution, without the problem of overfitting to the training data. Since the pancreas, however, is extremely variable and relatively small, it might be the case that a different number of layers is more suitable for this experiment. Because of that, different numbers of layers were compared to each other on their performance. For the implementation of the MKNet, three, four, five, and six layers were tested. The results obtained are the average achieved DSC for one run on all four folds. The achieved DSC for the three, four, five, and six layers was 80.40%, 79.9%, 80.8%, and 80.3% respectively when using a convolutional downsampling layer. Interestingly, when looking at the convergence within the first 5.5 hours, i.e. 1000 epochs, on fold 1, the convergence is barely shaped any different, particularly when looking at the time. As can be seen in Figure 4.1, the DSC per epoch converges approximately the same. Looking at the convergence compared to the number of epochs, it can be seen that the 3-layer network converges slightly slower at the beginning, whereas the other networks are approximately the same. Even though the larger networks have significantly more parameters, the convergence is roughly equal indicating that the number of layers does not come at a large cost.



(a) Relative to time

(b) Relative to epochs

Figure 4.1: The convergence of fold 1 for the MKNet for 3 (blue), 4 (pink), 5 (red), and 6 (brown) layer networks

When looking at the other evaluation metrics in Table 4.2, the 5-layer network is generally the best option, scoring the lowest HDs and the lowest standard deviations in all metrics. Even though the 4-layer network performs well on the HDs as well, the network is likely not big enough to contain all the necessary information to outperform in terms of DSC and NSD. It can also be seen that increasing to a 6-layer, does not improve performance on the validation set anymore. It is likely overfitting too much on the training data, showing a decrease in performance on the validation set compared to the 5-layer network. These additional metrics also show that the DSC is not the best single source of truth, as the 3-layer network performs well on the DSC, but the worst for HD and HD95.

# layers	DSC	HD (mm)	HD95 (mm)	NSD
3	$80.40\% \pm 7.23$	19.28 ± 14.23	9.67 ± 11.85	**$38.42\% \pm 7.24$**
4	$79.90\% \pm 7.84$	16.35 ± 10.49	8.09 ± 07.91	$37.99\% \pm 7.54$
5	**$80.78\% \pm 6.21$**	**16.16 ± 08.04**	**7.42 ± 05.73**	$38.18\% \pm 6.57$
6	$80.28\% \pm 7.51$	18.01 ± 12.82	9.09 ± 10.54	$38.20\% \pm 7.25$

Table 4.2: Comparison of different number of layers

Downsampling One of the characteristics of the UNet- / KNet-shaped models is the fact that the images are downsampled and upsampled. It is important for the network to have a way to downsample

the images in such a way that as little information as possible is lost. As one could have already read in Section 2.3, many different pooling methods exist to downsample the images. To evaluate the influence of the different downsampling methods, experiments were conducted with only changing the downsampling module of the architecture. The first method is using a regular 3D convolution with equal in and out channels, a kernel size of 3, and a stride of 2. The second method is a combination of max pooling and average pooling, of which the results are concatenated in the channel dimension. In order to have an equal number of in and out channels, the concatenation is followed by a regular 3D convolution with a kernel size of 1, halving the number of channels. The results of the convolutional and MaxAvg downsampling module on the 5-layer MKNet are 80.78% and 81.47% respectively, when it comes to the DSC. The HD and HD95 are 13.64mm and 5.89mm for the MaxAvg module, whereas the convolutional module does not get better than 16.16mm and 7.42mm respectively. Even though the convolutional module has the potential to learn the weights necessary to retrieve the correct information, the MaxAvg module is capable of converting both the information from the max pooling as well as from the average pooling. The additional convolution can then learn how to combine both results, which, looking at the performance increase, can more accurately save the information for the next layer, particularly when it comes to the shape. Because the MaxAvg seems to be the better option, this setting was used for all experiments reported from now on, unless specified otherwise.

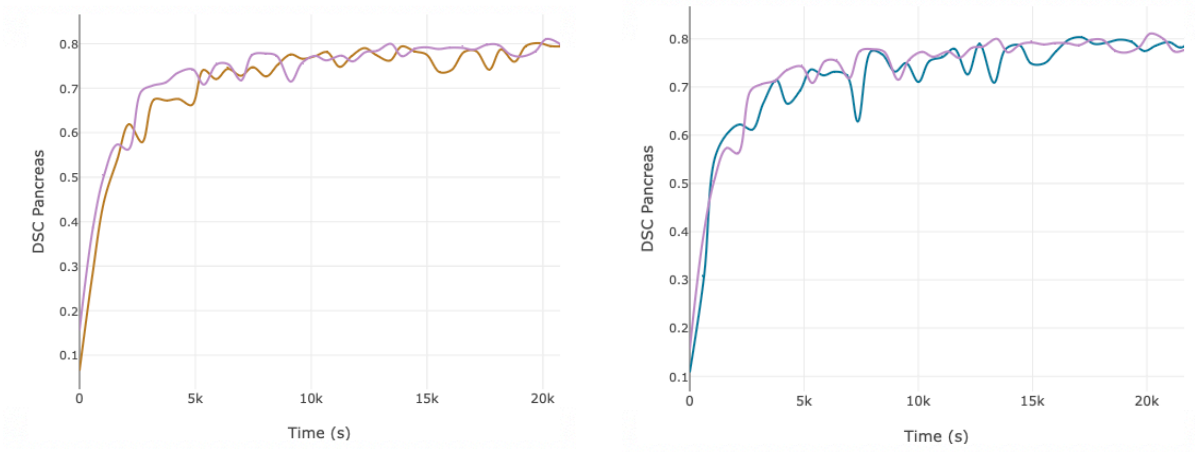
KLC In Section 3.2 the postprocessing is described, using the KLC operation on the final prediction to remove any noise from the prediction. It is hypothesized that this could slightly increase the accuracy of the model since potential little “islands” of pancreatic voxels in the prediction can be removed. As mentioned in Section 3.4, during all experiments both the DSC before and after the KLC operation is measured, making it easy to compare both accuracies. In all conducted experiments, the DSC after KLC was found to be larger than before with deltas varying from 0.2% up to 0.5%. This strengthens the hypothesis that KLC is capable of removing some noise from the final prediction and that it is beneficial for the final segmentation.

Loss Function Even though the Dice loss function is often the common choice for a segmentation algorithm, there has been research in more optimal loss functions [74]. As described in Section 3.3, it was chosen to use a combination of the Dice and Focal loss for this particular training process. To ensure this is indeed a beneficial decision, the Dice Focal loss function was compared to the Dice loss function, both using the implementation of MONAI. In the standard configuration described in Section 3.2, using the MaxAvg downsampling module, the Dice and Dice Focal loss function converged to a DSC of 81.01% and 81.47% respectively. The loss function with the fastest convergence differs a bit per fold, but generally, this does not make a difference. On the HD metrics, the Dice Focal loss shows an improvement of approximately 1mm in both the 100^{th} percentile as well as the 95^{th} in addition to the NSD which is also 1 percentage point higher. This all supports the results from Ma et al. [74] stating the Dice Focal compound loss performs better than the Dice loss.

Activation Perhaps the most simple activation function, and certainly the most often-used one, is the ReLU activation. As hypothesized in Section 3.2, the Mish activation function could improve the performance as suggested by Misra [82]. To verify this is indeed the case, experiments were conducted to evaluate the performance, both in DSC, HD, and NSD as well as in time, of the Mish activation function compared to ReLU. Since the ReLU activation is often combined with batch instead of layer normalization, experiments were conducted using both normalization techniques. Using layer normalization, the DSC for ReLU and Mish were 80.45% and 81.47%, whereas the batch normalization resulted in 80.85% and 80.80%. Looking at all results in Figure 2.5, it is immediately apparent that the normalization methods can have a big impact on the results of the activation function. Whereas the performance of ReLU does not seem to differ too much when changing the normalization method, the results of Mish are completely different. These results could explain why research tends to use batch normalization in combination with ReLU, as ReLU can be a bit more robust to other variables and is seemingly performing better when using batch normalization. In combination with layer normalization, however, Mish is superior in every metric with lower standard deviations as well. As can be seen in Figure 4.2, the convergence curve of both activation functions is shaped approximately the same, supporting the proposition that the CUDA variant of MISH does not come with a significantly increased computational complexity.

Normalization	DSC	HD (mm)	HD95 (mm)	NSD
Layer				
- Mish	**81.47% ± 6.36**	**13.64 ± 5.40**	**5.89 ± 3.62**	**39.17% ± 7.24**
- ReLU	80.45% ± 6.72	15.46 ± 8.58	7.07 ± 5.76	37.56% ± 6.82
Batch				
- Mish	80.80% ± 6.51	17.25 ± 10.16	8.35 ± 8.78	38.66% ± 6.73
- ReLU	80.85% ± 6.82	15.47 ± 06.61	6.76 ± 4.51	38.49% ± 6.96

Table 4.3: Comparison of different activation functions



(a) Layer Normalization for ReLU (pink) and Mish (yellow)

(b) Batch Normalization for ReLU (pink) and Mish (blue)

Figure 4.2: The convergence of fold 1 for the MKNet using the ReLU and Mish activation functions

Normalization As hypothesized in Section 3.2, in the current setup layer normalization should perform better than other normalization methods, e.g. batch, instance, or group. To support this hypothesis, different normalization methods were compared. For group normalization, a group size of eight is chosen as this is one of the best-performing group sizes according to Wu and He [132]. This experiment is particularly interesting as Wu and He stated that group normalization works particularly well on semantic segmentation tasks. The results of the experiments are reported in Table 4.4. Layer normalization indeed performs best on all metrics with the lowest variance, which supports the argumentation given in Section 3.2. Group normalization indeed performs better on the shape-aware metrics than batch and instance normalization as the original authors already claimed. Even though Batch normalization is the second best performing method looking at the DSC, it is the worst in both HD metrics, again highlighting the importance of using different metrics in evaluating a model.

Normalization	DSC	HD (mm)	HD95 (mm)	NSD
Layer	**81.47% ± 6.36**	**13.64 ± 05.40**	**5.89 ± 3.62**	**39.17% ± 7.24**
Batch	80.80% ± 6.51	17.25 ± 10.16	8.35 ± 8.78	38.66% ± 6.73
Instance	80.48% ± 6.91	16.32 ± 09.53	7.73 ± 7.11	38.54% ± 6.73
Group	80.32% ± 8.03	15.62 ± 06.32	7.09 ± 4.63	38.97% ± 7.88

Table 4.4: Comparison of different normalization methods

Kernel Size MCB Even though the MCB as designed by Ma, Zou, and Liu [72] contains only 3×3 convolutions in the branches and the adapted version as proposed in this research, up until now, also only used $3 \times 3 \times 3$ convolutions, it might be beneficial to use differently sized kernels in order to extract

even more information. Since it is hard to argue upfront what would have the potential to perform better, a couple of alternatives were selected to experiment with. It was chosen to design the different kernel sizes in such a way that for every branch, the n^{th} convolution would have equal size, e.g. if the selected variant would have $3^3, 5^3$ and 7^3 kernel sizes then branch one would have a single convolution with a kernel size of $3 \times 3 \times 3$, branch two consecutive convolutions with $3 \times 3 \times 3$ and $5 \times 5 \times 5$, and the kernels of branch three would be with size $3 \times 3 \times 3$, $5 \times 5 \times 5$ and $7 \times 7 \times 7$. In this research, such an MCB with kernel sizes $3^3, 5^3, 7^3$ is formulated as 3-5-7. The author is aware that this is not at all discovering all the different possibilities, but since these possibilities are in fact endless, this was decided to be the scope of this research.

All alternatives consist of cubic kernels with equal size of an odd number in all dimensions as customary in convolutional kernels, which is why a kernel size of 3 in this chapter represents a kernel sized $3 \times 3 \times 3$. Furthermore, kernel sizes of more than 7 would introduce too many additional parameters and would greatly slow down the training and inference time, which is why larger kernels have not been experimented with. The results for all different variants that have been experimented with are reported in Table 4.5 including the number of parameters in the model. As can be seen, an increase in kernel size has an enormous impact on the number of parameters in the model. In particular, a kernel size of 7 results in a huge increase in parameters, which is to be expected since every $7 \times 7 \times 7$ kernel has almost thirteen times more parameters than a $3 \times 3 \times 3$ kernel. It is likely due to this great increase in parameters that the performance on the validation set is lower for the large models. Simply too many parameters have been optimized for the distribution of the training set, such that the model does not generalize well to the validation set anymore. On the other hand, the few parameters of the 1-1-1 kernel sizes can not use enough contextual information to grasp the complexity of the pancreatic tissue. The 3-3-3 kernel size is the best-performing in all metrics, showing the best balance between contextual information and generalization.

Kernel Sizes	# Model Params	DSC	HD (mm)	HD95 (mm)	NSD
1-1-1	$16.5M$	$76.86\% \pm 9.83$	18.74 ± 12.27	9.76 ± 8.78	$35.06\% \pm 8.52$
1-3-5	$49.8M$	$80.36\% \pm 6.54$	14.76 ± 06.47	6.92 ± 4.69	$38.32\% \pm 7.61$
3-3-3	$37.8M$	**$81.47\% \pm 6.36$**	**13.64 ± 05.40**	**5.89 ± 03.62**	**$39.17\% \pm 7.24$**
3-5-7	$127.2M$	$79.96\% \pm 7.30$	16.16 ± 09.66	7.81 ± 07.76	$37.75\% \pm 6.92$
5-3-1	$52.5M$	$79.33\% \pm 7.48$	16.76 ± 09.67	8.13 ± 07.40	$37.77\% \pm 7.96$
5-3-5	$74.1M$	$80.66\% \pm 6.49$	15.60 ± 07.67	7.12 ± 05.42	$38.54\% \pm 6.75$
5-5-5	$108.4M$	$79.71\% \pm 7.22$	16.48 ± 08.88	8.06 ± 06.83	$37.32\% \pm 6.60$
7-5-3	$134.0M$	$79.66\% \pm 9.80$	17.40 ± 12.55	8.85 ± 10.56	$37.43\% \pm 7.56$

Table 4.5: Comparison of different kernel sizes within the MCBs

Partial Convolution One of the difficulties as described in Section 2.3 is the fact that the images need to be padded in order to remain resolution. Often zero padding is used, simply filling the additional border voxels with zeros, but Lippi and Mattiuzzi [70] described a way to treat the padded voxels differently through partial convolution-based padding, reweighting the padded voxels. This could impact performance significantly, particularly through the lower layers as the accumulated border voxels from higher layers take up a big portion of the images. Apart from that, the preprocessing step of cropping the images to a $128 \times 128 \times 64$ cube has as a consequence that it can not be known if the pancreas is in the center of the region or at the border. As their implementation is online available [94], it could be implemented directly into the network replacing all convolutions in the MCBs within the encoder. It was chosen not to replace the convolutions in all decoders, since the impact of border voxels is less there. The partial convolution (PC) variant could not increase the performance. On the contrary, PC scored one percentage point lower on the DSC, 2.5mm and 1.5mm worse on the HD and HD95 respectively, and 0.5 percentage points lower on the NSD, with all standard deviations being higher than the baseline without PC. It is expected that the multiple decoders already ensure that the border voxels, of the lower layers in particular, are weighted differently through the convolutions, making the PC disadvantageous since information gets lost from the border voxels that are convoluted with actual voxels.

Skip Connections and Attention One of the characteristics of the UNet is the presence of skip connections that transfer information from different layers of the encoder to their counterpart in the decoder, without traveling through the lower layers of the network. MKNet as described up until now, does not have such connections since the different decoders should already contain the information from the different encoder layers. However, this information is only combined at the end of all decoders, meaning that an individual decoder only has the information provided by the lowest layer of that decoder. To test the hypothesis that the different decoders make the skip connections redundant, an MKNet variant with skip connections from the encoder layers to their counterpart in every decoder was implemented: the Multi-Scale Skip KNet (MSKNet). Some slight changes in the architecture needed to be applied in order to design such a network practically.

The encoder still consists of 5 consecutive MCBs followed by a downsampling module. However, where in MKNet the output features of the encoder, i.e. the input features for the decoders, are retrieved directly after downsampling, in MSKNet the features are extracted just before the downsampling module. This is to ensure a correct image size for the skip connections. As a direct consequence of this, MSKNet has one decoder less when compared to MKNet as a decoder in MSKNet requires at least two features, one from the lower layer and one skip connection.

As seen in Section 4.2.1, Attention UNet performed quite well on the DSC metric particularly. With MSKNet and the addition of skip connections, it becomes possible to add attention to the network. Another architecture was implemented that is equal to MSKNet, but with the attention gates applied to the skip connections as described in [93] resulting in the Multi-scale Attention KNet (MAKNet).

In Table 4.6 one can see the scores on all evaluation metrics for the different architectures. In terms of DSC and NSD, MSKNet in particular shows quite a performance increase, indicating that the additional information in the skip connections can indeed be used to correctly predict more voxels, even though the amount of parameters is slightly smaller due to the fewer number of decoders. However, in terms of HD, MSKNet and MAKNet both perform noticeably worse than the original MKNet. In the easier problem of the preoperative state, the additional skip connection information does not come with an increased shape awareness, whereas it does come with better border awareness. Experiments in the postoperative state will need to point out what architecture is better in that scenario.

Surprisingly, the addition of the attention gates in MAKNet when compared to MSKNet does not come with any performance increase on any metric. Whereas a UNet structure can greatly benefit from the attention mechanisms, the KNet structure with its multiple decoders seems to perform better without attention.

