# Deep Convolutional Neural Networks for Segmenting Unruptured Intracranial Aneurysms from 3D TOF-MRA Images

Surasith Boonaneksap

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
in
Computer Engineering

Creed Jones, Chair
Ruoxi Jia
Ismini Lourentzou

December 15th, 2021
Blacksburg, Virginia

Keywords: Convolutional Neural Networks, Intracranial Aneurysms, Image Segmentation

# Deep Convolutional Neural Networks for Segmenting Unruptured Intracranial Aneurysms from 3D TOF-MRA Images

Surasith Boonaneksap

(ABSTRACT)

Despite facing technical issues (e.g., overfitting, vanishing and exploding gradients), deep neural networks have the potential to capture complex patterns in data. Understanding how depth impacts neural networks performance is vital to the advancement of novel deep learning architectures. By varying hyperparameters on two sets of architectures with different depths, this thesis aims to examine if there are any potential benefits from developing deep networks for segmenting intracranial aneurysms from 3D TOF-MRA scans in the ADAM dataset.

# Deep Convolutional Neural Networks for Segmenting Unruptured Intracranial Aneurysms from 3D TOF-MRA Images

Surasith Boonaneksap

(GENERAL AUDIENCE ABSTRACT)

With the technologies we have today, people are constantly generating data. In this pool of information, gaining insight into the data proves to be extremely valuable. Deep learning is one method that allows for automatic pattern recognition by iteratively improving the disparity between its prediction and the ground truth. Complex models can learn complex patterns, and such models introduce challenges. This thesis explores the potential benefits of deep neural networks whether they stand to gain improvement despite the challenges. The models will be trained to segment intracranial aneurysms from volumetric images.

# Contents

# List of Figures

# List of Tables

# List of Abbreviations

ADAM Aneurysm Detection And segMentation Challenge 2020

API Angiographic Parametric Imaging

ARC Advanced Research Computing

AUC Area under the curve

CNN Convolutional Neural Network

CPU Central Processing Unit

CTA Computed Tomography Angiography

DSA Digital Subtraction Angiography

DSC Dice Similarity Coefficient

FC Fully-connected

GFD Generic Fourier Descriptor

GPU Graphical Processing Unit

HPC High Performance Computing

IoU Intersection over Union

JI Jaccard Index

k-NN k-Nearest Neighbors

LBP Local Binary Pattern

MIP Maximum Intensity Projection

MLP Multi-layer Perceptron

MRA Magnetic Resonance Angiography

RA Rotational Angiography

ReLU Rectified Linear Unit

ROC Receiver Operating Characteristic

ROI Region of Interest

SGD Stochastic Gradient Descent

TOF Time-of-flight

WBC Weighted binary cross-entropy

# Chapter 1

# Introduction

Deep learning has become an integral part of our daily routines with applications such as content recommendations, spam filtration systems, and voice assistance. It has also proven to be useful beyond these products and services. Its ability to learn from data without feature extraction has unlocked the potential to learn complex patterns in a way that does not require humans to manually construct sophisticated algorithms.

Researchers always strive to develop better and more useful deep learning algorithms to take advantage of the massive amount of information we have today. This requires artificial neural networks to be more innovative and have more parameters. At some point, however, the neural networks would encounter problems when the model is simply too large. The problems of overfitting, and vanishing and exploding gradient have been a part of the challenge in developing novel model architectures.

In the field of medical imaging, radiologists obtain images from the human body to detect certain anomalies from the scans — ideally in a non-invasive manner — and analyze them according to their professional practice. Unlike other forms of imaging scans, the machines used to capture the internal structure of a human body cannot use light reflecting off of the subjects and need to be approached differently whether through the use of radiation or magnetization. Some of these techniques are considered to be invasive. This means that the human body has to be exposed to foreign materials.

Intracranial aneurysms are balloon-shaped tissues protruding from the blood vessels. Small and unruptured aneurysms do not pose significant damage to the brain. Aneurysms that rupture, however, can be very dangerous and fatal. Early detection and constant monitoring are necessary. To capture this via MRI — an imaging technique used to view soft tissues — a contrast agent needs to be injected to help with the visualization [4]. This is considered to be invasive and would be best if avoided. In the absence of contrast agents, the task of detecting the aneurysms becomes more challenging. This is where automatic image segmentation using deep learning would be able to assist the radiologists.

Developing a complex neural network architecture will allow a model to recognize complex patterns. This thesis explores the use of deep convolutional neural networks for this application to discover whether the choices and values of hyperparameters can help increase the accuracy.

# Chapter 2

# Review of Literature

To understand the motivation behind this thesis, it is worth looking into what contributions have been made so far in this field. Over the past decade, many researchers have proposed different methods to automatically detect, locate, and segment intracranial aneurysms across many imaging modalities. This section will explore various aspects of those proposals - the benefits, drawbacks, and potential research gap.

## 2.1 Traditional Methods

With the recent advancement in computer hardware and the rapid growth of digital information, machine learning has quickly become a norm for solving all sorts of problems in many fields. These include data analytics, applied statistics, natural language processing, and computer vision. With promising results, it seems reasonable to think that the first method of choice should be machine-learning-based. However, it is important to look at how automated aneurysms detection performs without using algorithms that rely on finding patterns in the data, not only to see whether machine learning can provide improvements but also to see if it warrants the time to step away from the traditional methods.

In 2011, Larrabide et al. [21] proposed a method for segmenting intracranial aneurysms from 3D digital subtraction angiography (DSA) images by using a deformable model. The process skeletonizes and calculates the centerline of the segmented voxel. A cylindrical simplex mesh

is formed around the centerline and expands until it reaches the vessel walls. The region where the mesh does not come into contact with is the neck of the aneurysm. The aneurysm itself is where the closest point on the surface of the vessels is a part of the aneurysm's neck. The result from testing on 26 3D-DSA images consisting of saccular intracranial aneurysms shows that the proposed method can get a result close to the ground truth qualitatively [1]. Because the process of getting the region of interest (ROI) is out of scope for this paper, it cannot be verified how well the method can identify and locate the aneurysms.

Also in 2011, Cardenes et al. [8] proposed another method for aneurysm segmentation. The initial process is similar to Larrabide et al.'s method. The input voxels have already been accurately segmented as well as the centerline. The method then classifies centerline branches. Every point on the surface of the vessels calculates the closest branch. If the closest branch is an aneurysm branch, then that point is also part of an aneurysm. This calculation applies to other types of branches that form a Voronoi surface as well. The result serves as a good initial segmentation, but the result can overestimate or underestimate the aneurysm region. For this reason, the authors increase the initial region size and employed fast marching and backtracking to find the best cutoff point to be the neck of the aneurysms. Fast marching finds a distance field to a fixed point on a surface — later chosen to be a point closest to the bifurcation point — which will be used to find a closed loop around the aneurysms. Then the backtracking process uses the surface gradient to find the neck — the minimum length closed curve that can form using two paths that make a $\pi$ angle from each other.

Both methods provided good results despite variations in the shape and size of the aneurysms. However, there are some limitations. First, both methods are not fully automated. This means that a separate process for selecting ROI needs to be developed, and the result must

[1] Quantitative analysis exists but concerns with quantification accuracy.

be good enough to fit the requirements set by both methods. Otherwise, good results as demonstrated in the paper may be unachievable. Second, both methods assume a particular topology of the aneurysms — namely saccular aneurysms. Although 90% of all aneurysms are saccular [38], these methods will not be able to detect other less common types of aneurysms where its topology is not the same as saccular. In the case of fusiform aneurysms, for example, the "neck" of the aneurysm is wider than its dome [41].

As opposed to finding a structure that matches that of an aneurysm, a method proposed by Lauric et al. [22] in 2010 finds an anomaly instead. The method uses a writhe number to check the symmetry of a vessel. If the vessel in a local neighborhood is straight and cylindrical or parabolic, the writhe number is zero. Otherwise, it is non-zero. This non-zero region is flagged as an aneurysm. In addition to the primary detection step, the authors also proposed a false positive reduction step that set a threshold for region index which is a product of the volume of a voxel and the number of voxels in a region. The method achieved 100% sensitivity with false positives of 0.66 per case when tested on 10 3D rotational angiography (RA) images and 5.36 per case when tested on 10 computed tomography angiography (CTA). The authors claimed that CTA has lower resolution compared to 3D-RA which results in more false positives. Unlike the previous two methods which tested only on saccular aneurysms, this method uses a test set that includes saccular, fusiform, and bifurcation aneurysms. This shows that Lauric et al. method is capable of detecting aneurysms of various types other than the most common one. While this does expand the ability of an automated aneurysm detection system to cover more cases, the test set is still quite small.

In 2020, a new approach by Hu et al. [13] was proposed. This method segments one aneurysm at a time from a 3D-DSA using a filter whose parameters are automatically found through Bayesian optimization. The shape of the aneurysm is approximated to be the same as a sphere. The Bayesian optimization finds the size and response intensity parameter that

maximizes the contrast between the aneurysm and the rest. The aneurysm is extracted using adaptive threshold and region growth. The process repeats to get the remaining aneurysms if they exist. With 165 aneurysms across 145 patients, this test set is much larger compared to the previously mentioned methods. This approach achieved a precision of 94.6% and a sensitivity of 96.4% with a false positive rate of 6.2%.

Even though all methods so far have been able to detect aneurysms as they are intended to do, they are not without any problems. Apart from what was mentioned previously, all of these methods used invasive imaging technologies. 3DRA which is the imaging modality used by Lauric et al., Cardenes et al., and Larrabide et al. requires a contrast injection for the patients [22]. CTA which is partially part of the test images by Lauric et al. needs an x-ray and iodine-based contrast injection [22]. Lastly, 3D-DSA used by Hu et al. also needs contrast injection [11]. The reason for this is to enhance the contrast between the subject of interest and the background. This helps the radiologists to distinguish the aneurysm apart from the rest of the structure, e.g., bones and tissues. Although the use of contrast agent is deemed to be safe, there are risks associated with it, including undesirable reactions like anaphylaxis and potential long-term health effects [4].

Another issue that has been previously mentioned is that these methods require assumptions for the input. Many can only detect saccular aneurysms or perfectly segmented input voxels. The bottom line is the structure of the aneurysms is simply too complex to be generalized into a single algorithm. This is the primary motivation to consider learning-based approaches. Instead of having to invent complicated procedures, developing an algorithm to recognize the patterns within the data will cover more aneurysms cases.

## 2.2 Machine Learning Methods

In 2015, a method was proposed by Jerman et al. [16] This method enhances a 3D-DSA image and calculates the blobness and vesselness of each point in the image. The points with high blobness value are candidates for the location of the aneurysm. However, vessels that are curved can also resemble a blob-like shape. If such a method were to be used directly, there would be many false positives. In order to find the right location of the aneurysm, the data is trained against manually labeled images on a random forest to eliminate false positives, i.e. points with high blobness value that are not within aneurysms. Once the false positives are eliminated, the next step is segmentation. Rays are cast from the center point of the aneurysm to the wall of the vessels. The surface that is not part of the aneurysm will have a significantly larger distance from the center points. A closed loop can be drawn around the neck of the aneurysm. The model achieved 100% sensitivity with 0.4 false positives per dataset. This method has a similar approach to the method by Hu et al. in that they both look for spherical shape as the primary aneurysm candidates on 3D-DSA. While machine learning is not directly part of the detection step, it plays an important role in recognizing outliers.

Another method for detecting and localizing intracranial aneurysms has been proposed in 2019 by Rahmany et al. [30] The method uses k-Nearest Neighbors (k-NN) to classify uniform local binary pattern (LBP) from 2D DSA images. LBP binarizes the local patches of size 3 by 3 around the pixel center. Each pixel neighbor is multiplied by squaring an increasing base (from 0 to 8) in a clockwise direction. The total sum becomes the value of that patch. A histogram can be created using the frequency of each LBP value. The method uses the window-sliding technique of varying window sizes to generate LBPs to capture small, medium, and large aneurysms. k-NN is trained on the LBPs and classified using L2 distance on 256-dimensional data.

After the initial classification, false positives are removed using the opening and closing operator and region thresholding. Additionally, the authors employed a generic Fourier descriptor (GFD). The method achieved a sensitivity of 0.91, a specificity of 0.99, and a precision of 0.63. This is the first fully automated aneurysm detection algorithm so far. Not only can it detect the presence of the aneurysm, but it can also locate it. Despite this progress, using k-NN may not be the right choice for this task. First, k-NN will have a long computational cost when the training dataset is large as it has to compare against each datapoint every single time. Second, k-NN suffers from the curse of dimensionality since the LBP histogram requires 256 dimensions to work with. This could be the reason why the initial classification produced a lot of false positives.

While there are not many traditional machine learning techniques out there, it goes to show the benefits of machine learning in practical settings. So far, however, only aneurysm detection and localization are performed with the help of machine learning. To perform a semantic segmentation task, a different kind of machine learning is better suited.

## 2.3 Segmentation Methods Using CNNs

Convolutional neural network (CNN) is a very popular neural network model. CNNs have been used in many research areas due to their efficiency and effectiveness. In 2018, a method proposed by Paoletti et al. [28] is able to classify types of land in remote sensing images with multiple spectral bands. The CNN used in this method has three convolutional layers and four fully connected (FC) layers. The method performed much better compared to multi-layer perceptrons (MLP). Another relevant contribution during this time is a method proposed by Emrah Irmak [14] in 2020. This method used a CNN with two convolutional layers and one FC layer to detect COVID-19 from X-ray images with an accuracy of 99.20%

and only one false positive case out of 125 test images.

What is interesting about CNNs is not only its ability to quickly and accurately classify an image, but that it can be extended to segment any specific objects of interest from an image. There are many methods in the literature that use a deep learning approach, however, some of them are designed to only detect and/or localize the aneurysms. The following methods perform semantic segmentation which give the results well-defined borders around the aneurysms.

There is another method proposed by Jerman et al. [17] in 2017. The authors proposed a deep learning method that is trained on a distance map to calculate the probability of any given point being a part of an aneurysm. 15 DSA images were enhanced to get the initial candidates. A 2D intra-vascular distance map is calculated by projecting rays to the wall of the vessel. By varying the direction angle, the distance, the starting point to the point along a direction with the sharpest decline in intensity, can be encoded to an image that will be trained on a CNN. The model achieved 100% sensitivity and 2.4 false positives per dataset.

As pointed out by the authors, sufficient contrast is necessary to calculate proper distance maps. In the case of time-of-flight magnetic resonance angiography (TOF-MRA) where no contrast is given to the patient, low contrast images and overlapping vessels may interfere with the intensity gradient [32].

Since the method uses distance as a primary input, it may be necessary to have large aneurysms and/or visible necks for the CNN to pick up the pattern in the distance maps. Since the authors used a small dataset and did not provide the size of the aneurysms used in training and testing, it's difficult to gauge the effectiveness of the CNN in case of small aneurysms (<3 mm).

2 years later, in 2019, Stember et al. [34] proposed a method that uses Unet-like architecture

to detect and segment maximum intensity projection (MIP) images from MRA. Additionally, the authors want to be able to determine the size of the aneurysms since it's a factor in determining the risk of rupture. The network was able to achieve 98.8% accuracy. (85 of the 86 images have some overlap between the prediction and the ground truth) The size of the aneurysm on average is different from the ground truth by 27%. The receiver operating characteristic (ROC) curve shows that the network can achieve about a 95% true positive rate with a 20% false positive rate. The area under the curve (AUC) of the ROC is 0.87.

There are a couple of drawbacks. First, it is easier for the prediction to be considered a true positive in this method compared to Chen's method. Cheng's method requires at least 30% of the prediction region to overlap with the labeled region to count as a true positive, while this method doesn't have a minimum threshold. The authors recommend using 3D images for training as they can provide more info. However, it's expected that the network needs more training data since the aneurysm will have a smaller proportion (aneurysms appear smaller in volume compared to on a plane). Second, the method uses MIP images to train the network. That is the images used are not volumetric which strips away a lot of information. The authors also acknowledged that 3D images are more suitable for training since, apart from the size, the volume of the aneurysm is also a factor in determining the risk of rupture.

Another method proposed by Podgorsak et al. [29] in 2020. The authors proposed a deep learning method for detecting and segmenting aneurysms from angiographic parametric imaging (API) images that uses digital subtraction angiography (DSA). The method uses VGG-16 as an encoder, and the decoder expands the image to its original size. In addition, radiomic features — bolus arrival time, time to peak, mean transit time, area under the time density curve, and peak height — were calculated for the aneurysms. 350 DSA images were used, 250 of which were used for training and the rest are for testing. The images

include patients who had a coiling treatment [2] and those without one. The average Jaccard index(JI)/Intersection over union(IoU) is 0.823, and the average Dice similarity coefficient (DSC) is 0.903. There is no significant difference between those with and without coiling. Because the authors aimed for a better training time, therefore, a shallow network with fewer parameters is used in exchange for some performance. The method also has the potential to gain some performance if it had taken the advantage of an information-rich 3D version of DSA.

In the same year, Chen et al. [9] proposed a method for detecting and segmenting cerebral aneurysms in 3D TOF-MRA images. The method preprocesses images before feeding then into a 3D-UNET for segmenting. The preprocessing step removes the skull and extracts only the vessels for ease of segmentation. The method was able to achieve 83.9% sensitivity with 0.86 false positives per case. However, the method struggles at detecting small aneurysms. Despite being able to detect aneurysms larger than 10 mm, the method's performance decreases when encountering aneurysms at small diameters. The method also requires some preprocessing to assist the 3D-UNET. This signifies that the network is not able to detect aneurysms when the presence of skull and soft tissues are obstructing the vessels.

Based on what has been done so far, it can be concluded that CNNs can learn to recognize the complex structure of intracranial aneurysms despite the lack of contrast enhancement. Therefore, it has the potential to be used in clinical settings. One thing that all the proposed CNNs have in common is that the depth is less than 30 layers.

[2] refers to the process of inserting a metal wire into the aneurysms to make the blood within coagulate and prevent blood flow into the site [41].

## 2.4 Motivation for Pursuing a Very Deep Network

The number of layers in a deep learning model is one indication that can tell how complex the data the model tries to learn is. Each convolutional layer added gives a model more parameters to capture the pattern of the data. In 2017, Li et al. [24] proposed a convolutional neural network based on the U-Net architecture for segmenting land-sea satellite images. The network incorporates Plus connections, and the network is about twice as deep as the U-Net. The result shows a higher precision, recall, and F-1 score than U-Net and SegNet.

Qualitatively, the result looks much better than the U-Net and SegNet. It can handle fine details, and land spaces that are closed together are well separated.

The contrast of the training images is high. Unlike medical images which are often low contrast and grayscale, the land and sea can easily be distinguished.

All sample images have a clear borderline between the land and the sea. This is because the land is mostly man-made structures. There are no structures with ambiguous borderlines like islands with a shallow shore or ocean sinkholes.

However, this does not mean a model should have as many layers as possible. A model that is too complex introduces many undesirable side effects. In the case of deep learning, more computation time is required to update all the additional parameters. This also means that the gradient propagation path from the output to the input is longer, and this makes the model more susceptible to vanishing and exploding gradient. A long chain of multiplication means that a number slightly above or below 1 will gradually converge to infinity or 0 respectively. Lastly, the model is more prone to overfitting since it may learn too well. Despite all these drawbacks, there are reasons to not completely disregard a very deep network just yet.

In 2015, Shuying Liu and Weihong Deng [25] uses a modified VGG-16 trained on CIFAR-10, a relatively small dataset compared to ImageNet used by other networks. The authors suggest that strong dropout and batch normalization are keys to avoiding overfitting in a very deep network. The network converges faster. The result shows that the modified VGG-16 has a lower error rate than the baseline network with fewer layers. The paper shows promise that a deep network is better than shallow networks provided that the deep network has proper hyperparameters to avoid overfitting.

The primary interest on this thesis is how very deep networks perform in a segmentation task. Specifically, on U-Net architecture. Liu et al. work shows that it is possible to learn deep network on a small dataset. However, the work was for classification task and the depth of the network is relatively small compared to more recently developed network.

# Chapter 3

# Methodology

To better understand how deep convolutional neural networks behave, two architectures are created based on a well-known deep learning architecture. The first is a standard model which is kept to be very similar to the original models. The second model is modified to have more layers which substantially increases the number of parameters. The idea is to have the first model serves as control when compared with a more complex model. Both models are then tested under the same configuration and hyperparameters.

Next, different values or choices for each selected hyperparameter will be tested on each model. These hyperparameters and their choices are selected are based on their likelihood to have an impact on the difference in model complexity. This likelihood is determined based on the recent research papers and the author's hypothesis.

This setup is not a reflection of what practical neural networks are supposed to be. The goal is to understand the effects of each hyperparameter choice on standard and deep networks. Therefore, the training time is not a concern unlike in the paper by Liu and Deng [25] and relative differences in results are the focus of this experiment.

The subsequent sections go into more details on how the tests are set up.

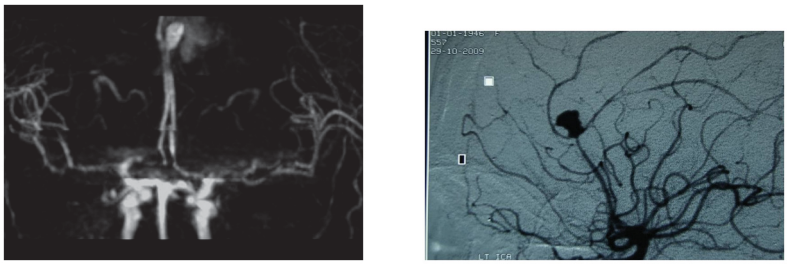

Figure 3.1: TOF-MRA scan (left) and DSA scan (right) capturing an intracranial aneurysm. The aneurysm in the TOF-MRA scan is clouded with hemorrhages unlike in the DSA scan according to the original description. The image and the description are provided by [19].

## 3.1 Dataset

The dataset used in this experiment comes from Aneurysm Detection And segMentation Challenge 2020 or ADAM. ADAM consists of 113 3D TOF-MRA brain scans. As mentioned in the previous section, 3D TOF-MRA is a non-invasive imaging modality which comes at a cost of quality. An example is shown in Figure 3.1. Each case has a 3D TOF-MRA image and a structural image with manually segmented labels. 93 of the scans contain at least one untreated and unruptured intracranial aneurysm. The other 20 cases have no aneurysms. Of the 93 cases with aneurysms, 35 cases have a follow-up scan and 23 unique cases [36].

Other information provided with the dataset including machine parameters (i.e., modality, magnetic field strength, repetition time, and echo time), the location of the untreated aneurysms, its center of mass and radius, and the transformation parameters for aligning the structural image to the TOF-MRA image is not used in this experiment [36].