Model	DSC	HD (mm)	HD95 (mm)	NSD
MKNet	$81.47\% \pm 6.36$	$\mathbf{13.64 \pm 05.40}$	$\mathbf{5.89 \pm 3.62}$	$39.17\% \pm 7.24$
MSKNet	$\mathbf{82.17\% \pm 6.13}$	15.14 ± 08.56	6.71 ± 6.63	$\mathbf{41.32\% \pm 6.93}$
MAKNet	$81.61\% \pm 6.73$	17.02 ± 10.39	7.97 ± 8.17	$40.92\% \pm 7.62$

Table 4.6: Comparison of different architectures

4.2.2. Postoperative Only Pretrained

Since the data was received in batches, the first experimental results were only done on 15 postoperative scans. Although 15 scans are definitely not enough to be able to properly and reliably train and validate a model, it is still interesting to evaluate these 15 postoperative scans with models that have only been trained on preoperative scans, i.e. the NIH dataset [107]. The differences in the scores can give an indication of the complexity of segmenting the pancreas in healthy subjects, i.e. the preoperative state, compared to subjects that have undergone pancreatic resection, i.e. the postoperative state.

In Table 4.7 one can see the found DSC on the preoperative data, compared to the found DSC, HD, HD95, and NSD on the 15 postoperative scans, with the best score in bold. During the preoperative experiments, 4-fold cross-validation was used, meaning that there are four models for every architecture. All values reported in Table 4.7 are the averages of the metrics of the four models, where the standard deviations are not the standard deviations between the folds, but the average over the standard deviations for all four models. As can be seen, the DSC scores are not even close to the

DSC scores achieved on the preoperative data, which supports the hypothesis that the segmentation of the pancreas after resection is indeed a more complex problem to solve. It can also be seen that the standard deviation for all metrics is relatively large. This is primarily due to the fact that the models have a difficult time segmenting two or three CT scans, resulting in a DSC of 0.0% and very high HDs. Looking at Attention UNet, the maximum DSC is 71.01%, which is not necessarily a very low score given the circumstances. There are, however, also two CT scans that achieved a DSC of 0.0%, greatly reducing the mean DSC and greatly increasing the standard deviation. Similar situations arise for the other models.

In the table, one can also find MKNet, which is the architecture presented in this research. For now, the configuration is exactly similar to the base architecture as described in Section 3.2, apart from the downsampling module, which in this case is the MaxAvg module.

As can be seen in Table 4.7, PanKNet$_{Light}$ is capable of achieving the highest DSC on the postoperative data, even though it was the second worst model on the preoperative data. This further supports the proposition that postoperative segmentation is a different problem with a different complexity when compared to preoperative segmentation. For both HD and HD95 metrics, PanKNet$_{Light}$ is also capable of performing the best, suggesting that the shape of the pancreas is best predicted for this model. Looking at the NSD, MKNet is achieving the highest score. The intuitive interpretation is that MKNet shows the greatest overlap with the ground truth by looking at only the border pixels. In other words, 18.49% of the border pixels of the ground truth overlap with the border pixels of the prediction made by MKNet. Looking at HD and HD95, MKNet performs only slightly worse than PanKNet$_{Light}$, but the standard deviation is significantly lower. This indicates that MKNet is capable of predicting the shape of the pancreas more consistently than PanKNet$_{Light}$, indicating a more stable model. This can be due to the MCBs inside MKNet, which allow MKNet to retain more information, which, combined with the multiple decoders, makes the model more robust for different shapes and positions of the pancreas.

Method	DSC Pre-Op	DSC Post-Op	HD (mm)	HD95 (mm)	NSD
UNet [106, 56]	$82.14\% \pm 5.45$	$47.73\% \pm 26.12$	42.20 ± 29.42	35.16 ± 28.60	$18.07\% \pm 10.77$
Attention UNet [93]	**$83.07\% \pm 4.88$**	$46.84\% \pm 24.43$	43.09 ± 31.50	35.58 ± 29.95	$18.31\% \pm 10.30$
PanKNet$_{Light}$ [99]	$79.84\% \pm 7.57$	**$48.75\% \pm 26.07$**	**39.21 ± 27.31**	**31.55 ± 26.49**	$17.09\% \pm 09.92$
PanKNet [99]	$76.85\% \pm 6.96$	$40.36\% \pm 25.68$	46.46 ± 29.49	39.29 ± 28.87	$12.95\% \pm 08.86$
MUNet [72]	$81.82\% \pm 7.13$	$48.10\% \pm 23.31$	40.17 ± 26.89	32.64 ± 25.92	$18.01\% \pm 09.61$
MKNet (ours)	$81.48\% \pm 6.36$	$47.45\% \pm 25.12$	39.86 ± 25.07	32.67 ± 24.57	**$18.49\% \pm 10.77$**

Table 4.7: Comparison of the performance of the current state-of-the-art on postoperative scans

4.2.3. Postoperative

With the full dataset of 81 scans available, it becomes possible to also train on the postoperative data. With the knowledge of all the preoperative experiments as described in Section 4.2.1, there are three standard models: MKNet, MSKNet, and MAKNet. All of these models are 5-layer ones mostly as described in Section 3.2, i.e. using layer normalization, Mish activation, 3-3-3 kernel sizes within the MCBs, and a Dice Focal loss function, but as the MaxAvg downsampling module showed to outperform the convolutional one, the MaxAvg module was used for downsampling.

As the postoperative data is somewhat different from the preoperative data, some more thought needed to be given to the division in folds. All cases underwent pancreatic resection but through different surgeries. These surgeries can roughly be divided into two categories: pancreaticoduodenectomy (PD) and distal pancreatectomy (distal). PD describes the surgeries where the head of the pancreas is resected where distal surgeries aim to remove the body and/or tail of the pancreas. As these different categories of surgeries result in different abdominal characteristics and differing locations of organs, it is important to take this into account when training the algorithm. The dataset consists of 61 scans that fall into the PD category and 20 that are distal. In dividing the dataset into four folds, it was ensured that every fold consisted of 15 (or 16) PD scans and 5 distal scans. This ensures that the model trains on the most representative data and prevents the model from overfitting on one or the other resection type.

The results for MKNet, MSKNet, and MAKNet are reported in Table 4.8. As described in the previous section, these results report the averages over the four fold, with the averages over the standard deviations of each individual fold. It immediately becomes clear that the performance shifts slightly when compared to preoperative. Even though the DSC and NSD are still the best for MSKNet and MAKNet, just as in preoperative, both MSKNet as well as MAKNet are now performing better on the HD metrics as well. The higher variability in the postoperative pancreas seems to require more information being fed from the encoder to the decoders directly through the skip connections.

What is also interesting to see is the far worse performance when compared to preoperative, even though the amount of data is similar and the models are actually pretrained on preoperative data as well. The DSC is ± 20 percentage points lower, the HD 3-7mm higher, the HD95 almost doubled, and the NSD also decreased by ± 15 percentage points. As already stated in Section 4.2.2, it can be concluded that the postoperative segmentation problem is indeed far more difficult when compared to preoperative. Particularly for some extreme cases. There are, namely, several cases where the model predicts completely wrong and the prediction has no overlap with the ground truth. This immediately pushes down the averages and also explains the increased standard deviations for all metrics. Some cases are simply too difficult for the algorithm.

Model	# Model Params	DSC	HD (mm)	HD95 (mm)	NSD
MKNet	$37.8M$	$62.88\% \pm 15.36$	20.27 ± 12.79	13.78 ± 12.01	$25.57\% \pm 09.01$
MSKNet	$35.2M$	**$64.42\% \pm 14.58$**	**18.80 ± 11.83**	**12.68 ± 11.11**	**$27.47\% \pm 8.06$**
MAKNet	$35.2M$	$63.17\% \pm 16.49$	19.37 ± 13.46	13.57 ± 12.84	$26.88\% \pm 8.84$

Table 4.8: Comparison of different architectures

Layers

As the experiments on the preoperative data showed the number of layers can have a big impact on the performance, it is worth experimenting again with different numbers of layers on the postoperative data. Especially, since the variability in the data is so much larger, more or fewer parameters could have a different impact when compared to preoperative data. Furthermore, as the postoperative pancreas is smaller than the preoperative one, a deeper layer does not necessarily add additional information. The results for the 5-layer networks can be seen in Table 4.8, whereas the results for the 3- and 4-layer networks are reported in Table 4.9. It can be seen in the tables that a lower number of layers immediately results in a significant decrease in parameters. Going from a 5-layer network to a 4-layer network reduces the number of parameters by more than a factor of two, whereas a 3-layer network means a reduction of more than a factor of three. Whereas for MSKNet the 4- and 5-layer networks perform rather similarly, the difference for MAKNet is quite a bit bigger. Increasing the DSC by almost two percentage points and decreasing both HD as well as HD95 by 2 mm, while the standard deviation is reduced for all metrics. This indicates a more stable and more capable model. In its 4-layer variant, MAKNet suddenly outperforms all other architectures as well, whereas it was not capable of doing this before. The attention mechanisms seem to "see" better what voxels to focus on, which in combination with the higher generalization due to the smaller network, creates a well-generalized but still specialized network. This is further supported by the fact that the attention modules barely add any parameters to the network.

Even though the scores on the quantitative metrics give a good indication of which model is performing best, it is interesting to see how the predictions differ visually. In Figure 4.3, one can see a slice for the moderately difficult case 111288, where the predictions are done by a 5-layer MKNet, 5-layer MSKNet, and 4-layer MAKNet respectively. It can be seen that the prediction of MKNet still has difficulty to find the correct shape and even classifies non-pancreatic voxels as the pancreas. MSKNet is already better at finding the correct voxels, most likely as a result of the skip connections. However, it is clear that the additional attention in the skip connections of MAKNet is capable of getting very close to the ground truth. Only the tip of the pancreas could not be identified by all networks.

Normalization

The tissue in the postoperative state can be significantly different from the preoperative state. This might lead the convolutions to come to different feature maps with different ranges of values. The

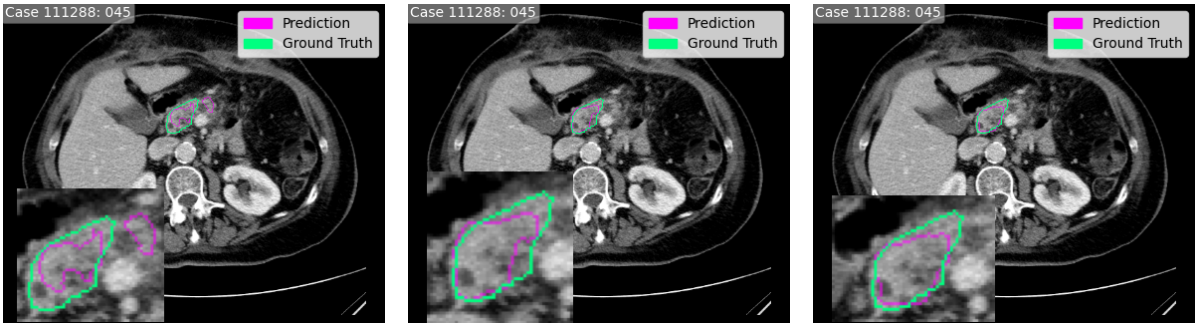
3-Layer networks

Model	# Model Params	DSC	HD (mm)	HD95 (mm)	NSD
MKNet	$10.7M$	$62.27\% \pm 16.29$	21.37 ± 12.56	15.15 ± 11.86	$25.56\% \pm 8.99$
MSKNet	$8.3M$	$63.04\% \pm 13.56$	20.10 ± 12.43	14.02 ± 12.05	$26.26\% \pm 7.54$
MAKNet	$8.3M$	$62.94\% \pm 15.65$	20.27 ± 14.59	14.25 ± 13.40	$26.43\% \pm 8.71$

4-Layer networks

Model	# Model Params	DSC	HD (mm)	HD95 (mm)	NSD
MKNet	$17.5M$	$63.38\% \pm 16.61$	22.36 ± 19.56	16.04 ± 18.53	$26.05\% \pm 9.10$
MSKNet	$15.1M$	$63.82\% \pm 15.13$	19.79 ± 15.61	13.65 ± 12.24	$26.52\% \pm 8.42$
MAKNet	$15.1M$	**64.93% ± 14.76**	**17.33 ± 11.16**	**11.52 ± 10.24**	**27.15% ± 8.19**

Table 4.9: Comparison of different layers for MKNet, MSKNet, and MAKNet



(a) 5L MKNet
DSC: 37.61%, HD: $31.70mm$,
HD95: $25.01mm$, NSD: 16.55%

(b) 5L MSKNet
DSC: 52.84%, HD: $25.02mm$,
HD95: $18.03mm$, NSD: 19.92%

(c) 4L MAKNet
DSC: 57.17%, HD: $15.10mm$,
HD95: $11.22mm$, NSD: 23.59%

Figure 4.3: Predictions of the different architecture for the same slice of case 111288.

distribution of those values is naturally normalized in every layer, but this normalization could have a different impact when compared to preoperative. To ensure that layer normalization is still the best-performing one, as was the case for preoperative, the different normalization techniques were also evaluated on the postoperative data. The results of these experiments can be found in Table 4.10 for a 5-layer MKNet. Layer and instance normalization are the best-performing normalization methods.

Both HD metrics are a bit better for instance normalization, whereas layer normalization is best in the DSC and NSD. This is somewhat surprising since instance normalization did not perform really well on the preoperative data, but apparently, the postoperative situation does suit instance normalization. This could partly be caused by the preprocessing step of taking a random sample from the volume. In instance normalization, all cases and all features of each case are independent. For the postoperative state, the variance between these cases is even bigger and so could the variance in the feature maps. Instance normalization would allow for every feature map in every case, to have its own distribution, increasing the ability to make use of the different feature maps.

Layer normalization on the other hand does show relatively similar performance to the preoperative data. It does have more parameters due to the fact that layer normalization has learnable parameters, but this does not increase the number of multiply-accumulate operations and thus not the computational time. Worth noticing is that group normalization is being outperformed by batch normalization in the postoperative state, whereas in the preoperative state group normalization performed better on the HD, HD95, and NSD metrics. The smaller groups over which the method normalizes seems to be disadvantageous for the different variety in the feature maps.

One might wonder if instance normalization might improve the HD and HD95 for the MSKNet and MAKNet architecture as well. Additional experiments for instance normalization are run for the two best-performing MSKNet and MAKNet configurations, namely the 5-layer and 4-layer ones respectively. In Table 4.11 reports the result for both networks using both normalization methods. It can be seen that for MAKNet and MSKNet instance normalization does not improve the HD and HD95. It does slightly

Normalization	# Model Params	DSC	HD (mm)	HD95 (mm)	NSD
Layer	$37.8M$	**62.88% ± 15.36**	20.27 ± 12.79	13.78 ± 12.01	**25.57% ± 09.01**
Batch	$23.8M$	$61.18\% \pm 16.68$	21.05 ± 14.18	14.99 ± 13.29	$24.87\% \pm 9.59$
Instance	$23.8M$	$62.18\% \pm 16.03$	**19.59 ± 12.72**	**13.71 ± 12.10**	$25.12\% \pm 8.91$
Group	$23.8M$	$60.13\% \pm 18.69$	22.66 ± 15.57	16.18 ± 14.74	$24.92\% \pm 9.50$

Table 4.10: Comparison of different normalization methods on a 5-layer MKNet

improve the NSD for MAKNet, but this difference is neglectable and does not weigh up to the better DSC, HD, and HD95 of layer normalization.

Normalization	DSC	HD (mm)	HD95 (mm)	NSD
4-layer MAKNet				
Layer	**64.93% ± 14.76**	**17.33 ± 11.16**	**11.52 ± 10.24**	$27.15\% \pm 8.19$
Instance	$63.63\% \pm 16.66$	20.73 ± 14.05	14.55 ± 13.23	**27.18% ± 9.45**
5-layer MSKNet				
Layer	**64.42% ± 14.58**	**18.80 ± 11.83**	**12.68 ± 11.11**	**27.47% ± 8.06**
Instance	$63.25\% \pm 17.83$	19.94 ± 12.83	14.05 ± 11.28	$26.86\% \pm 9.12$

Table 4.11: Comparison of layer and instance normalization for 4-layer MAKNet and 5-layer MSKNet.

Fintune Resection Type

Within the dataset, there are two types of resection as already discussed earlier: PD and distal. Since both result in drastically different abdominal situations, it would make sense to have a different model for both. Especially since the type of resection is already known, so it does not require more information or another action from a potential user. In order to finetune the models on the resection type, the following approach was taken. First, the model is pretrained on the preoperative data and trained on all postoperative data, just as for the other experiments. However, after converging, the model is further trained on only the scans from the respective resection type, i.e. the training starts again but all scans from the other resection type are removed from both the training as well as the validation set. This is done in such a way that the model still has not seen any of the scans in the validation set to ensure performance measures on unseen data. For PD, this means the training set now consists of 45 (or 46) training samples and 15 (or 16) validation samples, whereas the distal model is only trained on 15 samples and validated on 5. Although this is an even smaller dataset, it is hypothesized that a short amount of training time on the specific resection could improve the prediction performance on that resection. For these experiments, the 5-layer MSKNet and 4-layer MAKNet were used with the values reported being the averages over the 4 folds.

Resection	DSC	HD (mm)	HD95 (mm)	NSD
Distal				
Before Finetuning	$68.99\% \pm 7.87$	16.01 ± 4.75	10.23 ± 3.76	$25.92\% \pm 5.11$
After Finetuning	$69.47\% \pm 7.65$	15.07 ± 4.85	09.67 ± 3.70	$26.50\% \pm 5.27$
PD				
Before Finetuning	$62.92\% \pm 15.72$	22.01 ± 15.93	15.50 ± 14.86	$27.45\% \pm 8.10$
After Finetuning	$63.54\% \pm 14.86$	19.95 ± 13.35	13.77 ± 12.72	$28.22\% \pm 8.14$
Overall Result	*+0.58 (+0.91%)*	*-1.78 (-8.65%)*	*-1.44 (-10.11%)*	*+0.72 (+2.67%)*

Table 4.12: Comparison of finetuning the model on a resection type for a 5-layer MSKNet

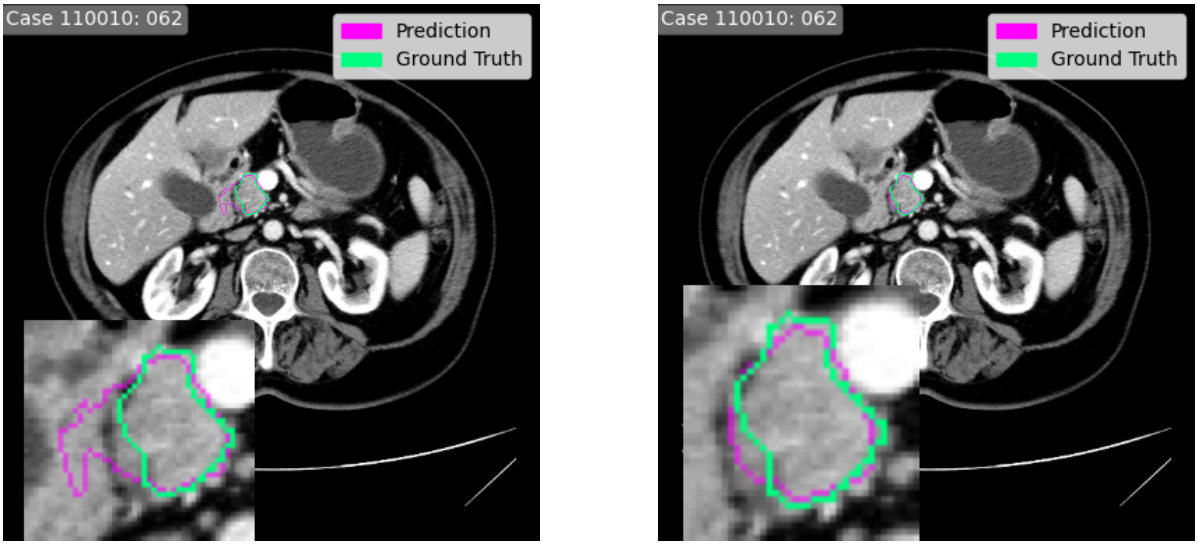
The performance improvements can be seen in Table 4.12 and Table 4.13. It was found that fine-

tuning the model is indeed beneficial. For MSKNet, both resection types show improvements in performance on all metrics. Surprisingly, the improvements for the PD resections are larger than for the distal ones when it comes to the HD and HD95. Because the complete dataset consists of 75% out of PD scans, one would expect the PD resections to benefit less than the distal ones. An explanation could be the very small dataset for the distal resection type, meaning there is less training data and more stochasticity in the final metrics. When it comes to the MAKNet, the opposite is true. Where the PD predictions barely score better, the distal ones show a major improvement on all measures. In the overall results, one can see that the relative performance increase for MSKNet is quite a bit larger than for MAKNet in both the HD as well as HD95 scores. An intuitive explanation is the differing number of parameters. The larger MSKNet contains more parameters that can be tweaked to the specific situation. Furthermore, the scores of MAKNet were already better, meaning that further improving the scores is more difficult.

Resection	DSC	HD (mm)	HD95 (mm)	NSD
Distal				
Before Finetuning	$66.91\% \pm 8.19$	15.86 ± 5.04	10.13 ± 04.19	$24.91\% \pm 4.02$
After Finetuning	$69.61\% \pm 7.97$	12.80 ± 3.49	08.07 ± 03.18	$25.81\% \pm 4.81$
PD				
Before Finetuning	$64.28\% \pm 15.55$	18.48 ± 11.81	12.38 ± 11.03	$27.55\% \pm 7.95$
After Finetuning	$64.28\% \pm 15.55$	18.13 ± 11.72	12.22 ± 11.06	$28.09\% \pm 8.14$
Overall Result	*+0.67 (1.02%)*	*-1.02 (-5.70%)*	*-0.63 (-5.32%)*	*+0.63 (+2.34%)*

Table 4.13: Comparison of finetuning the model on a resection type for a 4-layer MAKNet

As can be seen in Figure 4.4, the finetuning does improve the prediction visually as well. Whereas the non-finetuned 4-layer MAKNet model is not capable of correctly segmenting the pancreatic tissue, even though there is a clear change in voxel intensity, the finetuned model is capable of closely predicting the pancreas. In this distal case, the model might be influenced by the PD cases in the training set of the non-finetuned model. For a PD case, the model might expect a longer, more cylindrical shape, whereas the distal cases only include the head of the pancreas which is more of a spherical shape. This would also explain why the non-finetuned model adds additional voxels to the prediction, even though the voxel intensities are different.



(a) Before **(b)** After

Figure 4.4: Predictions of 4L MAKNet for a slice of the distal 110010 case before and after finetuning the model.

Baseline

Table 4.7 shows the performance of the untrained baseline models on a fraction of the dataset, but more interesting would be to see the results of trained baseline models on the postoperative data. In Table 4.14, all results for all state-of-the-art algorithms are reported with the best-performing models per metric in bold and the second-to-best-performing models in italics. All these results are without the finetuning per resection type. Even though Attention UNet is still performing slightly better when it comes to the DSC and NSD, MAKNet is by far the best on the HD and HD95 metrics, leading to assume it is far more shape-aware than the other models. The HD metrics in particular are the most important ones since these say more about clinical usability as the DSC and NSD are relatively close. With MSKNet following in second place for both HD metrics and MKNet in fourth place, it is safe to assume that the combination of MCBs with a KNet architecture can greatly increase the shape awareness of a segmentation model. Especially in the very variable abdominal composition of the postoperative state, MCB KNets show improvement over the current state-of-the-art.

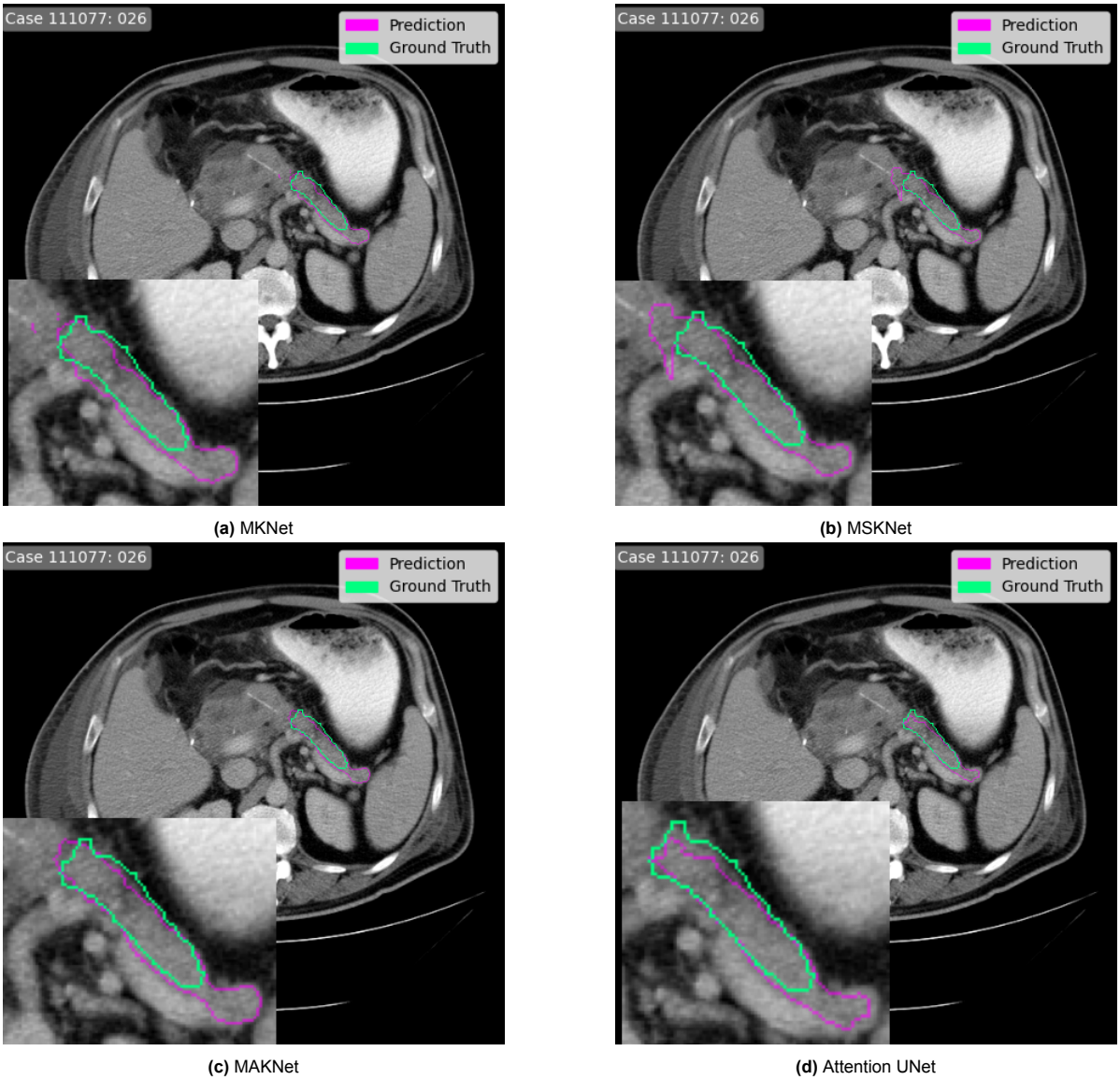
Method	# Model Params	DSC Post-Op	HD (mm)	HD95 (mm)	NSD
UNet [106, 56]	$9.5M$	$63.93\% \pm 16.09$	21.14 ± 14.96	14.93 ± 13.02	$26.38\% \pm 08.73$
Attention UNet [93]	$5.9M$	**65.98% ± 13.78**	19.69 ± 13.88	13.28 ± 12.67	**27.80% ± 08.40**
PanKNet$_{Light}$ [99]	$3.6M$	$62.94\% \pm 15.93$	22.12 ± 17.43	16.13 ± 16.49	$23.59\% \pm 07.42$
PanKNet [99]	$25.7M$	$59.73\% \pm 15.33$	20.42 ± 12.81	14.27 ± 11.87	$20.79\% \pm 07.06$
MUNet [72]	$26.7M$	$61.94\% \pm 16.83$	22.33 ± 16.99	16.09 ± 15.14	$25.95\% \pm 08.86$
5L MKNet (ours)	$37.8M$	$62.88\% \pm 15.36$	20.27 ± 12.79	13.78 ± 12.01	$25.57\% \pm 09.01$
5L MSKNet (ours)	$35.2M$	$64.42\% \pm 14.58$	*18.80 ± 11.83*	*12.68 ± 11.11*	*27.47% ± 8.06*
4L MAKNet (ours)	$15.1M$	*64.93% ± 14.76*	**17.33 ± 11.16**	**11.52 ± 10.24**	$27.15\% \pm 08.19$

Table 4.14: Comparison of the performance of the current state-of-the-art on postoperative scans

Figure 4.5 displays the prediction of one of the slices of case 111077 for all MKNet-family architectures and the current state-of-the-art Attention UNet. One can see that for this particular example, attention brings an increased accuracy when compared to MKNet and MSKNet, which both do not have attention modules. Looking at both MAKNet and Attention UNet, the predictions are rather similar. However, MAKNet seems to follow the ground truth a little closer. Both predictions have marked additional tissue as pancreatic in the lower right corner. In the slice below slice 26, the ground truth is indeed also covering this particular area. It seems that either the ground truth has not been annotated here with a high enough resolution, or that the rescaling during the preprocessing has caused the label to not be as high of a resolution. Either way, it seems that both models are actually more accurate than the ground truth on this specific slice for this particular case. If that would be the case, it can be seen that MAKNet follows the border of the tissue far more accurately than Attention UNet, which does not extend to the edge of the tissue. This would indicate a better performance from MAKNet and more accurate borders, which would most likely lead to a higher NSD in particular.

4.3. Qualitative Results

As argued in Section 3.4, quantitative metrics cannot say anything about the clinical usability of an algorithm as described in this research. In order to get a deeper understanding of the clinical performance, different experienced radiologists and radiotherapists have been asked to evaluate the algorithm's predictions, according to the methods described in Section 3.4. Since the availability of radiologists and radiotherapists is extremely limited, they have only been asked to evaluate the results of one of the algorithms. At the time the predictions needed to be sent to the experts, the 5-layer MSKNet with layer normalization, Mish activation, 3-3-3 kernel sizes, and MaxAvg downsampling module was the best-performing model, which is why that particular model was used for the qualitative evaluation. The model that predicted the scans was not finetuned per resection, but solely trained on all postoperative data. To retrieve the predictions from the model for every scan, the model that was trained on the other folds was used, i.e. all predictions were made on models that originally had the scan in question in the validation set. This is to ensure that the prediction is done by a model that has never seen the CT scan



(a) MKNet **(b)** MSKNet

(c) MAKNet **(d)** Attention UNet

Figure 4.5: Predictions of the MKNet-family compared to the current state-of-the-art Attention UNet for case 111077

before, to simulate a realistic scenario.

The results reported in this chapter are the results based on the answers of four experts, who have a combined experience of over 40 years being a radiologist. The reported results are based on the mode of each scan, i.e. the category that received the most “votes” from the experts, is the category it is classified as.

4.3.1. Necessary Adjustments

To determine the usability of the scans, the experts indicated for every scan if they would need to make no, minor, substantial, or major adjustments to the annotation if they would want to use it in a clinical application. The results of this can be seen in Figure 4.6. Interestingly enough, for almost a third of the annotations, the experts would not make any adjustments, indicating the prediction is close to perfect. In half of the cases, it only requires a minor adjustment and in just 2% of the cases, the prediction is useless. This means that in 83% of the cases, the algorithm was capable of finding a prediction that is immediately or almost immediately useful in the clinical application.

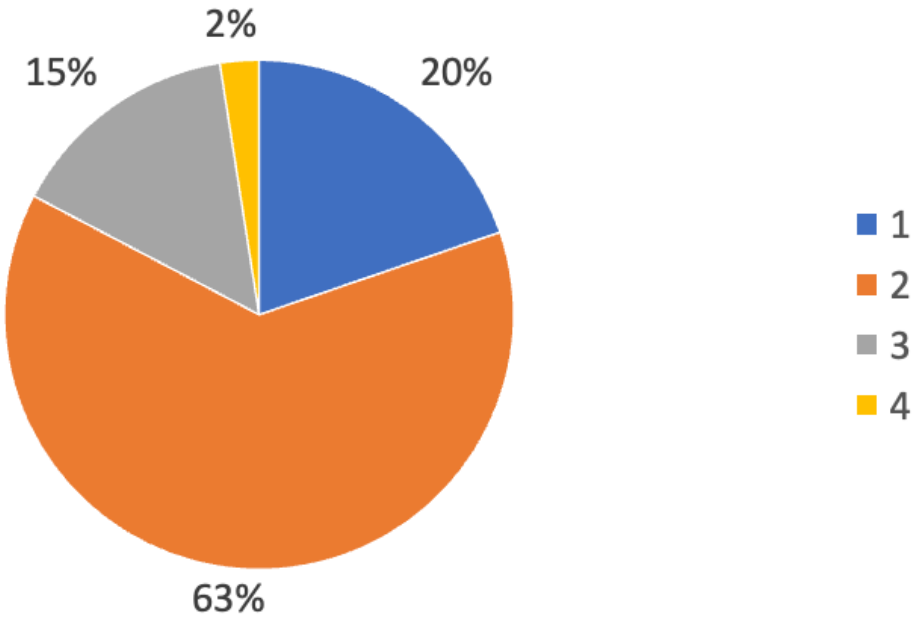


Figure 4.6: Adjustments necessary for clinical use: 1 - None; 2 - Minor; 3 - Substantial; 4 - Major

4.3.2. Future Use

In addition to the evaluation per scan, the experts were also asked how likely it was that they would use the algorithm again if they were to be asked to annotate postoperative pancreases on the following scale: very unlikely, unlikely, neutral, likely, and very likely. Three of the experts that evaluated the scans indicated that they are *likely* to use the algorithm again, whereas the fourth indicated they would be *very likely*. Comments such as *"a very strong basis"* and *"the pancreas is almost always correctly annotated, making it straightforward to annotate the pancreas quickly since little adjustments would be necessary"* showed they were impressed by the performance of the algorithm.