The use of this dataset has been permitted by the main organizer — Kimberlet Timmins from the Image Sciences Institutes, University Medical Center, Utrecht, Netherlands.

## 3.2 Data Preprocessing

To prepare this dataset, the first step starts at the 3D TOF-MRA images that have been preprocessed with N4 bias field correction. This correction is performed by ANTs (Advanced Normalization Tools) and is provided as part of the ADAM dataset [36]. This correction should reduce the low-frequency signal that blurred MRI images resulting in shaper images [18]. The corrected images are then resized and normalized to 128x128x128 images with values ranging from 0 to 1.

There are three labels per pixel in the original dataset — Background, Aneurysm, and Treated Aneurysm. Because treated aneurysms are not relevant in this experiment, pixels labeled as treated aneurysms will be recognized as background pixels. The labels are then resized to have the same resolution as the input images.

The images are shuffled before creating a training set and test set. The training set is shuffled once more after the augmentation step.

## 3.3 Data Augmentation

As with many medical imaging datasets, ADAM is a relatively small dataset compared to other datasets with common objects like CIFAR. To compensate for this disadvantage, data augmentation is performed to increase the size of the training dataset.

A random flip, random rotation, and random translation are performed on the training dataset. This increases the training set from 85 to 170. A random crop is then performed on all 170 images increasing the size to 340 images.

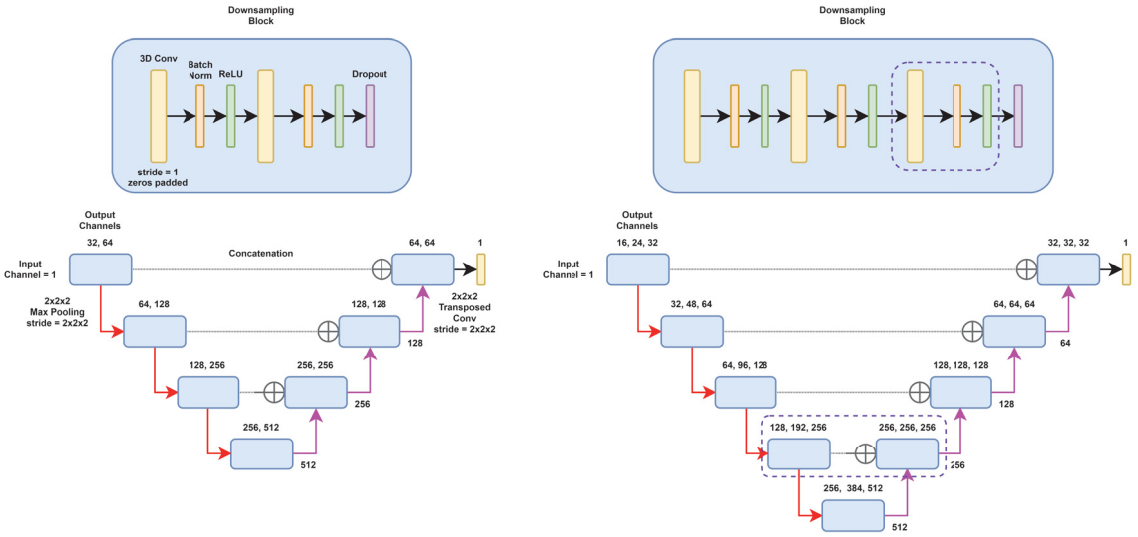


Figure 3.2: Base U-Net (left) and Deep U-Net (right) architecture represented by block diagrams

## 3.4 Architectures

The architecture that will be used in this experiment is 3D U-Net. This is because U-Net proposed by Ronneberger et al. in 2015 [31] is among the most popular fully convolutional neural networks for semantic segmentation tasks. The network is easy to understand and symmetric. This means that extending the network would be easier and intuitive to understand.

To be able to segment aneurysms from volumetric TOF-MRA scans, a variant of U-Net called 3D U-Net [42] is used. There will be two versions of the 3D U-Net used in this experiment. The first is kept to be similar to the original 3D U-Net. This model is the baseline model that will be used to compare to the second model which is modified to have more layers. Only small changes have been made to better fit the type of datasets used. The following subsections go into more detail on these two architectures.

### 3.4.1 Base U-Net

The base 3D U-Net, as shown in Figure 3.2, has an ”analysis” and ”synthesis” path like the original. The analysis path has 4 downsampling blocks. Each block contains two 3x3x3 convolutional layers and a 2x2x2 max-pooling layer. Each is followed by a batch normalization layer, a ReLU (rectified linear unit) activation function, and a dropout layer respectively. The synthesis path consists of 4 2x2x2 transposed convolutional layers. Each is connected to a convolutional block same as downsampling blocks. The input is a concatenation between the output of the transposed convolutional layers and the output of the downsampling block of the same resolution. The number of kernels starting at 32 is double after each downsampling block.

By default, this model uses the He normal weight initialization on all layers and a stochastic gradient descent optimizer. All dropout layers have a 0% drop rate and the default loss function is the weighted binary cross-entropy. These hyperparameters are kept to be mostly the same as the original 3D U-Net.

Unlike the original, this network is modified to have the same input and output resolution. To achieve this, each convolutional layer’s output is zero-padded so that the resolution of the input and output are the same.

Also, because this is a pixel-wise binary classification task, the last convolutional layer has only one kernel as an output followed by a softmax activation function to convert logits to probabilities. That is, each pixel has only one node for prediction unlike the original where there are 3 kernels for prediction.

The added dropout layers which are not a part of the original paper will be set to zero by default which has the same effect as not having one present. The reason for this addition will be elaborated further in section 3.6.1.

There are 18 layers with learnable parameters resulting in a total of 19,078,337 parameters. Of the total parameters, 19,073,665 parameters are trainable.

### 3.4.2 Deep U-Net

This architecture, shown in Figure 3.2, is modified from the base U-Net to increase the number of layers and number of learnable parameters within the network. All hyperparameters and the overall architecture are the same except there is an additional resolution step and another convolutional layer added to the downsampling block. The number of kernels starts at 16. The output of the middle convolutional layer has the number of kernels equal to the average between the number of output kernels in the first layer and the last (third) layer. For example, the numbers of kernels in the first downsampling block are 16, 24, and 32. The hyperparameters are the same as the base U-Net by default.

This results in a 32-layer deep network and a total of 27,512,009 parameters (27,504,665 trainable parameters) — an increase of 44.2%.

## 3.5 Training and Testing

### 3.5.1 Configuration

The dataset is separated into two sets. 75% are for the training set and 25% for the testing set. 20% of the training set is used for validation. This means, after data augmentation, there are 272 images for training, 68 images for validation, and 28 images for testing. The global batch size is 12 for training and 4 for testing. The maximum training epoch is 200.

The initial learning rate is found via the Hyperband tuner [23]. The search space depends

on the optimizer used in each model. More details in the section 3.6.3 later on.

To prevent the model from overfitting, the training will stop before reaching epoch 200 if the validation loss is not improving for 5 consecutive epochs.

### 3.5.2 Training Hardware

All models will be tested on TinkerCliffs HPC (High Performance Computing) nodes — a Virginia Tech's ARC (Advanced Research Computing) cluster. Each run uses a 128-core AMD EPYC 7742 and four NVIDIA A100-80G GPUs [1]. More information regarding Tinkercliff in the Appendix.

The primary season for using an HPC for training is due to the lack of video memory available in consumer GPUs which is required for loading training volumetric images and storing model weights. Optimization for reducing video memory consumption is possible. For example, reducing precision of the model weights from 32-bit floating-point to 16-bit floating-point can improve both training time and memory usage, but it may also impact the performance.

## 3.6 Hyperparameters

Apart from the original hyperparameter settings, there are other choices of hyperparameters that may improve the performance or help overcome problems associated with deep neural networks. These hyperparameters are dropout, loss function, optimizer, and weight initialization.

### 3.6.1 Dropout

According to Liu and Deng [25], having a strong dropout help reduce overfitting in image classification task in deep networks. To test whether this claim scales up to the image segmentation task, there are three settings of dropout — 0% (default), 15%, 30%. A dropout layer is placed at the end of each downsampling block and all dropout layers have the same setting.

### 3.6.2 Loss Function

The loss function is an important part in determining the correctness of the model prediction. The loss functions must work even with highly imbalanced datasets and they must produce the gradient that can be backpropagated through a deep network. Apart from cross-entropy loss which is the standard, Dice loss is put into consideration as it is commonly used [15]. And based on the results and suggestion by Jadon [15], Focal Tversky loss will also be tested.

#### Weighted Binary Cross-Entropy Loss

Weighted binary cross-entropy (WBC) loss is defined as

$$WBC_L = \sum_{i=1}^{N} \frac{-(\beta \cdot y_i \cdot log(\hat{y}_i + \epsilon) + (1 - y_i) \cdot log(1 - \hat{y}_i + \epsilon))}{N}$$

where $N$ is the total number of pixels in a volumetric image. $\hat{y}_i$ is the predicted probability for pixel $i$ being an aneurysm and $y$ is the ground truth for pixel $i$. $\beta$ is a weight constant and is set to 3. $\epsilon$ is a constant added to avoid negative infinity in case the predicted probability is equal to 0 inside the natural log function. $\epsilon = 10^{-6}$.

The definition is based on the equation used by Yuan et al. [39].

### Dice Loss

The definition of Dice loss is the same as the paper by Abraham and Khan [3]. It is defined as

$$D_L = 2 - \frac{\sum_{i=1}^{N} \hat{y}_i \cdot y_i + \epsilon}{\sum_{i=1}^{N} \hat{y}_i + y_i + \epsilon} - \frac{\sum_{i=1}^{N} (1 - \hat{y}_i) \cdot (1 - y_i) + \epsilon}{\sum_{i=1}^{N} (1 - \hat{y}_i) + (1 - y_i) + \epsilon}$$

$\epsilon$ is added to avoid division by zero and is set to 0.01.

Unlike in the WBC loss, Dice loss does not penalize exponentially. (The cost tends toward infinity as $x$ in $-log(x)$ approaches 0.)

### Focal Tversky Loss

The definition of Focal Tversky loss is the same as the original paper by Abraham and Khan [3]. The Tversky similarity index is defined as

$$TI_c = \frac{\sum_{i=1}^{N} \hat{y}_{ci} \cdot y_{ci} + \epsilon}{\sum_{i=1}^{N} \hat{y}_{ci} \cdot y_{ci} + \alpha \cdot \sum_{i=1}^{N} \hat{y}_{\bar{c}i} \cdot y_{ci} + \beta \cdot \sum_{i=1}^{N} \hat{y}_{ci} \cdot y_{\bar{c}i} + \epsilon}$$

where $\hat{y}_{ci}$ is the probability of pixel $i$ being a part of an aneurysm and $\hat{y}_{\bar{c}i} = 1 - \hat{y}_{ci}$ is the probability of pixel $i$ not being a part of an aneurysm.

The Focal Tversky loss is defined as

$$FT_L = (1 - TI_c)^{1/\gamma} + (1 - TI_{\bar{c}})^{1/\gamma}$$

The constants are set to the following $\alpha = 0.7, \beta = 0.3, \gamma = \frac{4}{3}, \epsilon = 0.01$. These values are chosen to be the same as the original except for $\epsilon$.

As pointed out in the original paper, Dice loss penalizes false negatives and false positives equally which is not a desirable effect if getting either the correct positive or negative labels are more important than the other. Focal Tversky loss fixes this issue by adding weights (i.e., $\alpha$, $\beta$). This loss can also focus on the incorrect predictions more by adding the constant, $\gamma$. When $\gamma > 1$, the loss will be impacted less if the model predictions are already close to ground truth.

### 3.6.3 Optimizer

Although various kinds of optimizers are not tested by Liu and Deng [25], the authors mention that fast convergence is important in improving the error rate in deep models. Many optimizers in the past decade have been improving to better adjust the learning rate and therefore improve the convergence performance.

In addition to the standard stochastic gradient descent (SGD), Adam [20] and Adadelta [40] are included for the testing. These two optimizers have been used in deep neural networks for aneurysm segmentation in [17] [34] [29] [9].

The initial learning rate will be different depending on the model and other configurations. To make sure every configuration gets to produce the best result they could, the initial learning rate is set based on the best value found during the learning rate search using Hyperband tuner [23].

Each optimizer will have a different search space due to how the learning rate is adjusted and calculated during training. Each optimizer has an option to choose its initial learning rate from the following sets.

$$SGD_{lr} = \{a \times 10^{-b} : a \in \{3, 7\}, b \in \{1, 2, 3, 4, 5\}\}$$

$$Adam_{lr} = \{a \times 10^{-b} : a \in \{3, 7\}, b \in \{5, 6, 7\}\}$$

$$Adadelta_{lr} = \{a \times 10^{-b} : a \in \{3, 7\}, b \in \{1, 2, 3, 4, 5\}\}$$

The range for SGD is based on an article by Brownlee [7] and the range for Adam is adjusted to work on both architectures based on the range of SGD.

Additionally, the SGD optimizer will have a learning rate scheduler where the learning rate is halved when the training loss remains the same for more than one epoch. This scheduler does not apply to Adam or Adadelta.

### 3.6.4 Weight Initialization

How the weight of the layers is initialized can affect the convergence during training [5]. In this experiment, there are four weight initialization methods used — He normal, He uniform, Xavier normal, and Xavier uniform. These initialization methods are common and have been used in many CNN models in remote sensing tasks [5].

## 3.7 Metrics

The model keeps track of five metrics including the number of true positives, true negatives, false positives, false negatives, and AUC-ROC. The threshold is set to 0.5.

In addition to the aforementioned metrics which are tracked during the training and testing step, the accuracy, sensitivity, and specificity are also calculated when testing. These specific

metrics are used to gauge the model performance. Each metric gives a different insight into how the model performs. For example, the sensitivity which can tell how well the model recognizes the true positive pixels and AUC-ROC which can tell how well the model learns are important.

Due to the highly-imbalanced nature of the dataset, an F-1 score will also be calculated per an insight given in a textbook by Murphy [26]. If either recall or precision performs poorly, then the model will have a low F-1 score. Thus, F-1 provides a better indication than averaging recall and precision or using one of them by themselves.

These metrics are defined as follows:

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

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

Specificity $= \frac{TN}{TN+FP}$

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

F-1 score $= \frac{2 \times Precision \times Recall}{Precision+Recall}$

TP, TN, FP, FN are the total number of pixels in the testing set classified as true positive, true negative, false positive, and false negative respectively.

## 3.8 Implementation Details

All the code used in this experiment is implemented in Python (approximately 400 lines) using Tensorflow [2], Keras [10], and KerasTuner [27]. Additional libraries including Numpy [12], Scikit-image [37], and Nibabel [6] for data processing.

# Chapter 4

# Results

In this section, all the results from testing 20 models will be shown. These include the raw performance and the metrics outlined in the Methodology section. Each variation of hyper-parameter will have plots of training and validation loss for the Base and Deep model and a testing result in a table. This table includes the testing loss, AUC-ROC, accuracy, sensitivity, specificity, F-1 score, and the members of the confusion matrix — the true positive, true negative, false positive, and false negative pixel counts in the testing set. The sum of all pixel counts in the testing set is equal to 58,720,256 pixels. (The number of pixels in one image, $128 \times 128 \times 128 = 2,097,152$, times the number of images in the testing set, 28)

## 4.1 Baseline

This is the result of the models with no changes in the default configuration which served as a baseline performance when compared with other models. These two models have a 0% dropout rate, used the weighted binary classification loss, used stochastic gradient descent as an optimizer, and initialized the weights with He normal.

In Figure 4.1, the training stopped at epoch 27 for the Base model with the training loss of $3.24 \times 10^{-4}$ and the validation loss of $3.47 \times 10^{-4}$. The training stopped at epoch 38 for the Deep model with the training loss of $4.16 \times 10^{-4}$ and the validation loss of $3.53 \times 10^{-4}$.

| | Base | Deep |
|---|---|---|
| Testing Loss | 0.000345 | 0.000542 |
| True Positives | 5 | 0 |
| True Negatives | 58,708,344 | 58,708,560 |
| False Positives | 246 | 30 |
| False Negatives | 11,663 | 11,668 |
| AUC-ROC | 0.562 | 0.568 |
| Accuracy | 99.979719% | 99.980078% |
| Sensitivity | 0.042852% | 0.000000% |
| Specificity | 99.999581% | 99.999949% |
| F-1 score | 0.083900% | N/A |

Table 4.1: Testing result for the Base and Deep U-Net model without hyperparameter modification

The N/A values in Table 4.1 and in the subsequent tables indicate that the values are not available due to division by zero. This is due to either both the precision and recall are zero or $TP + FP = 0$ or $TP + FN = 0$.

## 4.2 Dropout

### 4.2.1 15% dropout rate

| | Base | Deep |
|---|---|---|
| Testing Loss | 0.000303 | 0.000357 |
| True Positives | 0 | 0 |
| True Negatives | 58,708,584 | 58,708,588 |
| False Positives | 1 | 0 |
| False Negatives | 11,668 | 11,668 |
| AUC-ROC | 0.542 | 0.510 |
| Accuracy | 99.980128% | 99.980130% |
| Sensitivity | 0.000000% | 0.000000% |
| Specificity | 99.999998% | 100.000000% |
| F-1 score | N/A | N/A |

Table 4.2: Testing result for the Base and Deep U-Net model with 15% dropout

In Figure 4.2 where the models have a 15% dropout rate, the training stopped at epoch 25 for the Base model with the training loss of $3.77 \times 10^{-4}$ and the validation loss of $2.54 \times 10^{-4}$. The training stopped at epoch 35 for the Deep model with the training loss of $4.46 \times 10^{-4}$ and the validation loss of $6.18 \times 10^{-4}$.

### 4.2.2 30% dropout rate

| | Base | Deep |
|---|---|---|
| Testing Loss | 0.000242 | 0.000292 |
| True Positives | 0 | 0 |
| True Negatives | 58,708,588 | 58,708,588 |
| False Positives | 0 | 0 |
| False Negatives | 11,668 | 11,668 |
| AUC-ROC | 0.501 | 0.506 |
| Accuracy | 99.980130% | 99.980130% |
| Sensitivity | 0.000000% | 0.000000% |
| Specificity | 100.000000% | 100.000000% |
| F-1 score | N/A | N/A |

Table 4.3: Testing result for the Base and Deep U-Net model with 30% dropout

In Figure 4.3 where the models have a 30% dropout rate, the training stopped at epoch 18 for the Base model with the training loss of $4.30 \times 10^{-4}$ and the validation loss of $2.91 \times 10^{-4}$. The training stopped at epoch 24 for the Deep model with the training loss of $5.19 \times 10^{-4}$ and the validation loss of $2.97 \times 10^{-4}$.

Compared with the baseline results, the testing loss decreases in both the Base and Deep model the higher the dropout rate. In the Base model, the testing loss decreases from 0.000345 to 0.000303 with a 15% dropout rate and down to 0.000242 with a 30% dropout rate. The change is more dramatic in the Deep model. The testing loss decreases from 0.000542 to 0.000357 with a 15% dropout rate and down to 0.000292 with a 30% dropout rate.

## 4.3 Loss Function

### 4.3.1 Dice Loss

| | Base | Deep |
|---|---|---|
| Testing Loss | 0.375195 | 0.375465 |
| True Positives | 1,335 | 1,712 |
| True Negatives | 58,549,612 | 58,351,856 |
| False Positives | 158,978 | 356,733 |
| False Negatives | 10,333 | 9,956 |
| AUC-ROC | 0.690 | 0.652 |
| Accuracy | 99.711665% | 99.375532% |
| Sensitivity | 11.441550% | 14.672609% |
| Specificity | 99.729208% | 99.392367% |
| F-1 score | 1.552497% | 0.925123% |

Table 4.4: Testing result for the Base and Deep U-Net model with Dice loss

In Figure 4.4 where the models used Dice loss, the training stopped at epoch 33 for the Base model with the training loss of $3.7498 \times 10^{-1}$ and the validation loss of $3.7485 \times 10^{-1}$. The training stopped at epoch 26 for the Deep model with the training loss of $3.7481 \times 10^{-1}$ and the validation loss of $3.7495 \times 10^{-1}$.

### 4.3.2 Focal Tversky Loss

In Figure 4.5 where the models used Focal Tversky loss, the training stopped at epoch 14 for the Base model with the training loss of $2.4998 \times 10^{-1}$ and the validation loss of $2.5003 \times 10^{-1}$. The training stopped at epoch 14 for the Deep model with the training loss of $2.4643 \times 10^{-1}$ and the validation loss of $2.5081 \times 10^{-1}$.

Compared with the WBC loss used in the baseline models. The models that use Dice loss and Focal Tversky Loss have a higher sensitivity. In the Base model, the model that uses

| | Base | Deep |
|---|---|---|
| Testing Loss | 0.250040 | 0.247460 |
| True Positives | 0 | 1,046 |
| True Negatives | 58,708,588 | 58,664,352 |
| False Positives | 0 | 44,237 |
| False Negatives | 11,668 | 10,622 |
| AUC-ROC | 0.500 | 0.616 |
| Accuracy | 99.980130% | 99.906576% |
| Sensitivity | 0.000000% | 8.964690% |
| Specificity | 100.000000% | 99.924650% |
| F-1 score | N/A | 3.673333% |

Table 4.5: Testing result for the Base and Deep U-Net model with Focal Tversky loss

Dice loss outperformed the other two models with a sensitivity of 11.44% compared with 0.00% in the model with WBC loss and 8.96% in the model with Focal Tversky loss. In the Deep model, the sensitivity is higher. The sensitivities for Deep models using WBC loss, Dice loss, and Focal Tversky loss are 0.04%, 14.67%, and 8.96% respectively.

Note that the increase in true positives also comes with an increase in false positives.