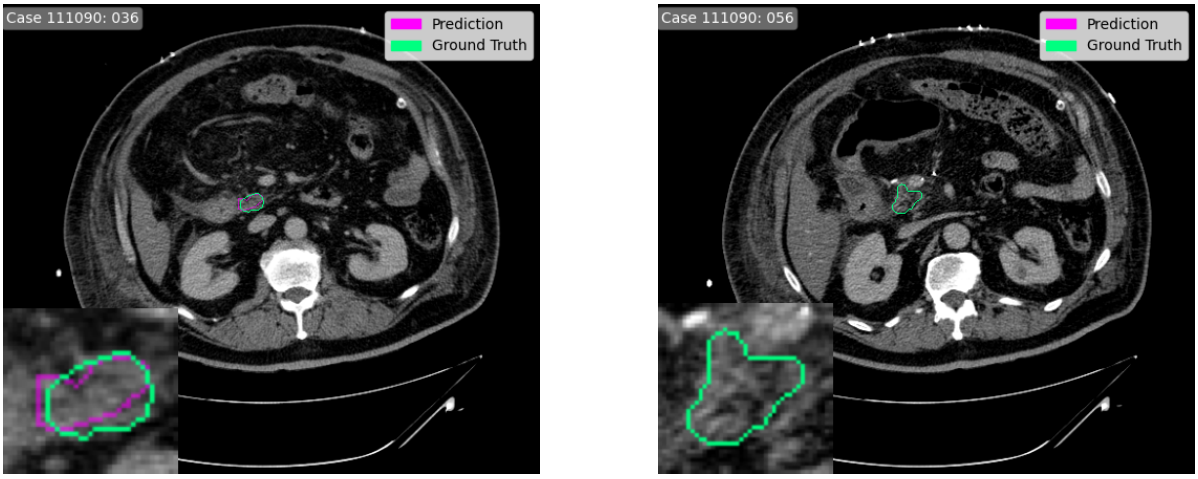
One of the radiologists provided a deeper analysis of the performance: *"the algorithm tends to underestimate the pancreatic parenchyma (functional pancreatic tissue, ed.), especially in the head. This underestimation is exacerbated when there is significant atrophy, fatty infiltration, or cystic areas. Hypodense regions are not properly included in the prediction. This is particularly evident in the pancreatic head. At the level of the head, the algorithm also struggles to differentiate pancreatic tissue from the duodenum, often resulting in underestimated tissue. It performs better in the tail. In the tail, it is common that the prediction includes a portion of the splenic artery or vein in the segmentation"*. It is interesting to see that the radiologist indicated that the model generally performed better on the PD scans, the ones where the head is removed, as the experiments reported in Section 4.2.3 indicated a better DSC, HD, and HD95 for the distal resection compared to the PD ones. It was only on the NSD that the PD resections scored better. The latter is in line with the deeper medical analysis, as a higher NSD indicates less under- and overestimation of the pancreatic tissue as a greater part of the surface voxels is equal to the ground truth. Even though a bigger part of the voxels are predicted correctly and the distance between all points of the predictions to all points of the ground truth, i.e. the Hausdorff distance, is smaller, the clinical performance can be lower, as the surfaces are too far apart.

4.3.3. Inter-Rater Variability

As already discussed before, the postoperative state is characterized by higher variability and increased difficulty in segmenting the pancreas. This is not just the case for an algorithm, but also for a radiologist. The pancreas is notoriously one of the most difficult organs to find on a CT scan, but even more so after a resection. Even in the evaluations of the experienced radiologists, there was a high variance in the categorization of the prediction. In categorizing the 81 scans, the radiologists were unanimously classifying the prediction in the same category in only 24 cases, i.e. only 24 cases got categorized equally by all four radiologists. This can partly be explained by the subjective nature of qualitative evaluation metrics, meaning one radiologist would consider the necessary adjustments minor whereas

another one would categorize them as substantial. However, 7.4% of the cases were categorized as category 1 by at least one radiologist, whereas at least one other radiologist categorized it as category 3. In other words, some say the prediction is perfect and requires no adjustments, whereas others say it requires major adjustments. For these cases, it is ruled out that subjectivity in the interpretation of the different categories is an explanation for the varying results and it shows the segmentation of the postoperative pancreas is a difficult task, even for experienced radiologists.

In the case of 111090, all four radiologists categorized it as a different category, i.e. there were no two radiologists that categorized it equally. In Figure 4.7 one can see two different slices with the prediction and the ground truth according to the experienced radiologist from the RACU that annotated the scan. As one can see, Figure 4.7a shows significant overlap in the prediction and ground truth, whereas Figure 4.7b shows the prediction does not reach all the way to the end of the ground truth. This would indicate a category 3 based on the ground truth. However, it cannot be said that the ground truth is indeed true, as there is little reason to assume that the ground truth is more likely to be correct than the categorizations of the other radiologists. Looking at the quantitative metrics, based on the ground truth as specified by the RACU radiologist, the scores are as follows: 30.29% (DSC), $31.45mm$ (HD), $25.73mm$ (HD95), and 7.93% (NSD). Since these quantitative metrics are based on the annotation of a single radiologist, and this example showed to be a challenging example to segment, it is difficult to draw any conclusions from this. The quantitative metrics do support the statement that the RACU radiologist would have probably categorized this case as a category 3.



(a) (b)

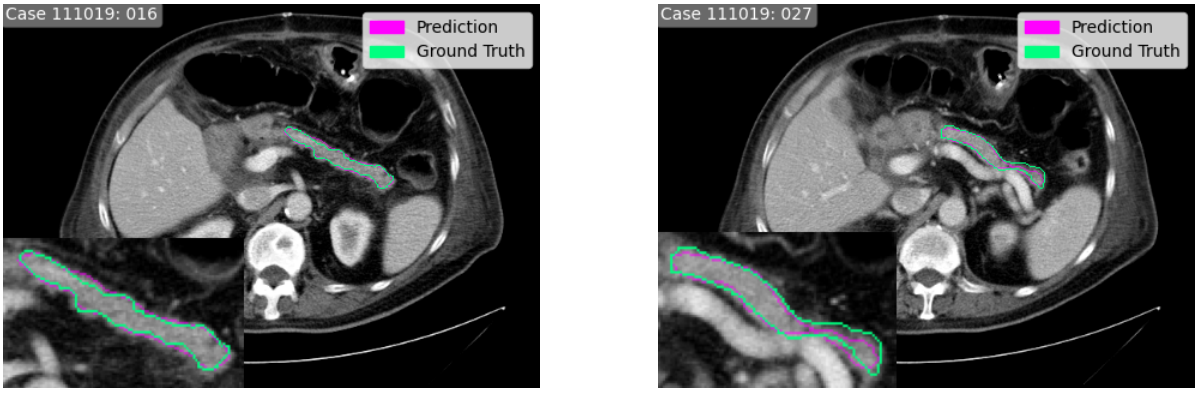
Figure 4.7: The prediction compared to the ground truth on two different slices for postoperative case 111090. In Figure 4.7b, the model did not see any pancreatic voxels so only the ground truth is shown.
DSC: 30.29%, HD: $31.45mm$, HD95: $25.73mm$, NSD: 7.93%

4.4. Combined Results

The reasoning for including qualitative evaluation metrics is based on the hypothesis that quantitative metrics do not necessarily give a correct representation of clinical usability. Since this research is in the unique situation of having both quantitative as well as qualitative metrics, a comparison between the scores on both metrics can be made.

One of the cases that was categorized by all medical experts as category 1, i.e. no adjustments are necessary, was case 111019. In Figure 4.8 one can see two of the slices including both the prediction of 5-layer MSKNet as well as the ground truth as annotated by a radiologist from RACU. As can already be seen, both are extremely close and seem to follow the organ's structure on the CT scan closely. This particular case achieved a DSC of 81.01%, HD of $5.66mm$, HD95 of $3.00mm$, and NSD of 40.78%. When it comes to the DSC, this case is scoring below average, even though it was unanimously chosen as a category 1 prediction. On the other hand, it is scoring exceptionally well on the HD, HD95, and NSD metrics. To put the HD metrics in perspective, the size of a single voxel is $1mm$ cubed, which means that the voxel that is the furthest away from the ground truth is only 3 voxels away if the 5^{th} percentile outliers are ignored. The high NSD corresponds with the image as seen in Figure 4.8, as the

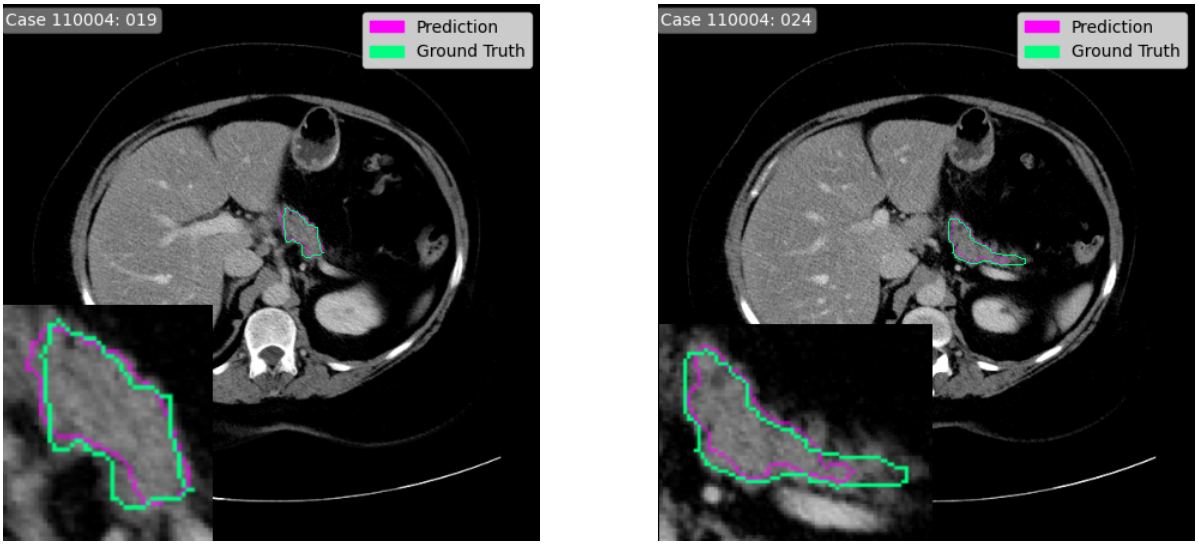
two circumferences are extremely close. Looking at this particular scenario, therefore, the HD, HD95, and NSD metrics are far more informative when it comes to clinical performance than the DSC.



(a) (b)

Figure 4.8: The prediction compared to the ground truth on two different slices for postoperative case 111019. DSC: 81.01%, HD: $5.66mm$, HD95: $3.00mm$, NSD: 40.78%

The 110004 case was unanimously categorized as a category 2. In Figure 4.9 one can see two of the slices of the specific case, where it can be seen that the prediction indeed roughly covers the same region, but differs slightly on the edges. In Figure 4.9b one can see that the ground truth stretches further out to the right, whereas the prediction stops slightly earlier not covering the full pancreas. The prediction had the following scores on the quantitative measures: 69.77% (DSC), $23.28mm$ (HD), $14.35mm$ (HD95), 33.40% (NSD). As one would expect the difference between the HD and HD95 scores is relatively large, which indeed indicates that the prediction does not cover all edges of the pancreas. The DSC and NSD are well above average, indicating that many voxels have been correctly classified, including a third of the surface voxels. It is, however, the shape that differs which explains the higher HD and HD95.

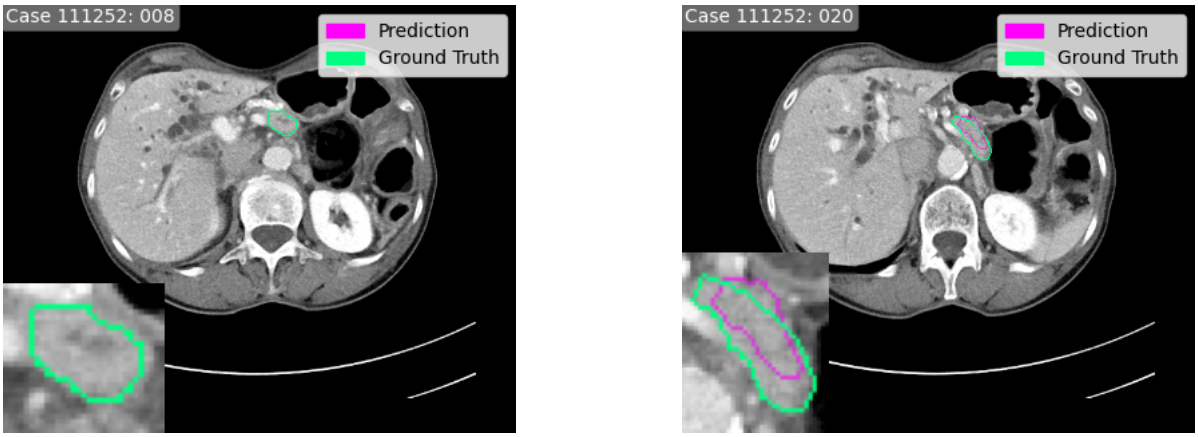


(a) (b)

Figure 4.9: The prediction compared to the ground truth on two different slices for postoperative case 110004. DSC: 69.77%, HD: $23.28mm$, HD95: $14.35mm$, NSD: 33.40%

One of the cases that has been categorized as category 3 by all radiologists was 11252 of which two slices can be seen in Figure 4.10. For this case, the model is capable of finding the correct tissue, but it is somewhat too conservative in its prediction, i.e. the ground truth is a significantly larger body than the prediction. In one of the lower slices in Figure 4.10a, the pancreas is already present, indicated by the green circumference, whereas the prediction only starts several slices later. In Figure 4.10b one

can also see the prediction is smaller than the ground truth, although it does consist fully of pancreatic voxels. The quantitative performance of this case is as follows: 8.94% (DSC), $89.37mm$ (HD), $82.37mm$ (HD95), 4.01% (NSD). As the voxels of the prediction consist almost completely of the inner voxels of the ground truth, it is to be expected that the score on the NSD is extremely low as the surface voxels of both volumes barely overlap. The low DSC can be explained similarly, as the ground truth simply consists of a significant amount of more voxels. Since the DSC is measured relatively between two volumes, it is low if many voxels of one volume are not included in the other. One could expect the HD and HD95 to be extremely low as the prediction consists of almost only pancreatic voxels. However, the HD is defined by $\max(h(A,B), h(B,A))$, so it measures the maximum distance from one body to another and vice versa. The surface voxels of the ground truth are far away from the surface voxels of the prediction. Since the HD and HD95 are defined by the maximum distance from the ground truth to the prediction and this distance is very large, since the ground truth stretches over a significant amount of more slices, the scores on both metrics must be low as well.



(a) (b)

Figure 4.10: The prediction compared to the ground truth on two different slices for postoperative case 111252. In Figure 4.10a, the model did not see any pancreatic voxels so only the ground truth is shown.
DSC: 8.94%, HD: $89.37mm$, HD95: $82.37mm$, NSD: 4.01%

Lastly, the 110002 case was specified unanimously as a category 4. In Figure 4.11 one can see the reason for this, as the prediction is in a completely different region of the abdomen. It does not surprise that both the DSC as well as the NSD are 0.00%, whereas the HD and HD95 are $67.01mm$ and 62.34 respectively. The fact that the HD and HD95 are still better than, for example, in the case of 11252, can be explained by the fact that the volumes happen to be close resulting in a lower HD and HD95, but naturally, this does not give a good indication of the true performance.

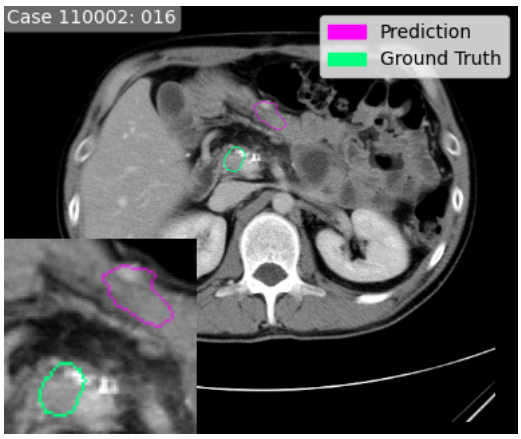
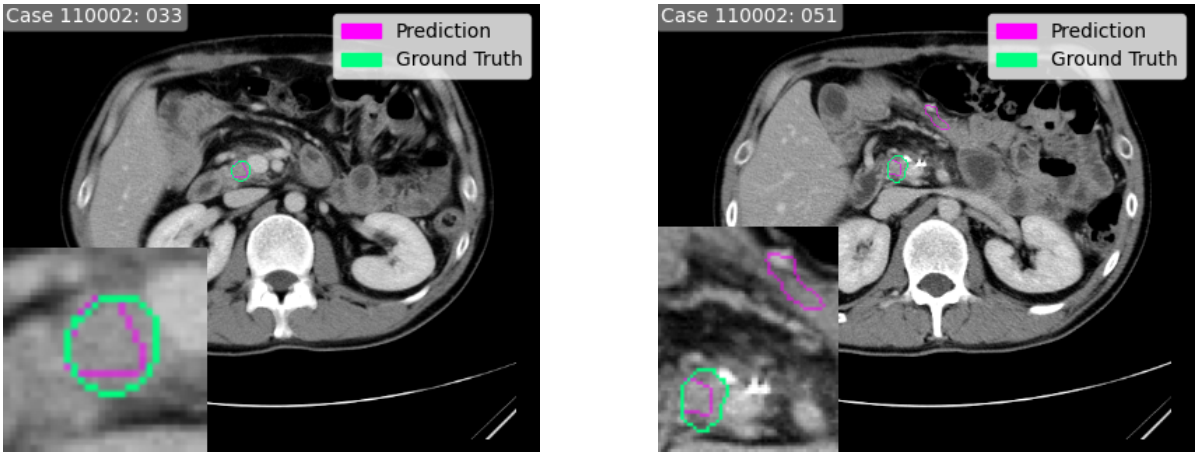


Figure 4.11: The prediction compared to the ground truth for postoperative case 110002.
DSC: 0.00%, HD: $67.01mm$, HD95: $62.34mm$, NSD: 0.00%

Interestingly for this specific case, is the fact that the postprocessing step of the KLC as earlier described in Section 3.1 is not beneficial. The DSC before KLC was 35.65%. Even though this is not a good score, it does show the impact of the KLC operation on badly predicted pancreases. If a wrongly predicted volume is larger than a correctly predicted volume, all pancreatic voxels will get lost in the prediction. In Figure 4.12 one can see the prediction before KLC is applied. It can be seen that there is indeed a prediction of the pancreas in the same volume as the ground truth, but those pancreatic voxels are removed from the prediction as the connected component in the pancreatic region is smaller than the wrongly predicted connected component. From a clinical point of view, it would potentially be better to retain from applying the KLC as it is trivial for a radiologist that there is only one pancreas.