## 4.4 Optimizer

### 4.4.1 Adam

In Figure 4.6 where the models used Adam Optimizer, the training stopped at epoch 117 for the Base model with the training loss of $2.43 \times 10^{-4}$ and the validation loss of $3.45 \times 10^{-4}$. The training stopped at epoch 50 for the Deep model with the training loss of $36.130 \times 10^{-4}$ and the validation loss of $35.613 \times 10^{-4}$.

| | Base | Deep |
|---|---|---|
| Testing Loss | 0.000335 | 0.004754 |
| True Positives | 4 | 4 |
| True Negatives | 58,707,296 | 58,668,160 |
| False Positives | 1,288 | 40,430 |
| False Negatives | 11,664 | 11,664 |
| AUC-ROC | 0.692 | 0.387 |
| Accuracy | 99.977943% | 99.911284% |
| Sensitivity | 0.034282% | 0.034282% |
| Specificity | 99.997806% | 99.931134% |
| F-1 score | 0.061728% | 0.015354% |

Table 4.6: Testing result for the Base and Deep U-Net model with Adam optimizer

| | Base | Deep |
|---|---|---|
| Testing Loss | 0.000394 | 0.000815 |
| True Positives | 1 | 3 |
| True Negatives | 58,708,564 | 58,708,552 |
| False Positives | 25 | 32 |
| False Negatives | 11,667 | 11,665 |
| AUC-ROC | 0.595 | 0.652 |
| Accuracy | 99.980089% | 99.980080% |
| Sensitivity | 0.008570% | 0.025711% |
| Specificity | 99.999957% | 99.999945% |
| F-1 score | 0.017103% | 0.051269% |

Table 4.7: Testing result for the Base and Deep U-Net model with Adadelta optimizer

### 4.4.2 Adadelta

In Figure 4.7 where the models used Adadelta optimizer, the training stopped at epoch 25 for the Base model with the training loss of $3.72 \times 10^{-4}$ and the validation loss of $3.22 \times 10^{-4}$. The training stopped at epoch 53 for the Deep model with the training loss of $7.32 \times 10^{-4}$ and the validation loss of $9.40 \times 10^{-4}$.

The AUC-ROC values are higher in the Deep model when compared to the Base model on all except for the models that use the Adam optimizer. The AUC-ROC for the Base model with Adam optimizer decreases from 0.69 to 0.39 in the Deep model.

## 4.5 Weight Initialization

### 4.5.1 Xavier Normal

| | Base | Deep |
|---|---|---|
| Testing Loss | 0.000319 | 0.000327 |
| True Positives | 0 | 0 |
| True Negatives | 58,708,584 | 58,708,588 |
| False Positives | 4 | 0 |
| False Negatives | 11,668 | 11,668 |
| AUC-ROC | 0.589 | 0.545 |
| Accuracy | 99.980123% | 99.980130% |
| Sensitivity | 0.000000% | 0.000000% |
| Specificity | 99.999993% | 100.000000% |
| F-1 score | N/A | N/A |

Table 4.8: Testing result for the Base and Deep U-Net model with Xavier normal weight initialization

In Figure 4.8 where the models initialized weights with Xavier normal, the training stopped at epoch 27 for the Base model with the training loss of $3.03 \times 10^{-4}$ and the validation loss of $2.89 \times 10^{-4}$. The training stopped at epoch 22 for the Deep model with the training loss of $3.71 \times 10^{-4}$ and the validation loss of $3.64 \times 10^{-4}$.

### 4.5.2 Xavier Uniform

In Figure 4.9 where the models initialized weights with Xavier uniform, the training stopped at epoch 33 for the Base model with the training loss of $2.62 \times 10^{-4}$ and the validation loss of $2.63 \times 10^{-4}$. The training stopped at epoch 25 for the Deep model with the training loss of $3.94 \times 10^{-4}$ and the validation loss of $3.68 \times 10^{-4}$.

| | Base | Deep |
|---|---|---|
| Testing Loss | 0.000338 | 0.000379 |
| True Positives | 0 | 0 |
| True Negatives | 58,708,588 | 58,708,588 |
| False Positives | 0 | 0 |
| False Negatives | 11,668 | 11,668 |
| AUC-ROC | 0.580 | 0.613 |
| Accuracy | 99.980130% | 99.980130% |
| Sensitivity | 0.000000% | 0.000000% |
| Specificity | 100.000000% | 100.000000% |
| F-1 score | N/A | N/A |

Table 4.9: Testing result for the Base and Deep U-Net model with Xavier uniform weight initialization

| | Base | Deep |
|---|---|---|
| Testing Loss | 0.000385 | 0.000470 |
| True Positives | 2 | 2 |
| True Negatives | 58,708,452 | 58,708,392 |
| False Positives | 135 | 194 |
| False Negatives | 11,666 | 11,666 |
| AUC-ROC | 0.544 | 0.563 |
| Accuracy | 99.979903% | 99.979803% |
| Sensitivity | 0.017141% | 0.017141% |
| Specificity | 99.999770% | 99.999670% |
| F-1 score | 0.033884% | 0.033715% |

Table 4.10: Testing result for the Base and Deep U-Net model with He uniform weight initialization

### 4.5.3 He Uniform

In Figure 4.10 where the models initialized weights with He uniform, the training stopped at epoch 33 for the Base model with the training loss of $3.01 \times 10^{-4}$ and the validation loss of $3.05 \times 10^{-4}$. The training stopped at epoch 26 for the Deep model with the training loss of $4.03 \times 10^{-4}$ and the validation loss of $5.70 \times 10^{-4}$.

The models whose weights are initialized with He uniform have higher false positives than

the rest of the initialization methods. The false positives increase from 135 in the Base model to 194 in the Deep model.

There is also a slight decrease in AUC-ROC when using the Xavier normal initialization method from 0.59 in the Base model to 0.54 in the Deep model.

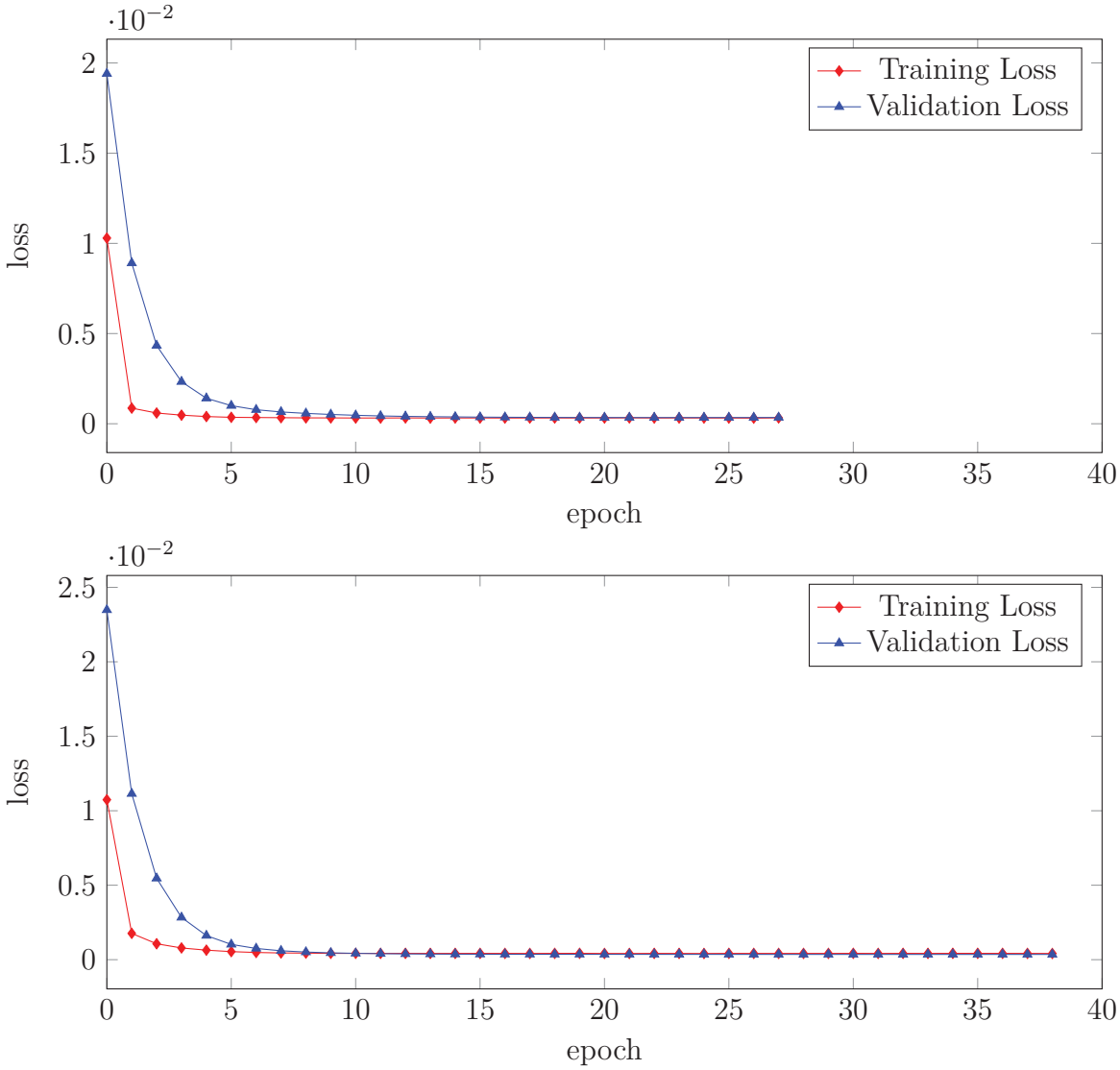


Figure 4.1: The training and validation loss for Base U-Net (top) and Deep U-Net (bottom) without hyperparameter modification

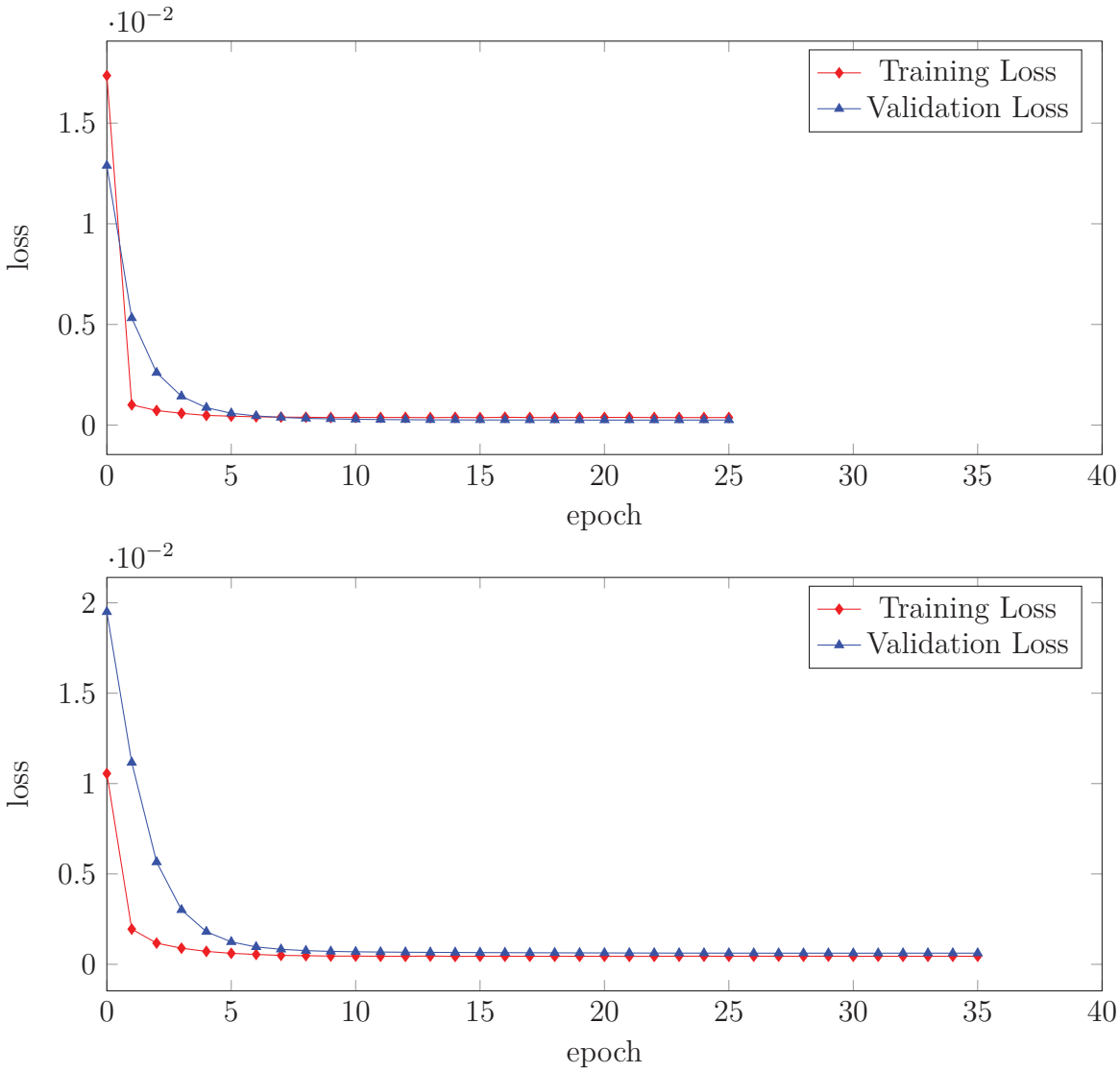


Figure 4.2: The training and validation loss for Base U-Net (top) and Deep U-Net (bottom) with 15% dropout