(a) (b)

Figure 4.12: The prediction compared to the ground truth on two different slices for postoperative case 110002, without the KLC postprocessing operation

4.4.1. Conclusion

Although each individual quantitative metric can say something about the qualitative performance, it is only the combination of metrics that gives a good impression. A mediocre DSC with good HD, HD95 and NSD, can already indicate a clinically perfect prediction. On the other hand, an HD of $89.37mm$ does not necessarily mean the prediction is better than one with an HD of $67.01mm$. Only when one can see that the DSC and NSD for the latter are 0.00%, it becomes clear this prediction is worse. All in all, the combined quantitative metrics can give a relatively good indication of the qualitative score. However, the individual quantitative metrics can not, which leads to another example of the importance of using multiple quantitative metrics when evaluating a model.

5 Discussion

The main goal of this research was to find answers to the question: *To what extent can the pancreatic remnant be accurately segmented on CT scan images of patients that underwent pancreatic resection, leveraging the potential of deep learning?*. In finding these answers many choices were made that could have been better in hindsight. Furthermore, there were a significant amount of insights during the research that fell outside the scope of this research but could be interesting to investigate further. In the following chapter, the results and methods of this research are discussed, followed by a description of the practical applications in the medical domain of this research. Lastly, some limitations of this work and possibilities for future work are highlighted after which a short conclusion and highlight of the contributions are given.

5.1. Results

During this research, as described in Chapter 3, the most focus was given to the top two performing algorithms at the time of writing, based on the DSC that was reported in the papers respectively. Even though the characteristics of these algorithms can theoretically boost the performance on postoperative data, the reported performance on preoperative data could not be reproduced, despite the fact that the code of Proietto Salanitri et al. [99] was publicly available. Hypotheses as to what could explain this difference in performance are already given in Section 4.2.1, but it is unlikely these hypotheses account for the full performance gap. Another explanation for the performance gap could be a different computation of the DSC. Particularly in the case of 2D networks, such as the one in Ma, Zou, and Liu [72], it matters how the data is divided into the four folds and how the DSCs of the individual slices are combined into one final performance score. As already identified by Maier-Hein et al. [76], averaging over all slices can result in a significantly different score than averaging over the slices of one CT scan and consecutively averaging over all CT scans. The first approach will lead to a significantly higher score. In addition to that, when for example using the *DiceMetric* implementation of MONAI, the DSC can be computed either over both the foreground as well as the background prediction channels or just over the foreground channel. Naturally, this gives different scores. The difficulty with comparing the scores between the different papers is that it is extremely rarely mentioned how exactly the scores are computed. This makes it impossible to state that this would explain the difference in performance, even though it could indeed be a plausible explanation.

Looking at the results on the preoperative data compared to the postoperative data, the hypotheses made in Section 3.2 about the MCB and KNet variants are supported. Both the different decoders as well as the multi-scale convolutions aim to retrieve different types of information from the data and store that in the network. This seems to pay off in both the preoperative as well as postoperative state resulting in better HDs, and HD95s. During a pancreatic resection, different parts of the pancreas can be removed and the overall composition of the abdomen varies greatly from patient to patient. In the preoperative state, the pancreas differs slightly in shape and location, but in the postoperative state, it is even more difficult to segment pancreatic tissue. This requires a great amount of flexibility in the model, which seems to be provided by the MCBs and KNet architecture which show great shape awareness in the predictions.

Even though the scores on the quantitative evaluation metrics were quite a bit worse for the post-operative state, the qualitative metrics showed a great clinical performance. With 83% of the cases requiring little to no adjustments, according to radiologists, the algorithm is capable of predicting most of the cases very accurately. It seems the combination of the different quantitative metrics reported in this research can give a good indication of the qualitative performance as reported by the medical experts. The cases that had both relatively high scores on the DSC and NSD, and relatively low scores on the HD and HD95, also scored well on the qualitative evaluation.

Something which became apparent during further visual analysis of the predictions is that the ground truth does not always continuously flow through the slices, i.e. the circumference is equal for several consecutive slices after which it "jumps" to a rather differently shaped area. This is due to the resolution of the CT scans and the preprocessing steps. The slice thickness can be up to four millimeters for some scans, whereas the preprocessing steps ensure that every slice is only one millimeter thick. As already described in Section 3.1, the label is reshaped using nearest neighbor instead of trilinear interpolation to the example of Proietto Salanitri et al. [99] in order to have discrete values for the ground truth voxels. The nearest neighbor method in combination with a large slice thickness, however, makes the ground truth in the intermediate slices rather off. This results both in less accurate training as well as lower measured performance on those slices. Even though, hypothetically, the prediction is more accurate than the ground truth on those specific slices.

5.2. Methodology

In order to come to the best-performing postoperative segmentation model, the research setup was as follows. Firstly, a good overview of the current state-of-the-art for preoperative pancreas segmentation needed to be created. When this was available, it became clear what the most promising techniques were and what techniques could theoretically be beneficial in implementing a postoperative segmentation model. In this case, the PanKNet and MUNet models showed the most promising results, and combining both could theoretically improve accuracy significantly for postoperative data specifically, because of the even higher variability of the data. However, trying to reproduce the reported dice scores eventually consumed a significant amount of time, especially since experiments run for a long time and computational capacity is scarce. In hindsight, it would have been wiser to create working prototypes of multiple algorithms in a short amount of time, after which a combination of multiple, or simply the most promising one could have been finetuned.

As the author has a computer science background and no previous experience with the medical domain, there were some difficulties with for example the qualitative evaluation method. Since the DICOM format is specifically designed for medical applications, it is not straightforward to transfer from the Tensor predictions in Python back to the DICOM format again. Consequently, a different approach had to be taken to let the radiologists and radiotherapists evaluate the predictions. The initial format in which the predictions were sent to the experts was not experienced convenient, resulting in additional work for both the expert as well as for the author. Although it would have been difficult to prevent this upfront, more care could have been taken in the preparation of the qualitative evaluation.

One of the most important things in training an ANN is the quality of the data and in particular the quality of the ground truth. The process of creating the ground truth for the RACU data has been designed and executed with the greatest care. However, all scans are verified by only a single radiologist. Even though in most cases this is not an issue, the results reported in Section 4.3.3 show that even for an experienced radiologist it can be difficult to exactly point out where the pancreas is located. Since the ground truth is only verified by a single radiologist, there is a chance some annotations are not completely correct. The model then learns incorrect patterns, which harms the overall performance of the model. To mitigate this, the ground truth can be verified by multiple experienced radiologists, but that was outside of the scope of this research.

The NSD metric requires threshold values to determine the border region in which a surface voxel should be classified as correct. The values for the X-, Y- and Z-axis are set to three mm, three mm, and two mm respectively in this research. Even though these numbers are based on experiences with the dataset and results from experiments, no argument is given from a qualitative point of view why these values are optimal. On the contrary, the optimal values are likely to be different. Although it was outside of the scope of this research to determine the best threshold values, further research could go further into finding the most representative values that best agree with the qualitative evaluations.

5.3. Practical Application

When starting this research, the broader view was that this algorithm will be used in a larger pipeline that can recommend medical doctors as to whether or not a patient that has had a pancreatic resection due to pancreatic cancer suffers from recurrent pancreatic cancer. Although this is still the main purpose after finishing the research, an additional benefit popped up. Checking annotations is significantly less time-consuming than actually annotating a scan. Particularly for less experienced annotators, it can easily take an hour to annotate about thirteen scans, whereas checking if a scan is annotated correctly only takes a fraction of that time. This means that if an experienced radiologist only has to check annotations, instead of having to create all of them from scratch, great time savings can be achieved by using this algorithm, i.e. radiologists could use the algorithm to do a lot of the preparatory work. This is particularly useful for educational purposes, where radiologists in training can use automatically segmented scans to better understand the abdominal region after pancreatic resection.

The practical use of this research is further shown by the qualitative evaluation as the experts are (very) likely to use the algorithm again to help in annotating postoperative pancreases. With a fifth of the cases not even requiring any adjustments and 63% of the cases only minor adjustments, the annotation of data can be greatly accelerated. This means the creation of more data can more cheaply and easily be done. In a field where data is extremely scarce like in the (postoperative) pancreas segmentation, this can have a huge impact on future research. Furthermore, it can easily be used for educational purposes already since in most cases it is correct and in some cases leaves work to do for the student.

5.4. Limitations and Future Work

Hypothetically, one of the most important reasons for the lack of research on the segmentation of the pancreas in the postoperative state is the fact that there is no publicly available data. Publishing medical data can be difficult for hospitals, both for legal reasons as well as for economic reasons. During this research a collaboration with the RACU was made, making it possible to combine forces in both the medical domain as well as the computer science domain. This also includes the availability of data from the RACU and man-hours from people from the RACU to annotate that data. However, this is also a limitation of this particular research. At the time of writing, the dataset used in this research for all postoperative experiments is not publicly available. This means that the model in this research cannot be compared to other work on the postoperative state, simply because the data will be different or missing. To mitigate this to a certain extent, evaluation metrics were reported on the preoperative data that is publicly available. However, as can be seen in Section 4.2.2, the results on preoperative and postoperative data can vary significantly. To further mitigate this limitation, qualitative metrics were added. Although these can be subjective, they do allow for an evaluation that can be compared to future postoperative models on different data.

One of the difficulties in deep learning research, particularly in the medical domain, is the size and quality of the dataset. There is only a limited amount of high-quality annotated CT scans publicly available of which the distribution can impossibly match the underlying distribution, i.e. a dataset of 82 CT scans does not have the same distribution of a hypothetical dataset consisting of CT scans from all humans on this planet. This means that the model, although it can perform well on examples in the dataset on which it is trained, does not necessarily perform well on new examples. Usually, in the field of deep learning, this is mitigated by adding more examples to the training data, but this is not an option since there is no more data available. Therefore, it is impossible to say how well the reported evaluation metrics in this thesis transfer to new examples beyond the used datasets.

As already discussed in Section 4.4, the pipeline makes use of the KLC postprocessing operation. This ensures that potential noise in the prediction is removed, keeping only the largest connected component. However, it became apparent that for difficult cases, the largest connected component is not necessarily the connected component with the highest number of correctly classified voxels. It can occur that the connected component with the highest number of pancreatic voxels is not the largest, drastically reducing both the quantitative as well as qualitative scores. Future research could dive deeper into the postprocessing of the prediction, perhaps selecting the component with the highest certainty or some different heuristic.

Another limitation is in the preprocessing of the data. If the full CT volume were to be used, the training process would be too memory intensive on both the GPU as well as the RAM. In order to reduce the memory footprint, only the slices which include the pancreas were selected with a margin of

10 on both sides. This also means the algorithm has had input data that is not completely representative of a regular application where the full CT volume would be used as input. In order to crop a CT scan into a similar representation as used in the training, one needs to know where the pancreas is situated which is not the case for a new non-annotated scan. Future research could dive deeper into the impact of this preprocessing step on the performance, and potentially on a relatively cheap and fast way to automatically create such a bounding box.

The practical application of this research could be a bit cumbersome due to the time it takes to train the algorithm. With the preprocessing step generally taking up to 10-15 hours to train and the postoperative training some additional 3-5 hours, the total time of training is quite substantial. Although inference can happen relatively fast, in the order of 10-20 seconds per scan, the long training time might make future research more difficult. To improve the time performance, future work could conduct research in the application of the same techniques used in the MKNet family of architectures but then in 2D or 2.5D. It is expected that the achieved evaluation metrics scores would decrease, but it might be worth the trade-off if it is significantly faster.

Although a 2D variant of the MCB was already published by Ma, Zou, and Liu [72], it is still a new technique, particularly in 3D. In the current research, some variations of the MCB have been experimented with, but it does not come close to the vast possibilities of this convolutional block. The number of branches within an MCB could have a great impact on the performance of it. In particular in 2D where the cost of adding an additional branch is lower, it could be beneficial to have a different number of branches. More research could also be conducted into ways to ensure the branches retrieve different kinds of information, e.g. by having a set kernel size for the convolutions in every branch instead of the n^{th} convolution in each branch.

The work of Man et al. [77] introduced deformable convolutions in a UNet architecture. There is a good possibility this could work well in combination with the techniques introduced in this research, e.g. a branch with regular convolutions and a branch with deformable ones. Although there have been some efforts into implementing deformable convolutions in the MKNet architecture, this has not matured enough to come to tangible experiments. The same holds for spectral pooling or DiffStride [104], which shows promising results when it comes to maintaining structure information during downsampling. These alternatives could improve the performance even further.

Lastly, future work in the decoder part of the network could be useful. In this research, a great amount of effort has been put into improving the encoder, since the multiple decoders were hypothesized to already lead to an increased retrieval of information. However, the decoders are almost completely the same as the decoder part as proposed by Ronneberger, Fischer, and Brox [106] eight years ago. Future work could focus on implementing new techniques in the decoder part of the network in order to further enhance the predictions.

5.5. Contributions

The main contribution of this research is in the segmentation of the pancreas after pancreatic resection. To the best of the author's knowledge, there has not been any prior research into this area. With this research, the next step into an earlier detection of the recurrence of pancreatic cancer is taken. Not only by providing part of a final pipeline necessary to assist doctors to make a correct diagnosis but also by easing the annotation of more valuable data in this field of research.

When looking at Table 2.1, it can be seen that for most algorithms only the DSC score is reported. Sometimes not even with an accompanying standard deviation. This supports the proposition that in current research into the segmentation of the pancreas, it is mostly only the DSC score that is used for the evaluation of a model. Although the DSC is a proper measure for the performance of the model, it does not contain information on, for example, the shape of the prediction. In other words, a prediction can have a very high DSC but not look like a pancreas. Since algorithms like these are designed to eventually be implemented in a clinical environment, it is incredibly important that the algorithm in fact is useful in the clinical environment. This research is one of the few in many years that reports scores on multiple evaluation metrics, displaying several characteristics of the model. As could be seen in Table 4.1, Wang et al. [129] were the only in the current state of the art to report an additional metric, namely the HD, but did not report any other evaluation metrics such as the HD95 or NSD. The fact that this research reports multiple metrics with a different focus, i.e. border- and shape-aware metrics, can help nudge future research in using multiple metrics as well and allows one to understand the true

performance of this model better.

In addition to the multiple quantitative evaluation metrics, this research is, to the best of the knowledge of the author at the time of writing, the first that describes a method for evaluating pancreatic segmentation results in a qualitative manner. As already described, it is of great importance to create a model that is performing well in a clinical environment as well as to create a model of which the performance can be explained. The most apparent way to measure this is by asking the experts that would eventually need to use the model, i.e. (experienced) radiologists. Although the described method in Section 3.4 is still quite subjective, as inherent to qualitative measures, it does allow for quantifying qualitative results.

Even though this algorithm was specifically designed for the pancreas in the postoperative state, it essentially tries to solve the problem of finding a small, variable object within a large volume. This means that it has great potential to also perform well in other instances of this problem. This could be finding other small, variable organs on CT scans, but could potentially also transfer to other domains, e.g. small objects in high-resolution drone images or small objects in the camera view of autonomous cars.

At the start of this research, it became clear that there has been conducted more and more research into the segmentation of the pancreas in the last couple of years. However, it immediately stands out in the comparisons with other state-of-the-art algorithms in these papers, it often lacks most of the best-performing algorithms. The generic overview of the state-of-the-art semantic segmentation algorithms for the pancreas simply is clouded. In the literature study of this research, significant attention is paid to including all published best-performing algorithms up to and including 2022. In Section 2.5, a clear overview is given, which helps future research to understand what the current research state is, what has been tried, and what could be promising to elaborate on. This can potentially kickstart future research and speed up future development in this incredibly important field of research.

6

Conclusion

The aim of this research was to investigate the feasibility of segmenting the pancreatic remnant after pancreatic resection using deep learning methods. The study involved examining state-of-the-art segmentation models for the preoperative pancreas and integrating them with new techniques into a final postoperative segmentation pipeline that included pretraining on preoperative data.

The results demonstrated that the pancreas can be accurately segmented to a certain extent using a deep learning model. It was evident that the postoperative state posed greater challenges compared to the preoperative state. However, by employing advanced techniques like the KNet architecture [99] and MCBs [72], the performance of the segmentation models improved significantly. The newly designed architectures incorporating these techniques already achieved state-of-the-art results for the preoperative state. Notably, MKNet achieved the best recorded Hausdorff Distance ($13.64mm$) and 95th percentile Hausdorff Distance ($5.89mm$) ever reported on the NIH dataset [107]. In the postoperative state, MAKNet outperformed the current state-of-the-art as well with Hausdorff Distance and 95th percentile Hausdorff Distance scores of $17.33mm$ and $11.52mm$, respectively. Additionally, the models demonstrated increased robustness with lower variances.

In addition to comprehensive quantitative evaluation metrics, this research also incorporated a qualitative assessment, which, to the best of the author's knowledge, was the first-ever conducted for pancreatic segmentation. The qualitative methods evaluated the algorithm's performance in a clinical context, where it could potentially be implemented. In 83% of the cases, the algorithm accurately segmented the pancreas to an extent that required minimal or no modifications according to medical experts. Only 2% of the cases were deemed unsatisfactory. Consequently, the experts expressed willingness to utilize the algorithm for pancreas annotation.

This research made significant contributions by being the first study, to the best of the author's knowledge, to focus on the segmentation of the postoperative pancreas. It also introduced one of the most comprehensive quantitative evaluations of the last decade, encompassing both newly designed algorithms and existing state-of-the-art methods. Moreover, this research pioneered the evaluation of qualitative performance in a clinical setting with experienced radiologists and radiotherapists. The variability in the evaluations of the different medical experts further strengthens the motivation to continue research into the automatic segmentation of the pancreatic remnant.

Inherent to research, there are some limitations to this study. Since there is only a little amount of data available, it cannot be proven all reported results can be generalized to a larger dataset. Although great care was taken to make the results as representative as possible, it cannot be ensured the performance will be equal on a larger dataset.

Future work should explore the generalizability of the model to address the aforementioned limitation. Additionally, further experimentation with the MCB is warranted. Different designs, such as varying kernel sizes per branch or the number of branches within one MCB, as well as different types of convolutions within the branches, hold potential for investigation. The decoder part of the network also requires attention, as the design has remained largely unchanged from the decoder used in the original UNet architecture by Ronneberger, Fischer, and Brox [106] eight years ago. Exploring layers with enhanced information extraction capabilities from the feature maps could yield valuable insights.

In the long term, future research can go into more depth on the segmentation and detection of fibrosis and tumorous tissue in the postoperative state. This could, in combination with the segmentation of the pancreatic remnant, provide a solid foundation for a medical expert to argue whether or not surgery was successful and if recurrence is likely.

Despite its exploratory nature, this research holds practical implications. Data scarcity and the associated costs pose significant challenges in medical research, particularly in pancreas segmentation. While the algorithm's predictions may not be flawless, in 20% of cases experienced radiologists would make no changes. This indicates the potential of the algorithm to expedite data annotation and accelerate future research in this field. The experts' expressed willingness to reuse the algorithm further supports its practical utility.

In conclusion, this research demonstrates the efficacy of MKNet family architectures in segmenting the pancreas, both in preoperative and postoperative states, as evidenced by extensive quantitative and qualitative evaluations. It constitutes the first-ever study on postoperative pancreas segmentation and has been positively evaluated by medical experts for its usefulness and performance, with the added potential to reduce data creation costs and expedite future research endeavors.

References

[1] Horst Aichinger et al. *Radiation Exposure and Image Quality in X-Ray Diagnostic Radiology: Physical Principles and Clinical Applications*. en. Google-Books-ID: nPisjRy4LNAC. Springer Science & Business Media, Oct. 2011. ISBN: 978-3-642-11241-6.

[2] Eric Alcaide. *E-swish: Adjusting Activations to Different Network Depths*. Tech. rep. arXiv:1801.07145. arXiv:1801.07145 [cs, stat] type: article. arXiv, Jan. 2018. URL: http://arxiv.org/abs/1801.07145 (visited on 11/02/2022).

[3] Md Zahangir Alom et al. *Recurrent Residual Convolutional Neural Network based on U-Net (R2U-Net) for Medical Image Segmentation*. Tech. rep. arXiv:1802.06955. arXiv:1802.06955 [cs] type: article. arXiv, May 2018. URL: http://arxiv.org/abs/1802.06955 (visited on 11/14/2022).

[4] Andrea Apicella et al. "A survey on modern trainable activation functions". In: *Neural Networks* 138 (June 2021). arXiv:2005.00817 [cs, stat], pp. 14–32. ISSN: 08936080. DOI: 10.1016/j.neunet.2021.01.026. URL: http://arxiv.org/abs/2005.00817 (visited on 11/02/2022).

[5] Sylvain Arlot and Alain Celisse. "A survey of cross-validation procedures for model selection". In: *Statistics Surveys* 4.none (Jan. 2010). arXiv:0907.4728 [math, stat]. ISSN: 1935-7516. DOI: 10.1214/09-SS054. URL: http://arxiv.org/abs/0907.4728 (visited on 05/01/2023).

[6] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. *Layer Normalization*. Tech. rep. arXiv:1607.06450. arXiv:1607.06450 [cs, stat] version: 1 type: article. arXiv, July 2016. URL: http://arxiv.org/abs/1607.06450 (visited on 10/28/2022).

[7] Andrew L. Beam and Isaac S. Kohane. "Translating Artificial Intelligence Into Clinical Care". In: *JAMA* 316.22 (Dec. 2016), pp. 2368–2369. ISSN: 0098-7484. DOI: 10.1001/jama.2016.17217. URL: https://doi.org/10.1001/jama.2016.17217 (visited on 10/03/2022).

[8] Yoshua Bengio, Ian Goodfellow, and Aaron Courville. *Deep learning*. Vol. 1. MIT press Cambridge, MA, USA, 2017.

[9] Francesco Bianconi et al. "Comparative evaluation of conventional and deep learning methods for semi-automated segmentation of pulmonary nodules on CT". In: *Quantitative Imaging in Medicine and Surgery* 11.7 (July 2021), pp. 3286–3305. ISSN: 2223-4292. DOI: 10.21037/qims-20-1356. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8250017/ (visited on 09/28/2022).

[10] Gianluca Brugnara et al. "Automated volumetric assessment with artificial neural networks might enable a more accurate assessment of disease burden in patients with multiple sclerosis". eng. In: *European Radiology* 30.4 (Apr. 2020), pp. 2356–2364. ISSN: 1432-1084. DOI: 10.1007/s00330-019-06593-y.

[11] Delft High Performance Computing Centre (DHPC). *DelftBlue Supercomputer (Phase 1)*. 2022. URL: https://www.tudelft.nl/dhpc/ark:/44463/DelftBluePhase1.

[12] Jianxu Chen et al. "Combining Fully Convolutional and Recurrent Neural Networks for 3D Biomedical Image Segmentation". In: *Advances in Neural Information Processing Systems*. Vol. 29. Curran Associates, Inc., 2016. URL: https://proceedings.neurips.cc/paper/2016/hash/4dcf435435894a4d0972046fc566af76-Abstract.html (visited on 09/07/2022).

[13] Jieneng Chen et al. *TransUNet: Transformers Make Strong Encoders for Medical Image Segmentation*. Tech. rep. arXiv:2102.04306. arXiv:2102.04306 [cs] type: article. arXiv, Feb. 2021. URL: http://arxiv.org/abs/2102.04306 (visited on 11/14/2022).

[14] Liang-Chieh Chen et al. “DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 40.4 (Apr. 2018). Conference Name: IEEE Transactions on Pattern Analysis and Machine Intelligence, pp. 834–848. ISSN: 1939-3539. DOI: 10.1109/TPAMI.2017.2699184.

[15] Liang-Chieh Chen et al. “Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation”. en. In: *Computer Vision – ECCV 2018*. Ed. by Vittorio Ferrari et al. Vol. 11211. Series Title: Lecture Notes in Computer Science. Cham: Springer International Publishing, 2018, pp. 833–851. ISBN: 978-3-030-01233-5 978-3-030-01234-2. DOI: 10.1007/978-3-030-01234-2_49. URL: https://link.springer.com/10.1007/978-3-030-01234-2_49 (visited on 10/12/2022).

[16] Lifang Chen and Li Wan. “CTUNet: automatic pancreas segmentation using a channel-wise transformer and 3D U-Net”. en. In: *The Visual Computer* (Sept. 2022). ISSN: 1432-2315. DOI: 10.1007/s00371-022-02656-2. URL: https://doi.org/10.1007/s00371-022-02656-2 (visited on 11/28/2022).

[17] Qishang Cheng et al. “Parametric Deformable Exponential Linear Units for deep neural networks”. en. In: *Neural Networks* 125 (May 2020), pp. 281–289. ISSN: 0893-6080. DOI: 10.1016/j.neunet.2020.02.012. URL: https://www.sciencedirect.com/science/article/pii/S0893608020300575 (visited on 11/02/2022).

[18] Necip Cinar, Alper Ozcan, and Mehmet Kaya. “A hybrid DenseNet121-UNet model for brain tumor segmentation from MR Images”. en. In: *Biomedical Signal Processing and Control* 76 (July 2022), p. 103647. ISSN: 1746-8094. DOI: 10.1016/j.bspc.2022.103647. URL: https://www.sciencedirect.com/science/article/pii/S1746809422001690 (visited on 05/03/2023).

[19] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. *Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs)*. Tech. rep. arXiv:1511.07289. arXiv:1511.07289 [cs] type: article. arXiv, Feb. 2016. DOI: 10.48550/arXiv.1511.07289. URL: http://arxiv.org/abs/1511.07289 (visited on 11/02/2022).

[20] MONAI Consortium. *MONAI: Medical Open Network for AI*. June 2023. DOI: 10.5281/zenodo.8018287. URL: https://zenodo.org/record/8018287 (visited on 06/18/2023).

[21] Marius Cordts et al. *The Cityscapes Dataset for Semantic Urban Scene Understanding*. Tech. rep. arXiv:1604.01685. arXiv:1604.01685 [cs] type: article. arXiv, Apr. 2016. DOI: 10.48550/arXiv.1604.01685. URL: http://arxiv.org/abs/1604.01685 (visited on 06/18/2023).

[22] Tami D. DenOtter and Johanna Schubert. *Hounsfield Unit*. 2022. URL: http://europepmc.org/books/NBK547721.

[23] Vijaypal Singh Dhaka et al. “A Survey of Deep Convolutional Neural Networks Applied for Prediction of Plant Leaf Diseases”. en. In: *Sensors* 21.14 (Jan. 2021). Number: 14 Publisher: Multidisciplinary Digital Publishing Institute, p. 4749. ISSN: 1424-8220. DOI: 10.3390/s21144749. URL: https://www.mdpi.com/1424-8220/21/14/4749 (visited on 10/26/2022).

[24] Sander van Donkelaar et al. “Superpixel-based Context Restoration for Self-supervised Pancreas Segmentation from CT scans”. English. In: Nov. 2022. URL: https://bnaic2022.uantwerpen.be/.

[25] Shiv Ram Dubey and Soumendu Chakraborty. “Average Biased ReLU Based CNN Descriptor for Improved Face Retrieval”. In: *Multimedia Tools and Applications* 80.15 (June 2021). arXiv:1804.02051 [cs], pp. 23181–23206. ISSN: 1380-7501, 1573-7721. DOI: 10.1007/s11042-020-10269-x. URL: http://arxiv.org/abs/1804.02051 (visited on 11/02/2022).

[26] Shiv Ram Dubey, Satish Kumar Singh, and Bidyut Baran Chaudhuri. “Activation functions in deep learning: A comprehensive survey and benchmark”. en. In: *Neurocomputing* 503 (Sept. 2022), pp. 92–108. ISSN: 0925-2312. DOI: 10.1016/j.neucom.2022.06.111. URL: https://www.sciencedirect.com/science/article/pii/S0925231222008426 (visited on 11/02/2022).

[27] Dutch Pancreatic Canc Grp et al. “Detection, Treatment, and Survival of Pancreatic Cancer Recurrence in the Netherlands A Nationwide Analysis”. In: *Annals of Surgery* 275.4 (Apr. 2022), pp. 769–775. ISSN: 0003-4932. DOI: 10.1097/SLA.0000000000004093.

[28] Yabo Fu et al. “A review of deep learning based methods for medical image multi-organ segmentation”. English. In: *Physica Medica: European Journal of Medical Physics* 85 (May 2021). Publisher: Elsevier, pp. 107–122. ISSN: 1120-1797. DOI: 10.1016/j.ejmp.2021.05.003. URL: https://www.physicamedica.com/article/S1120-1797(21)00184-8/fulltext (visited on 10/03/2022).

[29] Xavier Gastaldi. *Shake-Shake regularization*. Tech. rep. arXiv:1705.07485. arXiv:1705.07485 [cs] type: article. arXiv, May 2017. URL: http://arxiv.org/abs/1705.07485 (visited on 10/24/2022).

[30] Alexandru-Lucian Georgescu et al. “Performance vs. hardware requirements in state-of-the-art automatic speech recognition”. In: *EURASIP Journal on Audio, Speech, and Music Processing* 2021.1 (July 2021), p. 28. ISSN: 1687-4722. DOI: 10.1186/s13636-021-00217-4. URL: https://doi.org/10.1186/s13636-021-00217-4 (visited on 05/01/2023).

[31] Golnaz Ghiasi, Tsung-Yi Lin, and Quoc V Le. “DropBlock: A regularization method for convolutional networks”. In: *Advances in Neural Information Processing Systems*. Vol. 31. Curran Associates, Inc., 2018. URL: https://proceedings.neurips.cc/paper/2018/hash/7edcfb2d8f6a659ef4cd1e6c9b6d7079-Abstract.html (visited on 10/24/2022).

[32] Swarnendu Ghosh et al. “Understanding Deep Learning Techniques for Image Segmentation”. In: *ACM Computing Surveys* 52.4 (Aug. 2019), 73:1–73:35. ISSN: 0360-0300. DOI: 10.1145/3329784. URL: https://doi.org/10.1145/3329784 (visited on 10/12/2022).

[33] Boris Ginsburg et al. *Stochastic Gradient Methods with Layer-wise Adaptive Moments for Training of Deep Networks*. Tech. rep. arXiv:1905.11286. arXiv:1905.11286 [cs, stat] type: article. arXiv, Feb. 2020. URL: http://arxiv.org/abs/1905.11286 (visited on 04/26/2023).

[34] Lee W. Goldman. “Principles of CT and CT Technology”. en. In: *Journal of Nuclear Medicine Technology* 35.3 (Sept. 2007). Publisher: Society of Nuclear Medicine Section: CONTINUING EDUCATION, pp. 115–128. ISSN: 0091-4916, 1535-5675. DOI: 10.2967/jnmt.107.042978. URL: https://tech-snmjournals-org.eur.idm.oclc.org/content/35/3/115 (visited on 10/10/2022).

[35] Shuhang Gu et al. “Fast Image Restoration With Multi-Bin Trainable Linear Units”. In: 2019, pp. 4190–4199. URL: https://openaccess.thecvf.com/content_ICCV_2019/html/Gu_Fast_Image_Restoration_With_Multi-Bin_Trainable_Linear_Units_ICCV_2019_paper.html (visited on 11/02/2022).

[36] Zaiwang Gu et al. “CE-Net: Context Encoder Network for 2D Medical Image Segmentation”. In: *IEEE Transactions on Medical Imaging* 38.10 (Oct. 2019). Conference Name: IEEE Transactions on Medical Imaging, pp. 2281–2292. ISSN: 1558-254X. DOI: 10.1109/TMI.2019.2903562.

[37] Nan Guo et al. “Active Vision for Deep Visual Learning: A Unified Pooling Framework”. In: *IEEE Transactions on Industrial Informatics* 18.10 (Oct. 2022). Conference Name: IEEE Transactions on Industrial Informatics, pp. 6610–6618. ISSN: 1941-0050. DOI: 10.1109/TII.2021.3129813.

[38] Kaiming He et al. “Deep Residual Learning for Image Recognition”. In: 2016, pp. 770–778. URL: https://openaccess.thecvf.com/content_cvpr_2016/html/He_Deep_Residual_Learning_CVPR_2016_paper.html (visited on 10/26/2022).

[39] Kaiming He et al. “Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification”. In: 2015, pp. 1026–1034. URL: https://openaccess.thecvf.com/content_iccv_2015/html/He_Delving_Deep_into_ICCV_2015_paper.html (visited on 11/02/2022).

[40] G. Hemalatha and C. P. Sumathi. “Preprocessing techniques of facial image with Median and Gabor filters”. In: *2016 International Conference on Information Communication and Embedded Systems (ICICES)*. Feb. 2016, pp. 1–6. DOI: 10.1109/ICICES.2016.7518860.

[41] *Hierarchical 3D Feature Learning for Pancreas Segmentation*. original-date: 2021-03-09T08:51:12Z. Mar. 2023. URL: https://github.com/perceivelab/panknet (visited on 05/02/2023).

[42] Jie Hu, Li Shen, and Gang Sun. “Squeeze-and-Excitation Networks”. In: 2018, pp. 7132–7141. URL: https://openaccess.thecvf.com/content_cvpr_2018/html/Hu_Squeeze-and-Excitation_Networks_CVPR_2018_paper.html (visited on 11/09/2022).