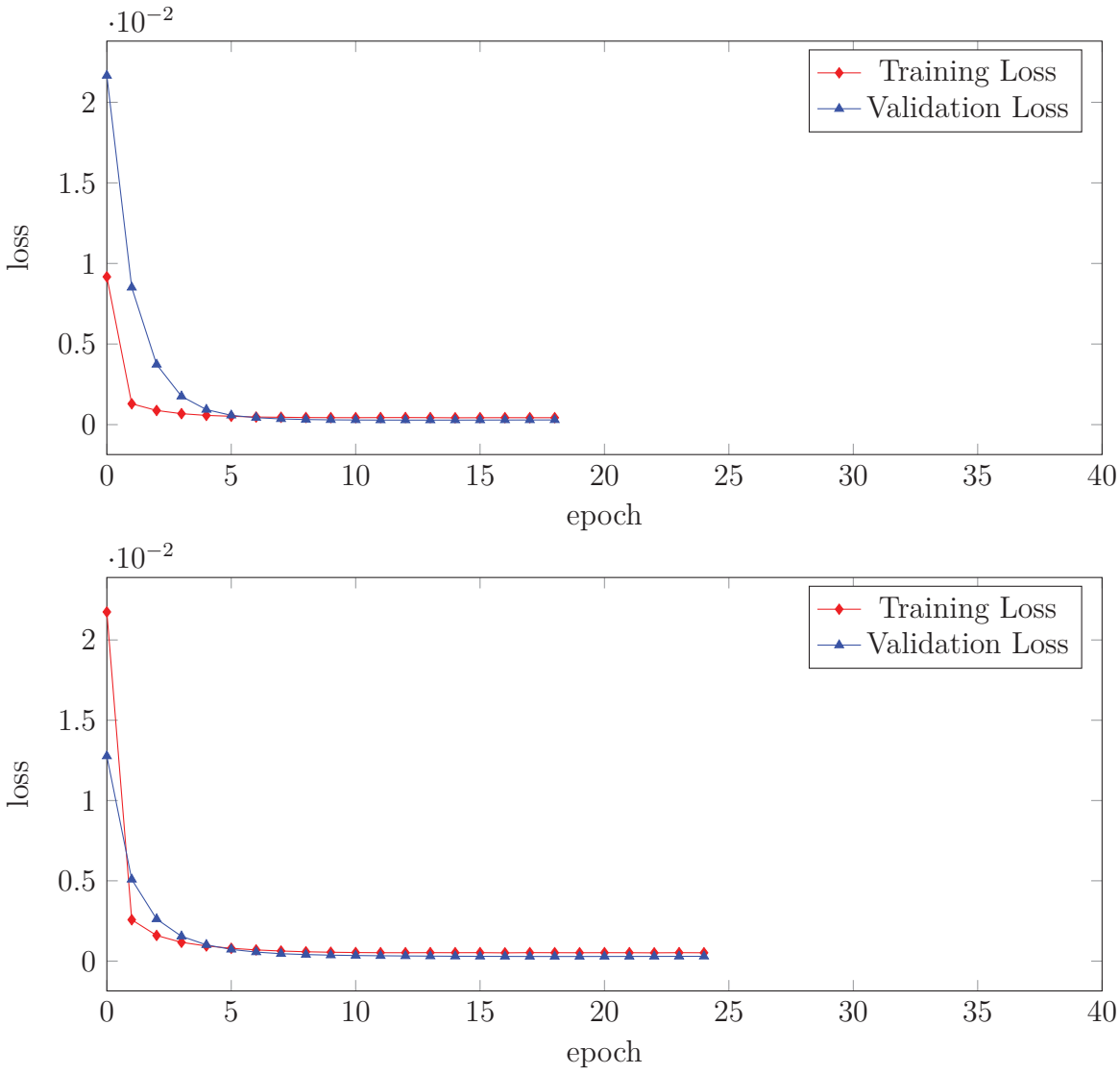


Figure 4.3: The training and validation loss for Base U-Net (top) and Deep U-Net (bottom) with 30% dropout

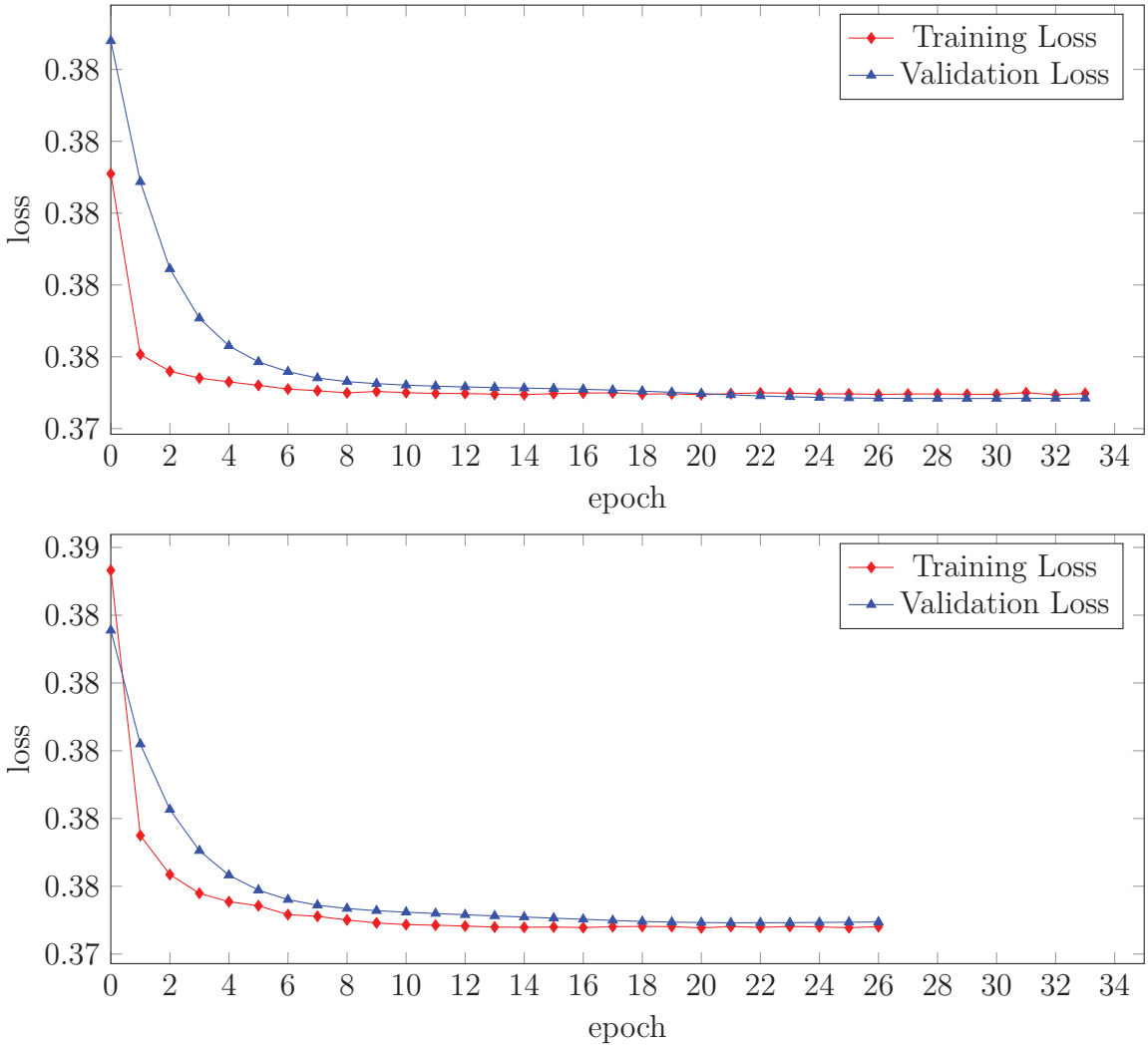


Figure 4.4: The training and validation loss for Base U-Net (top) and Deep U-Net (bottom) with Dice loss