[43] Yu-Hui Huang, Marc Proesmans, and Luc Van Gool. *Context-aware Padding for Semantic Segmentation*. Tech. rep. arXiv:2109.07854. arXiv:2109.07854 [cs] type: article. arXiv, Sept. 2021. URL: http://arxiv.org/abs/2109.07854 (visited on 10/19/2022).

[44] Lei Huang et al. *Normalization Techniques in Training DNNs: Methodology, Analysis and Application*. Tech. rep. arXiv:2009.12836. arXiv:2009.12836 [cs, stat] type: article. arXiv, Sept. 2020. URL: http://arxiv.org/abs/2009.12836 (visited on 10/26/2022).

[45] Zilong Huang et al. "CCNet: Criss-Cross Attention for Semantic Segmentation". In: 2019, pp. 603–612. URL: https://openaccess.thecvf.com/content_ICCV_2019/html/Huang_CCNet_Criss-Cross_Attention_for_Semantic_Segmentation_ICCV_2019_paper.html (visited on 11/09/2022).

[46] D.P. Huttenlocher, G.A. Klanderman, and W.J. Rucklidge. "Comparing images using the Hausdorff distance". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 15.9 (Sept. 1993). Conference Name: IEEE Transactions on Pattern Analysis and Machine Intelligence, pp. 850–863. ISSN: 1939-3539. DOI: 10.1109/34.232073.

[47] Sergey Ioffe and Christian Szegedy. "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift". en. In: (2015), p. 9.

[48] Fabian Isensee et al. "nnU-Net: a self-configuring method for deep learning-based biomedical image segmentation". en. In: *Nature Methods* 18.2 (Feb. 2021), pp. 203–211. ISSN: 1548-7091, 1548-7105. DOI: 10.1038/s41592-020-01008-z. URL: http://www.nature.com/articles/s41592-020-01008-z (visited on 05/03/2023).

[49] Md Amirul Islam et al. *Position, Padding and Predictions: A Deeper Look at Position Information in CNNs*. Tech. rep. arXiv:2101.12322. arXiv:2101.12322 [cs] type: article. arXiv, Jan. 2021. URL: http://arxiv.org/abs/2101.12322 (visited on 10/19/2022).

[50] L. Itti, C. Koch, and E. Niebur. "A model of saliency-based visual attention for rapid scene analysis". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 20.11 (Nov. 1998). Conference Name: IEEE Transactions on Pattern Analysis and Machine Intelligence, pp. 1254–1259. ISSN: 1939-3539. DOI: 10.1109/34.730558.

[51] Saumya Jetley et al. *Learn To Pay Attention*. Tech. rep. arXiv:1804.02391. arXiv:1804.02391 [cs] type: article. arXiv, Apr. 2018. URL: http://arxiv.org/abs/1804.02391 (visited on 11/09/2022).

[52] Terumi Kamisawa et al. "Pancreatic cancer". en. In: *The Lancet* 388.10039 (July 2016), pp. 73–85. ISSN: 0140-6736. DOI: 10.1016/S0140-6736(16)00141-0. URL: https://www.sciencedirect.com/science/article/pii/S0140673616001410 (visited on 09/30/2022).

[53] Konstantinos Kamnitsas et al. "Efficient multi-scale 3D CNN with fully connected CRF for accurate brain lesion segmentation". en. In: *Medical Image Analysis* 36 (Feb. 2017), pp. 61–78. ISSN: 1361-8415. DOI: 10.1016/j.media.2016.10.004. URL: https://www.sciencedirect.com/science/article/pii/S1361841516301839 (visited on 10/03/2022).

[54] Davood Karimi and Septimiu E. Salcudean. *Reducing the Hausdorff Distance in Medical Image Segmentation with Convolutional Neural Networks*. Tech. rep. arXiv:1904.10030. arXiv:1904.10030 [cs, eess, stat] type: article. arXiv, Apr. 2019. URL: http://arxiv.org/abs/1904.10030 (visited on 12/08/2022).

[55] John D. Kelleher. *Deep Learning*. en. Google-Books-ID: b06qDwAAQBAJ. MIT Press, Sept. 2019. ISBN: 978-0-262-53755-1.

[56] Eric Kerfoot et al. "Left-Ventricle Quantification Using Residual U-Net". en. In: *Statistical Atlases and Computational Models of the Heart. Atrial Segmentation and LV Quantification Challenges*. Ed. by Mihaela Pop et al. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2019, pp. 371–380. ISBN: 978-3-030-12029-0. DOI: 10.1007/978-3-030-12029-0_40.

[57] Hoel Kervadec et al. "Boundary loss for highly unbalanced segmentation". en. In: *Proceedings of The 2nd International Conference on Medical Imaging with Deep Learning*. ISSN: 2640-3498. PMLR, May 2019, pp. 285–296. URL: https://proceedings.mlr.press/v102/kervadec19a.html (visited on 12/07/2022).

[58] Soopil Kim et al. “Bidirectional RNN-based Few Shot Learning for 3D Medical Image Segmentation”. en. In: *Proceedings of the AAAI Conference on Artificial Intelligence* 35.3 (May 2021). Number: 3, pp. 1808–1816. ISSN: 2374-3468. URL: https://ojs.aaai.org/index.php/AAAI/article/view/16275 (visited on 09/07/2022).

[59] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. en. Number: arXiv:1412.6980 arXiv:1412.6980 [cs]. Jan. 2017. URL: http://arxiv.org/abs/1412.6980 (visited on 04/26/2023).

[60] Stefan Klein et al. “Automatic segmentation of the prostate in 3D MR images by atlas matching using localized mutual information: Automatic segmentation of the prostate in MR images”. en. In: *Medical Physics* 35.4 (Mar. 2008), pp. 1407–1417. ISSN: 00942405. DOI: 10.1118/1.2842076. URL: http://doi.wiley.com/10.1118/1.2842076 (visited on 12/12/2022).

[61] Alex Krizhevsky. “Learning Multiple Layers of Features from Tiny Images”. en. In: (2009), p. 60.

[62] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. “Deep learning”. en. In: *Nature* 521.7553 (May 2015). Number: 7553 Publisher: Nature Publishing Group, pp. 436–444. ISSN: 1476-4687. DOI: 10.1038/nature14539. URL: http://www.nature.com/articles/nature14539 (visited on 10/03/2022).

[63] Chen-Yu Lee et al. “Deeply-Supervised Nets”. en. In: *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics*. ISSN: 1938-7228. PMLR, Feb. 2015, pp. 562–570. URL: https://proceedings.mlr.press/v38/lee15a.html (visited on 11/14/2022).

[64] Feiyan Li et al. “Multiscale receptive field based on residual network for pancreas segmentation in CT images”. en. In: *Biomedical Signal Processing and Control* 57 (Mar. 2020), p. 101828. ISSN: 1746-8094. DOI: 10.1016/j.bspc.2019.101828. URL: https://www.sciencedirect.com/science/article/pii/S1746809419304094 (visited on 11/28/2022).

[65] Jason Li et al. *Jasper: An End-to-End Convolutional Neural Acoustic Model*. Tech. rep. arXiv:1904.03288. arXiv:1904.03288 [cs, eess] type: article. arXiv, Aug. 2019. URL: http://arxiv.org/abs/1904.03288 (visited on 05/01/2023).

[66] Shan Sung Liew, Mohamed Khalil-Hani, and Rabia Bakhteri. “Bounded activation functions for enhanced training stability of deep neural networks on visual pattern recognition problems”. en. In: *Neurocomputing* 216 (Dec. 2016), pp. 718–734. ISSN: 0925-2312. DOI: 10.1016/j.neucom.2016.08.037. URL: https://www.sciencedirect.com/science/article/pii/S0925231216308797 (visited on 11/02/2022).

[67] Sang-Heon Lim et al. “Automated pancreas segmentation and volumetry using deep neural network on computed tomography”. en. In: *Scientific Reports* 12.1 (Mar. 2022). Number: 1 Publisher: Nature Publishing Group, p. 4075. ISSN: 2045-2322. DOI: 10.1038/s41598-022-07848-3. URL: https://www.nature.com/articles/s41598-022-07848-3 (visited on 10/03/2022).

[68] Tsung-Yi Lin et al. “Focal Loss for Dense Object Detection”. In: 2017, pp. 2980–2988. URL: https://openaccess.thecvf.com/content_iccv_2017/html/Lin_Focal_Loss_for_ICCV_2017_paper.html (visited on 12/07/2022).

[69] Tsung-Yi Lin et al. *Microsoft COCO: Common Objects in Context*. Tech. rep. arXiv:1405.0312. arXiv:1405.0312 [cs] type: article. arXiv, Feb. 2015. DOI: 10.48550/arXiv.1405.0312. URL: http://arxiv.org/abs/1405.0312 (visited on 06/18/2023).

[70] Giuseppe Lippi and Camilla Mattiuzzi. “The global burden of pancreatic cancer”. In: *Archives of Medical Science* 16.4 (May 2020). Publisher: Termedia Publishing House, pp. 820–824. ISSN: 1734-1922, 1896-9151. DOI: 10.5114/aoms.2020.94845. URL: https://www.archivesofmedicalscience.com/The-global-burden-of-pancreatic-cancer,112351,0,2.html (visited on 09/28/2022).

[71] Guilin Liu et al. *Partial Convolution based Padding*. Tech. rep. arXiv:1811.11718. arXiv:1811.11718 [cs] type: article. arXiv, Nov. 2018. URL: http://arxiv.org/abs/1811.11718 (visited on 10/19/2022).

[72] Hao Ma, Yanni Zou, and Peter X. Liu. “MHSU-Net: A more versatile neural network for medical image segmentation”. en. In: *Computer Methods and Programs in Biomedicine* 208 (Sept. 2021), p. 106230. ISSN: 0169-2607. DOI: 10.1016/j.cmpb.2021.106230. URL: https://www.sciencedirect.com/science/article/pii/S0169260721003047 (visited on 11/23/2022).

[73] Jun Ma et al. *AbdomenCT-1K: Is Abdominal Organ Segmentation A Solved Problem?* Tech. rep. arXiv:2010.14808. arXiv:2010.14808 [cs] type: article. arXiv, July 2021. DOI: 10.48550/arXiv.2010.14808. URL: http://arxiv.org/abs/2010.14808 (visited on 09/28/2022).

[74] Jun Ma et al. "Loss odyssey in medical image segmentation". en. In: *Medical Image Analysis* 71 (July 2021), p. 102035. ISSN: 1361-8415. DOI: 10.1016/j.media.2021.102035. URL: https://www.sciencedirect.com/science/article/pii/S1361841521000815 (visited on 12/06/2022).

[75] Andrew L Maas, Awni Y Hannun, and Andrew Y Ng. "Rectifier Nonlinearities Improve Neural Network Acoustic Models". en. In: (2013), p. 6.

[76] Lena Maier-Hein et al. *Metrics reloaded: Pitfalls and recommendations for image analysis validation*. Tech. rep. arXiv:2206.01653. arXiv:2206.01653 [cs] type: article. arXiv, Sept. 2022. URL: http://arxiv.org/abs/2206.01653 (visited on 12/12/2022).

[77] Yunze Man et al. "Deep Q Learning Driven CT Pancreas Segmentation with Geometry-Aware U-Net". In: *IEEE Transactions on Medical Imaging* 38.8 (Aug. 2019). arXiv:1904.09120 [cs], pp. 1971–1980. ISSN: 0278-0062, 1558-254X. DOI: 10.1109/TMI.2019.2911588. URL: http://arxiv.org/abs/1904.09120 (visited on 09/08/2022).

[78] Dou El Kefel Mansouri et al. "The Mode-Fisher pooling for time complexity optimization in deep convolutional neural networks". In: *Neural Computing and Applications* 33.12 (June 2021), pp. 6443–6465. ISSN: 1433-3058. DOI: 10.1007/s00521-020-05406-4. URL: https://doi.org/10.1007/s00521-020-05406-4.

[79] Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi. *V-Net: Fully Convolutional Neural Networks for Volumetric Medical Image Segmentation*. Tech. rep. arXiv:1606.04797. arXiv:1606.04797 [cs] type: article. arXiv, June 2016. URL: http://arxiv.org/abs/1606.04797 (visited on 10/12/2022).

[80] Shervin Minaee et al. *Image Segmentation Using Deep Learning: A Survey*. Tech. rep. arXiv:2001.05566. arXiv:2001.05566 [cs] type: article. arXiv, Nov. 2020. URL: http://arxiv.org/abs/2001.05566 (visited on 10/07/2022).

[81] Jordi Minnema et al. "Comparison of convolutional neural network training strategies for cone-beam CT image segmentation". en. In: *Computer Methods and Programs in Biomedicine* 207 (Aug. 2021), p. 106192. ISSN: 0169-2607. DOI: 10.1016/j.cmpb.2021.106192. URL: https://www.sciencedirect.com/science/article/pii/S0169260721002662 (visited on 09/21/2022).

[82] Diganta Misra. *Mish: A Self Regularized Non-Monotonic Activation Function*. Tech. rep. arXiv:1908.08681. arXiv:1908.08681 [cs, stat] type: article. arXiv, Aug. 2020. URL: http://arxiv.org/abs/1908.08681 (visited on 11/02/2022).

[83] Diganta Misra et al. "Rotate to Attend: Convolutional Triplet Attention Module". en. In: *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*. Waikoloa, HI, USA: IEEE, Jan. 2021, pp. 3138–3147. ISBN: 978-1-66540-477-8. DOI: 10.1109/WACV48630.2021.00318. URL: https://ieeexplore.ieee.org/document/9423300/ (visited on 11/11/2022).

[84] *MLflow: A Machine Learning Lifecycle Platform*. original-date: 2018-06-05T16:05:58Z. June 2023. URL: https://github.com/mlflow/mlflow (visited on 06/19/2023).

[85] Volodymyr Mnih et al. "Recurrent Models of Visual Attention". In: *Advances in Neural Information Processing Systems*. Vol. 27. Curran Associates, Inc., 2014. URL: https://proceedings.neurips.cc/paper/2014/hash/09c6c3783b4a70054da74f2538ed47c6-Abstract.html (visited on 11/09/2022).

[86] Alejandro Molina, Patrick Schramowski, and Kristian Kersting. *Pad\'e Activation Units: End-to-end Learning of Flexible Activation Functions in Deep Networks*. Tech. rep. arXiv:1907.06732. arXiv:1907.06732 [cs] type: article. arXiv, Feb. 2020. URL: http://arxiv.org/abs/1907.06732 (visited on 11/02/2022).

[87] Vinod Nair and Geoffrey E Hinton. "Rectified Linear Units Improve Restricted Boltzmann Machines". en. In: (2010), p. 8.

[88] National Cancer Institute. *Cancer of the Pancreas - Cancer Stat Facts*. en. URL: https://seer.cancer.gov/statfacts/html/pancreas.html (visited on 09/28/2022).

[89] Anh-Duc Nguyen et al. “Distribution Padding in Convolutional Neural Networks”. In: *2019 IEEE International Conference on Image Processing (ICIP)*. ISSN: 2381-8549. Sept. 2019, pp. 4275–4279. DOI: 10.1109/ICIP.2019.8803537.

[90] Stanislav Nikolov et al. “Clinically Applicable Segmentation of Head and Neck Anatomy for Radiotherapy: Deep Learning Algorithm Development and Validation Study”. EN. In: *Journal of Medical Internet Research* 23.7 (July 2021). Company: Journal of Medical Internet Research Distributor: Journal of Medical Internet Research Institution: Journal of Medical Internet Research Label: Journal of Medical Internet Research Publisher: JMIR Publications Inc., Toronto, Canada, e26151. DOI: 10.2196/26151. URL: https://www.jmir.org/2021/7/e26151 (visited on 12/12/2022).

[91] NVIDIA, Péter Vingelmann, and Frank H.P. Fitzek. *CUDA, release: 11.6*. 2020. URL: https://developer.nvidia.com/cuda-toolkit.

[92] William Ocasio. “Attention to Attention”. In: *Organization Science* 22.5 (Oct. 2011). Publisher: INFORMS, pp. 1286–1296. ISSN: 1047-7039. DOI: 10.1287/orsc.1100.0602. URL: https://pubsonline.informs.org/doi/10.1287/orsc.1100.0602 (visited on 11/07/2022).

[93] Ozan Oktay et al. *Attention U-Net: Learning Where to Look for the Pancreas*. en. Number: arXiv:1804.03999 arXiv:1804.03999 [cs]. May 2018. URL: http://arxiv.org/abs/1804.03999 (visited on 11/09/2022).

[94] *Partial Convolution Layer for Padding and Image Inpainting*. original-date: 2018-11-05T03:02:27Z. June 2023. URL: https://github.com/NVIDIA/partialconv (visited on 06/07/2023).

[95] Adam Paszke et al. “PyTorch: An Imperative Style, High-Performance Deep Learning Library”. In: *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 8024–8035. URL: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[96] Hieu Pham and Quoc Le. “AutoDropout: Learning Dropout Patterns to Regularize Deep Networks”. en. In: *Proceedings of the AAAI Conference on Artificial Intelligence* 35.11 (May 2021). Number: 11, pp. 9351–9359. ISSN: 2374-3468. DOI: 10.1609/aaai.v35i11.17127. URL: https://ojs.aaai.org/index.php/AAAI/article/view/17127 (visited on 10/24/2022).

[97] *platipy: Processing Library and Analysis Toolkit for Medical Imaging in Python*.

[98] Pragati Baheti. *Activation Functions in Neural Networks [12 Types & Use Cases]*. en. 2022. URL: https://www.v7labs.com/blog/neural-networks-activation-functions,%20https://www.v7labs.com/blog/neural-networks-activation-functions (visited on 10/31/2022).

[99] Federica Proietto Salanitri et al. “Hierarchical 3D Feature Learning forPancreas Segmentation”. en. In: *Machine Learning in Medical Imaging*. Ed. by Chunfeng Lian et al. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2021, pp. 238–247. ISBN: 978-3-030-87589-3. DOI: 10.1007/978-3-030-87589-3_25.

[100] Yi Qu, Min Xia, and Yonghong Zhang. “Strip pooling channel spatial attention network for the segmentation of cloud and cloud shadow”. en. In: *Computers & Geosciences* 157 (Dec. 2021), p. 104940. ISSN: 0098-3004. DOI: 10.1016/j.cageo.2021.104940. URL: https://www.sciencedirect.com/science/article/pii/S0098300421002259 (visited on 11/11/2022).

[101] Prajit Ramachandran, Barret Zoph, and Quoc V. Le. *Searching for Activation Functions*. en. Number: arXiv:1710.05941 arXiv:1710.05941 [cs]. Oct. 2017. URL: http://arxiv.org/abs/1710.05941 (visited on 11/02/2022).

[102] Prashanth Rawla, Tagore Sunkara, and Vinaya Gaduputi. “Epidemiology of Pancreatic Cancer: Global Trends, Etiology and Risk Factors”. In: *World Journal of Oncology* 10.1 (Feb. 2019), pp. 10–27. ISSN: 1920-454X. DOI: 10.4021/wjon.v10i1.1166. URL: https://www.wjon.org/index.php/wjon/article/view/1166.

[103] Ronald Rensink. “Visual Search for Change: A Probe into the Nature of Attentional Processing”. In: *Visual Cognition* 7 (Jan. 2000), pp. 345–376. DOI: 10.1080/135062800394847.

[104] Rachid Riad et al. *Learning strides in convolutional neural networks*. Tech. rep. arXiv:2202.01653. arXiv:2202.01653 [cs] type: article. arXiv, Feb. 2022. URL: http://arxiv.org/abs/2202.01653 (visited on 10/19/2022).

[105] Anne-Marie Rickmann et al. “Recalibrating 3D ConvNets with Project & Excite”. In: *IEEE Transactions on Medical Imaging* 39.7 (July 2020). arXiv:2002.10994 [cs, eess, stat], pp. 2461–2471. ISSN: 0278-0062, 1558-254X. DOI: 10.1109/TMI.2020.2972059. URL: http://arxiv.org/abs/2002.10994 (visited on 11/28/2022).

[106] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. “U-Net: Convolutional Networks for Biomedical Image Segmentation”. en. In: *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*. Ed. by Nassir Navab et al. Vol. 9351. Series Title: Lecture Notes in Computer Science. Cham: Springer International Publishing, 2015, pp. 234–241. ISBN: 978-3-319-24573-7 978-3-319-24574-4. DOI: 10.1007/978-3-319-24574-4_28. URL: http://link.springer.com/10.1007/978-3-319-24574-4_28 (visited on 10/12/2022).

[107] Holger R. Roth et al. *Data From Pancreas-CT. The Cancer Imaging Archive.* 2016. URL: https://doi.org/10.7937/K9/TCIA.2016.tNB1kqBU.

[108] Holger R. Roth et al. “DeepOrgan: Multi-level Deep Convolutional Networks for Automated Pancreas Segmentation”. en. In: *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*. Ed. by Nassir Navab et al. Vol. 9349. Series Title: Lecture Notes in Computer Science. Cham: Springer International Publishing, 2015, pp. 556–564. ISBN: 978-3-319-24552-2 978-3-319-24553-9. DOI: 10.1007/978-3-319-24553-9_68. URL: http://link.springer.com/10.1007/978-3-319-24553-9_68 (visited on 09/05/2022).

[109] Olga Russakovsky et al. “ImageNet Large Scale Visual Recognition Challenge”. In: *International Journal of Computer Vision (IJCV)* 115.3 (2017), pp. 211–252. DOI: 10.1007/s11263-015-0816-y.

[110] Tim Salimans and Durk P Kingma. “Weight Normalization: A Simple Reparameterization to Accelerate Training of Deep Neural Networks”. In: *Advances in Neural Information Processing Systems*. Vol. 29. Curran Associates, Inc., 2016. URL: https://proceedings.neurips.cc/paper/2016/hash/ed265bc903a5a097f61d3ec064d96d2e-Abstract.html (visited on 10/28/2022).

[111] Mark Sandler et al. “MobileNetV2: Inverted Residuals and Linear Bottlenecks”. In: 2018, pp. 4510–4520. URL: https://openaccess.thecvf.com/content_cvpr_2018/html/Sandler_MobileNetV2_Inverted_Residuals_CVPR_2018_paper.html (visited on 11/23/2022).

[112] Claudio Filipi Goncalves do Santos et al. “MaxDropout: Deep Neural Network Regularization Based on Maximum Output Values”. In: *2020 25th International Conference on Pattern Recognition (ICPR)*. ISSN: 1051-4651. Jan. 2021, pp. 2671–2676. DOI: 10.1109/ICPR48806.2021.9412733.

[113] Claudio Filipi Gonçalves Dos Santos and João Paulo Papa. “Avoiding Overfitting: A Survey on Regularization Methods for Convolutional Neural Networks”. en. In: *ACM Computing Surveys* 54.10s (Jan. 2022), pp. 1–25. ISSN: 0360-0300, 1557-7341. DOI: 10.1145/3510413. URL: https://dl.acm.org/doi/10.1145/3510413 (visited on 10/24/2022).

[114] Junya Sato and Shoji Kido. *Large Batch and Patch Size Training for Medical Image Segmentation*. Tech. rep. arXiv:2210.13364. arXiv:2210.13364 [cs, eess] type: article. arXiv, Oct. 2022. URL: http://arxiv.org/abs/2210.13364 (visited on 05/03/2023).

[115] Rebecca L. Siegel et al. “Cancer statistics, 2022”. en. In: *CA: A Cancer Journal for Clinicians* 72.1 (2022). _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.3322/caac.21708, pp. 7–33. ISSN: 1542-4863. DOI: 10.3322/caac.21708. URL: https://onlinelibrary.wiley.com/doi/abs/10.3322/caac.21708 (visited on 06/21/2023).

[116] Satya P. Singh et al. “3D Deep Learning on Medical Images: A Review”. en. In: *Sensors* 20.18 (Jan. 2020). Number: 18 Publisher: Multidisciplinary Digital Publishing Institute, p. 5097. ISSN: 1424-8220. DOI: 10.3390/s20185097. URL: https://www.mdpi.com/1424-8220/20/18/5097 (visited on 09/23/2022).

[117] Jost Tobias Springenberg et al. *Striving for Simplicity: The All Convolutional Net*. Tech. rep. arXiv:1412.6806. arXiv:1412.6806 [cs] type: article. arXiv, Apr. 2015. DOI: 10.48550/arXiv.1412.6806. URL: http://arxiv.org/abs/1412.6806 (visited on 10/12/2022).

[118] Nitish Srivastava et al. “Dropout: A Simple Way to Prevent Neural Networks from Overfitting”. en. In: (2014), p. 30.

[119] Cecilia Summers and Michael J. Dinneen. *Four Things Everyone Should Know to Improve Batch Normalization*. Tech. rep. arXiv:1906.03548. arXiv:1906.03548 [cs, stat] type: article. arXiv, Feb. 2020. DOI: 10.48550/arXiv.1906.03548. URL: http://arxiv.org/abs/1906.03548 (visited on 10/28/2022).

[120] Manli Sun et al. "Learning Pooling for Convolutional Neural Network". en. In: *Neurocomputing* 224 (Feb. 2017), pp. 96–104. ISSN: 0925-2312. DOI: 10.1016/j.neucom.2016.10.049. URL: https://www.sciencedirect.com/science/article/pii/S0925231216312905 (visited on 10/17/2022).

[121] *Surface distance metrics*. original-date: 2018-07-19T13:46:05Z. May 2023. URL: https://github.com/deepmind/surface-distance (visited on 05/02/2023).

[122] Saeid Asgari Taghanaki et al. "Combo loss: Handling input and output imbalance in multi-organ segmentation". en. In: *Computerized Medical Imaging and Graphics* 75 (July 2019), pp. 24–33. ISSN: 0895-6111. DOI: 10.1016/j.compmedimag.2019.04.005. URL: https://www.sciencedirect.com/science/article/pii/S0895611118305688 (visited on 12/08/2022).

[123] Saeid Asgari Taghanaki et al. *Deep Semantic Segmentation of Natural and Medical Images: A Review*. Tech. rep. arXiv:1910.07655. arXiv:1910.07655 [cs, eess] type: article. arXiv, June 2020. URL: http://arxiv.org/abs/1910.07655 (visited on 12/06/2022).

[124] Xiaozhong Tong et al. "ASCU-Net: Attention Gate, Spatial and Channel Attention U-Net for Skin Lesion Segmentation". en. In: *Diagnostics* 11.3 (Mar. 2021). Number: 3 Publisher: Multidisciplinary Digital Publishing Institute, p. 501. ISSN: 2075-4418. DOI: 10.3390/diagnostics11030501. URL: https://www.mdpi.com/2075-4418/11/3/501 (visited on 11/07/2022).

[125] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. *Instance Normalization: The Missing Ingredient for Fast Stylization*. Tech. rep. arXiv:1607.08022. arXiv:1607.08022 [cs] version: 3 type: article. arXiv, Nov. 2017. URL: http://arxiv.org/abs/1607.08022 (visited on 10/28/2022).

[126] Guido Van Rossum and Fred L. Drake. *Python 3 Reference Manual*. Scotts Valley, CA: CreateSpace, 2009. ISBN: 1-4414-1269-7.

[127] Cristina Vasconcelos et al. "Impact of Aliasing on Generalization in Deep Convolutional Networks". en. In: 2021, pp. 10529–10538. URL: https://openaccess.thecvf.com/content/ICCV2021/html/Vasconcelos_Impact_of_Aliasing_on_Generalization_in_Deep_Convolutional_Networks_ICCV_2021_paper.html (visited on 10/19/2022).

[128] Fei Wang et al. "Residual Attention Network for Image Classification". en. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Honolulu, HI, USA: IEEE, July 2017, pp. 6450–6458. ISBN: 978-1-5386-0457-1. DOI: 10.1109/CVPR.2017.683. URL: https://ieeexplore.ieee.org/document/8100166/ (visited on 11/09/2022).

[129] Yuan Wang et al. "Pancreas segmentation using a dual-input v-mesh network". en. In: *Medical Image Analysis* 69 (Apr. 2021), p. 101958. ISSN: 1361-8415. DOI: 10.1016/j.media.2021.101958. URL: https://www.sciencedirect.com/science/article/pii/S1361841521000049 (visited on 11/28/2022).

[130] Ken C. L. Wong et al. "3D Segmentation with Exponential Logarithmic Loss for Highly Unbalanced Object Sizes". en. In: *Medical Image Computing and Computer Assisted Intervention – MICCAI 2018*. Ed. by Alejandro F. Frangi et al. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2018, pp. 612–619. ISBN: 978-3-030-00931-1. DOI: 10.1007/978-3-030-00931-1_70.

[131] Sanghyun Woo et al. "CBAM: Convolutional Block Attention Module". en. In: *Computer Vision – ECCV 2018*. Ed. by Vittorio Ferrari et al. Vol. 11211. Series Title: Lecture Notes in Computer Science. Cham: Springer International Publishing, 2018, pp. 3–19. ISBN: 978-3-030-01233-5 978-3-030-01234-2. DOI: 10.1007/978-3-030-01234-2_1. URL: https://link.springer.com/10.1007/978-3-030-01234-2_1 (visited on 11/09/2022).

[132] Yuxin Wu and Kaiming He. *Group Normalization*. Tech. rep. arXiv:1803.08494. arXiv:1803.08494 [cs] type: article. arXiv, June 2018. URL: http://arxiv.org/abs/1803.08494 (visited on 10/28/2022).

[133] Zifeng Wu, Chunhua Shen, and Anton van den Hengel. *Bridging Category-level and Instance-level Semantic Image Segmentation*. Tech. rep. arXiv:1605.06885. arXiv:1605.06885 [cs] type: article. arXiv, May 2016. URL: http://arxiv.org/abs/1605.06885 (visited on 12/06/2022).

[134] Haiying Xia et al. "MC-Net: multi-scale context-attention network for medical CT image segmentation". en. In: *Applied Intelligence* 52.2 (Jan. 2022), pp. 1508–1519. ISSN: 1573-7497. DOI: 10.1007/s10489-021-02506-z. URL: https://doi.org/10.1007/s10489-021-02506-z (visited on 11/14/2022).

[135] Haiying Xia et al. "Md-Net: Multi-scale Dilated Convolution Network for CT Images Segmentation". en. In: *Neural Processing Letters* 51.3 (June 2020), pp. 2915–2927. ISSN: 1573-773X. DOI: 10.1007/s11063-020-10230-x. URL: https://doi.org/10.1007/s11063-020-10230-x (visited on 11/14/2022).

[136] Saining Xie et al. "Rethinking Spatiotemporal Feature Learning: Speed-Accuracy Trade-offs in Video Classification". In: 2018, pp. 305–321. URL: https://openaccess.thecvf.com/content_ECCV_2018/html/Saining_Xie_Rethinking_Spatiotemporal_Feature_ECCV_2018_paper.html (visited on 11/23/2022).

[137] Yoshihiro Yamada et al. "Shakedrop Regularization for Deep Residual Learning". In: *IEEE Access* 7 (2019). Conference Name: IEEE Access, pp. 186126–186136. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2019.2960566.

[138] Fisher Yu and Vladlen Koltun. *Multi-Scale Context Aggregation by Dilated Convolutions*. Tech. rep. arXiv:1511.07122. arXiv:1511.07122 [cs] type: article. arXiv, Apr. 2016. DOI: 10.48550/arXiv.1511.07122. URL: http://arxiv.org/abs/1511.07122 (visited on 10/19/2022).

[139] Luiz Zaniolo and Oge Marques. "On the use of variable stride in convolutional neural networks". In: *Multimedia Tools and Applications* 79.19 (May 2020), pp. 13581–13598. ISSN: 1573-7721. DOI: 10.1007/s11042-019-08385-4. URL: https://doi.org/10.1007/s11042-019-08385-4.

[140] Qiulin Zhang et al. *Split to Be Slim: An Overlooked Redundancy in Vanilla Convolution*. Tech. rep. arXiv:2006.12085. arXiv:2006.12085 [cs] type: article. arXiv, June 2020. URL: http://arxiv.org/abs/2006.12085 (visited on 11/14/2022).

[141] Ningning Zhao et al. *Fully Automated Pancreas Segmentation with Two-stage 3D Convolutional Neural Networks*. Tech. rep. arXiv:1906.01795. arXiv:1906.01795 [cs] type: article. arXiv, July 2019. URL: http://arxiv.org/abs/1906.01795 (visited on 11/22/2022).

[142] Bolei Zhou et al. "Scene Parsing through ADE20K Dataset". en. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Honolulu, HI: IEEE, July 2017, pp. 5122–5130. ISBN: 978-1-5386-0457-1. DOI: 10.1109/CVPR.2017.544. URL: http://ieeexplore.ieee.org/document/8100027/ (visited on 06/18/2023).

[143] Yuyin Zhou et al. *Deep Supervision for Pancreatic Cyst Segmentation in Abdominal CT Scans*. Tech. rep. arXiv:1706.07346. arXiv:1706.07346 [cs] type: article. arXiv, June 2017. URL: http://arxiv.org/abs/1706.07346 (visited on 11/18/2022).

[144] Zongwei Zhou et al. "UNet++: A Nested U-Net Architecture for Medical Image Segmentation". In: *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support : 4th International Workshop, DLMIA 2018, and 8th International Workshop, ML-CDS 2018, held in conjunction with MICCAI 2018, Granada, Spain, S* 11045 (Sept. 2018), pp. 3–11. DOI: 10.1007/978-3-030-00889-5_1. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7329239/ (visited on 11/14/2022).

[145] Hui Zhu and Xiaofang Zhao. "TargetDrop: A Targeted Regularization Method for Convolutional Neural Networks". In: *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. ISSN: 2379-190X. May 2022, pp. 3283–3287. DOI: 10.1109/ICASSP43922.2022.9746657.

[146] Wentao Zhu et al. "AnatomyNet: Deep learning for fast and fully automated whole-volume segmentation of head and neck anatomy". en. In: *Medical Physics* 46.2 (2019), pp. 576–589. ISSN: 2473-4209. DOI: 10.1002/mp.13300. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/mp.13300 (visited on 12/08/2022).

[147] Zhuotun Zhu et al. *A 3D Coarse-to-Fine Framework for Volumetric Medical Image Segmentation*. Tech. rep. arXiv:1712.00201. arXiv:1712.00201 [cs] type: article. arXiv, Aug. 2018. URL: http://arxiv.org/abs/1712.00201 (visited on 11/18/2022).

[148] Barret Zoph and Quoc V. Le. *Neural Architecture Search with Reinforcement Learning*. Tech. rep. arXiv:1611.01578. arXiv:1611.01578 [cs] type: article. arXiv, Feb. 2017. DOI: 10.48550/arXiv.1611.01578. URL: http://arxiv.org/abs/1611.01578 (visited on 10/19/2022).