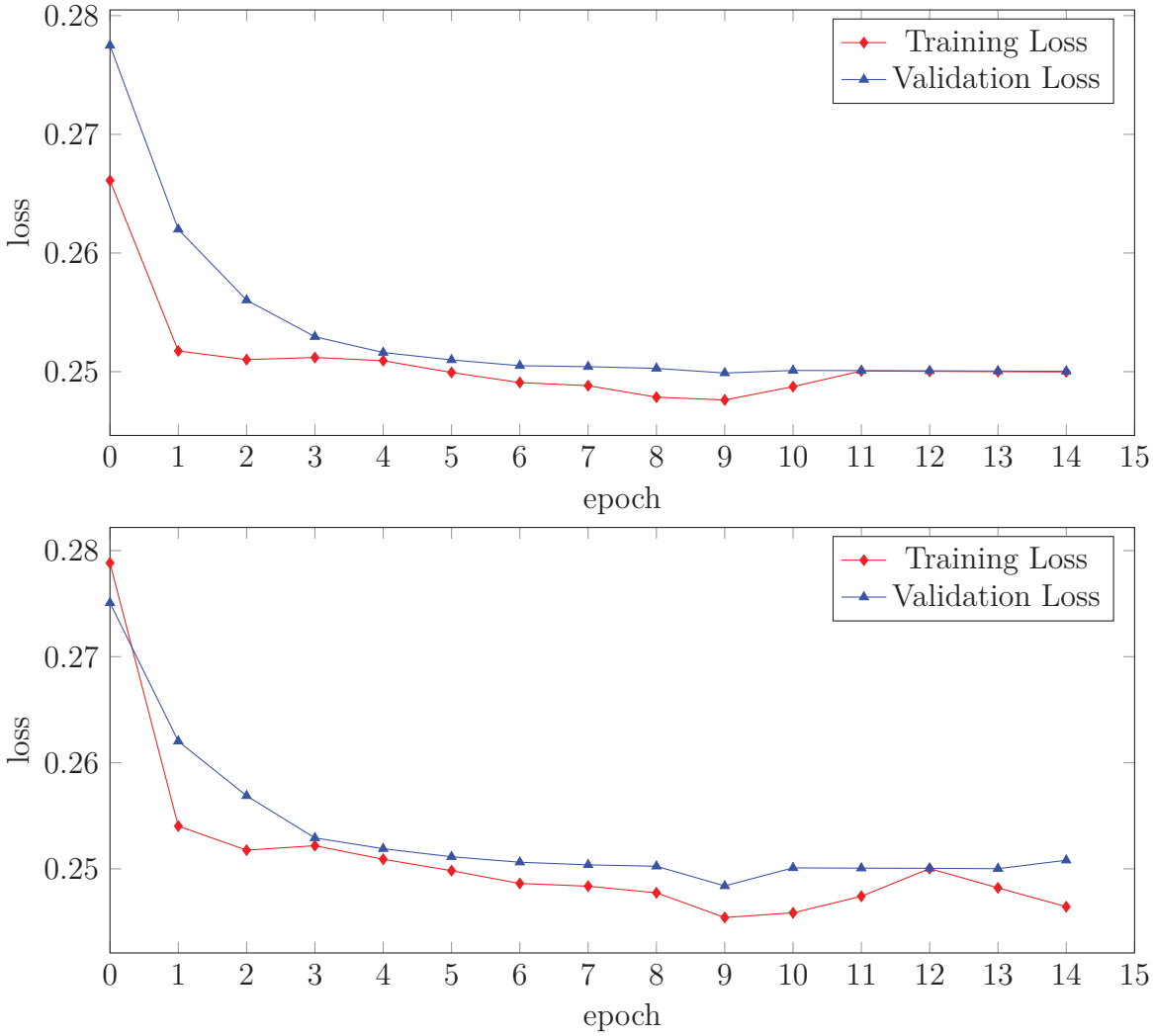


Figure 4.5: The training and validation loss for Base U-Net (top) and Deep U-Net (bottom) with Focal Tversky loss

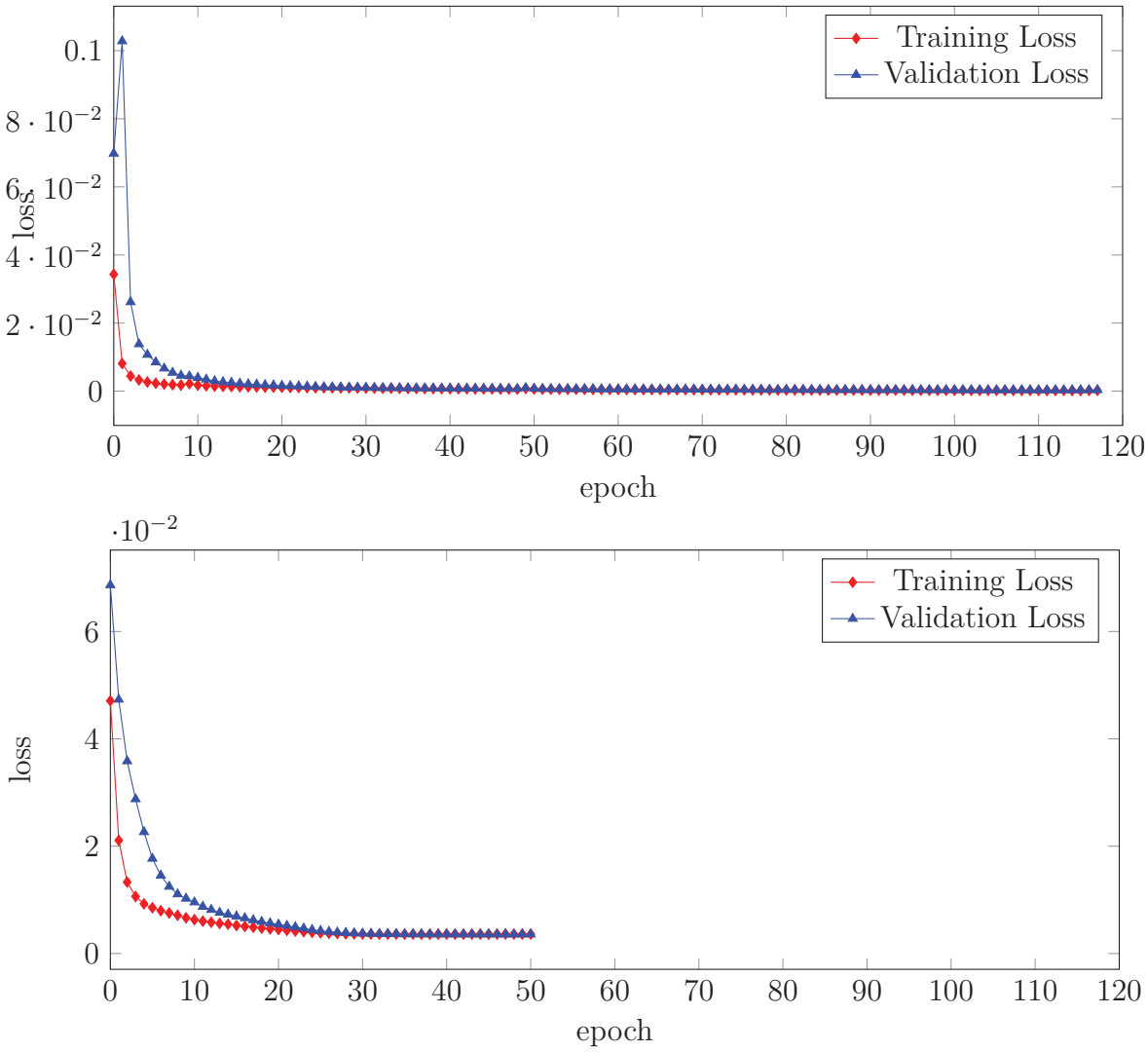


Figure 4.6: The training and validation loss for Base U-Net (top) and Deep U-Net (bottom) with Adam optimizer

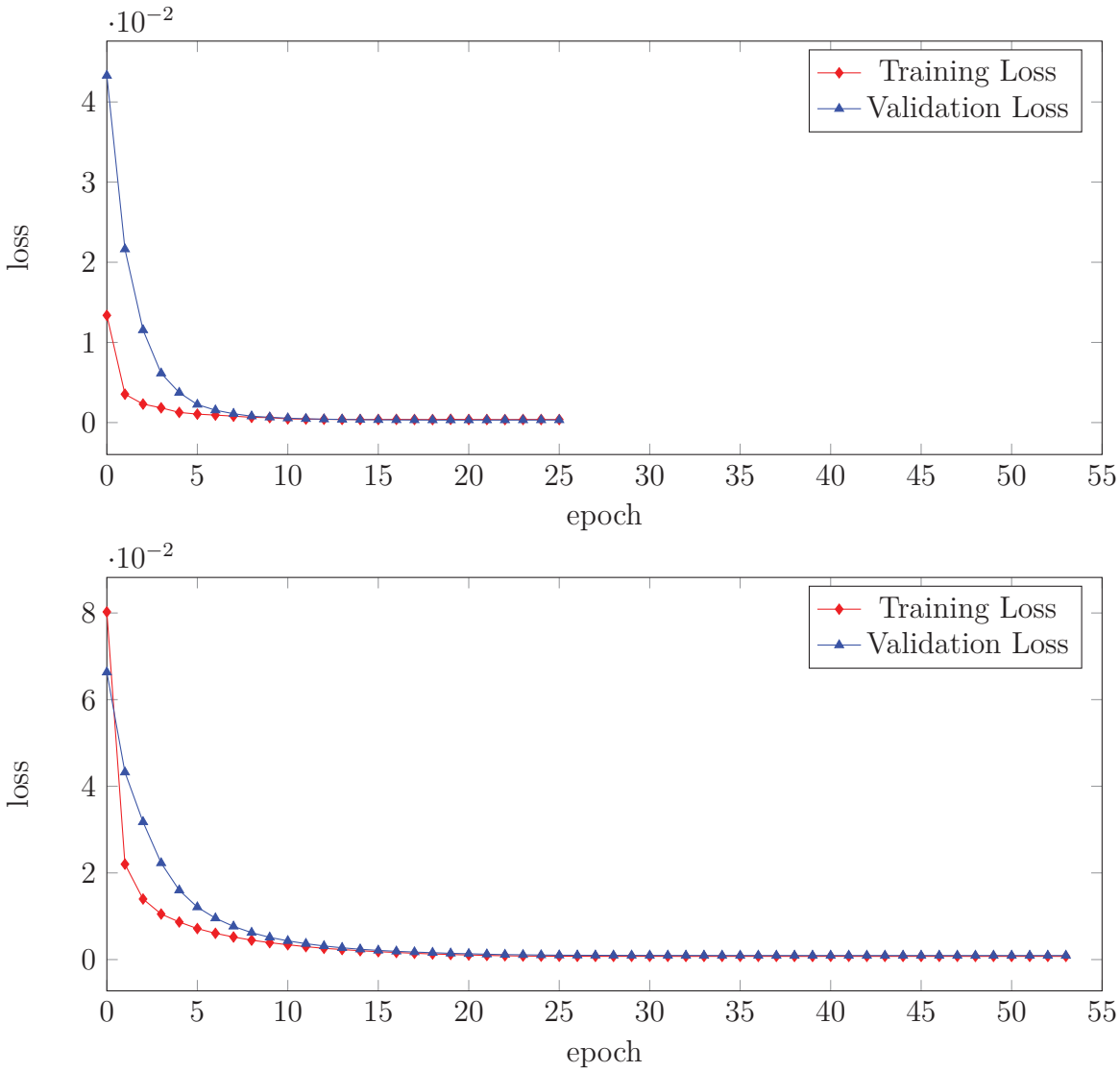


Figure 4.7: The training and validation loss for Base U-Net (top) and Deep U-Net (bottom) with Adadelta optimizer

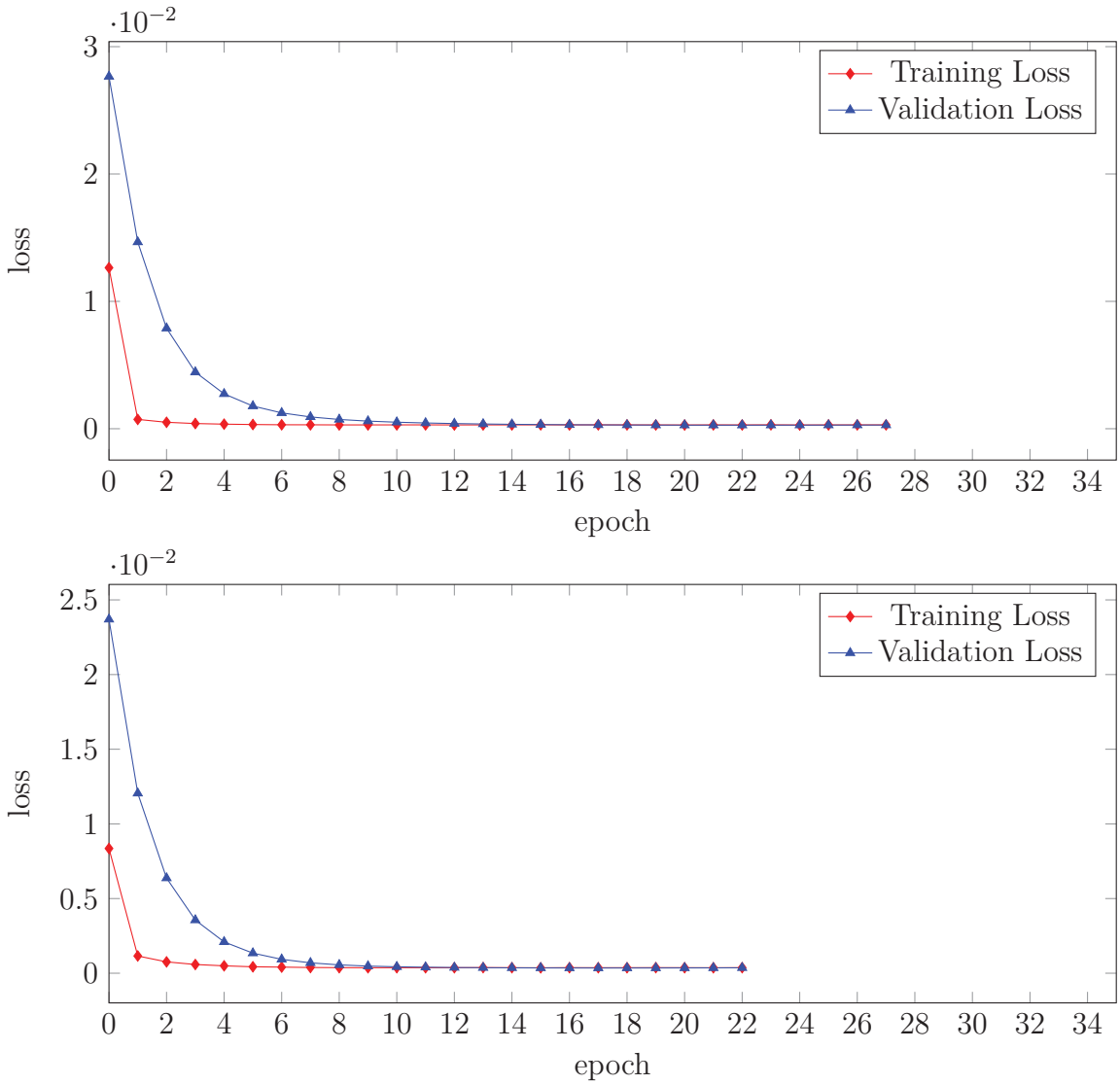


Figure 4.8: The training and validation loss for Base U-Net (top) and Deep U-Net (bottom) with Xavier normal weight initialization

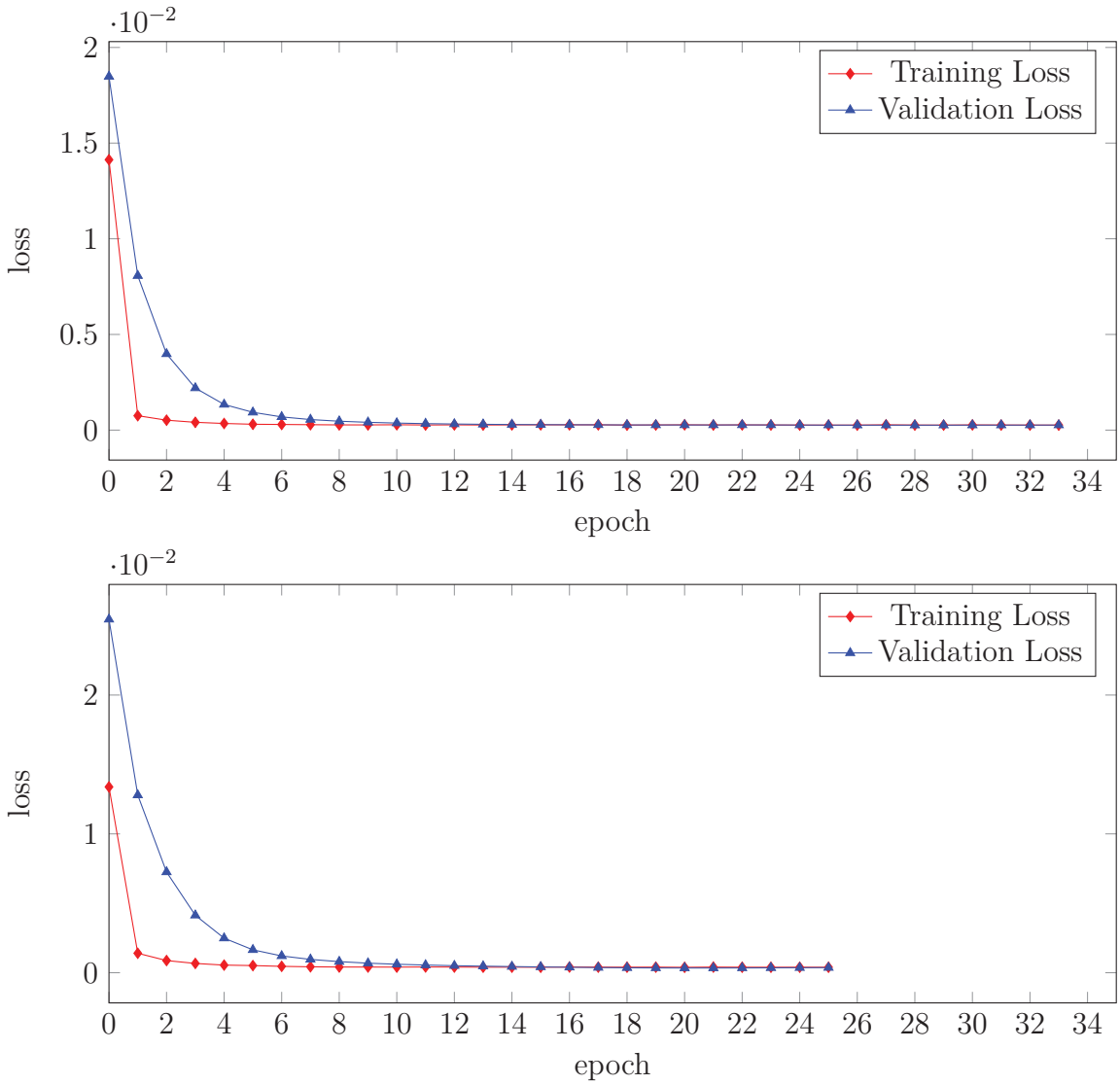


Figure 4.9: The training and validation loss for Base U-Net (top) and Deep U-Net (bottom) with Xavier uniform weight initialization

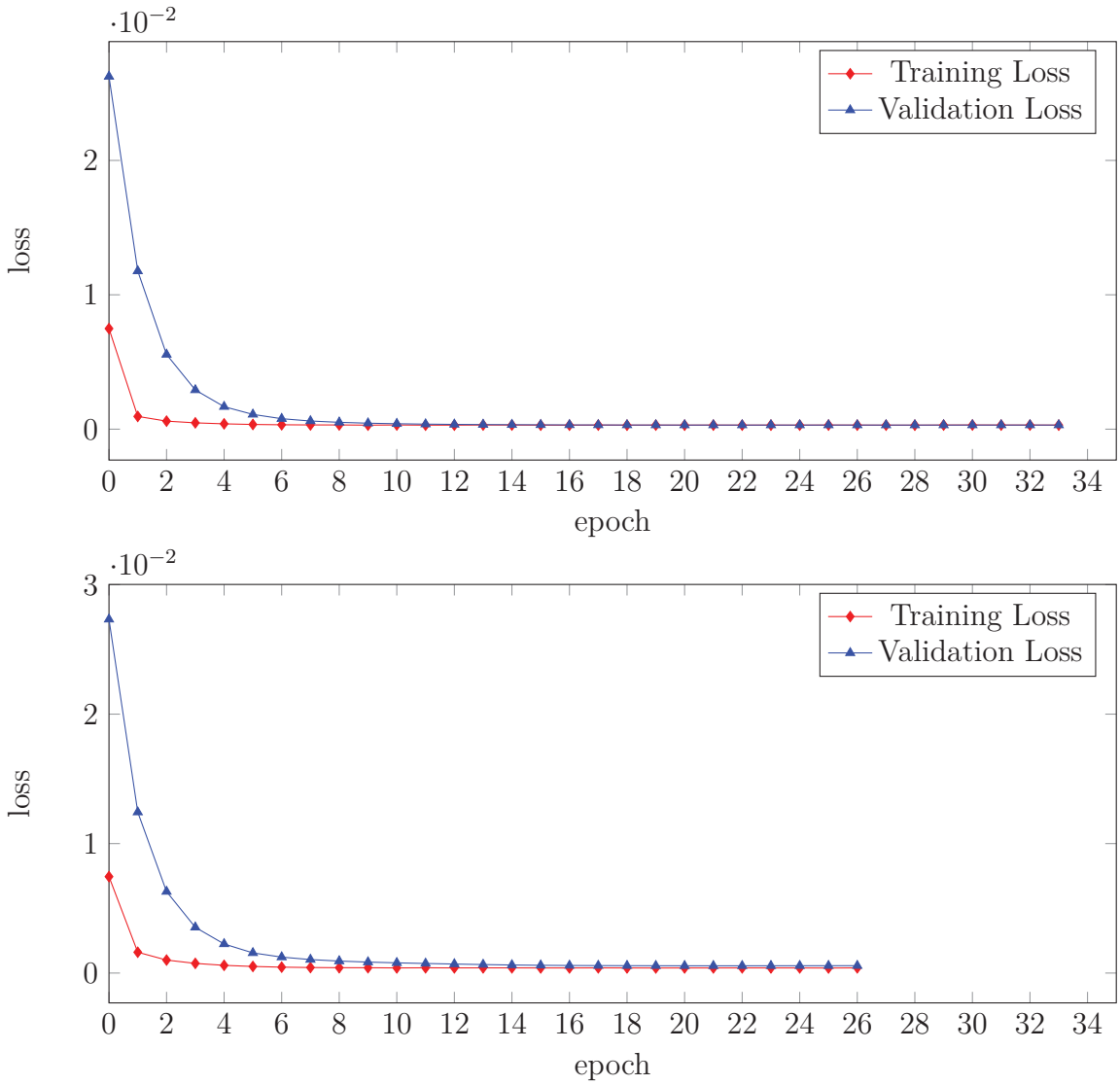


Figure 4.10: The training and validation loss for Base U-Net (top) and Deep U-Net (bottom) with He uniform weight initialization

# Chapter 5

# Discussion

Despite achieving more than 99% accuracy for all 20 models, this is not the right metric to help gain insight into the performance of each model. The high-percentage accuracy is due to the overwhelmingly large number of negative labels in the ADAM dataset. In medical imaging, the target is often an anomaly that only takes up a small portion of the human body while the rest are healthy tissues and bones. This effect is more prevalent in volumetric images. In this case, only 0.02% of the total pixels in the testing set are labeled as aneurysms.

When looking at the value of true positives and the sensitivity of each model, it appears that nearly all models failed to locate the aneurysms in the testing set. At a glance, this is a terrible outcome. Upon closer examination, however, each hyperparameter influences a model in many different ways. The subsequence sections will dive deeper into such impacts.

## 5.1 Dropout

The prediction performance on the four models with varying dropout rates is nearly identical. The accuracy and the specificity are near or at 100% while the sensitivity is the complete opposite. This may leave an impression that a stronger dropout rate does not help improve deep networks. However, when looking at the testing losses as shown in the bar chart in Figure 5.1, it is clear that the deep network get the benefits of dropout in terms of lowering the loss.

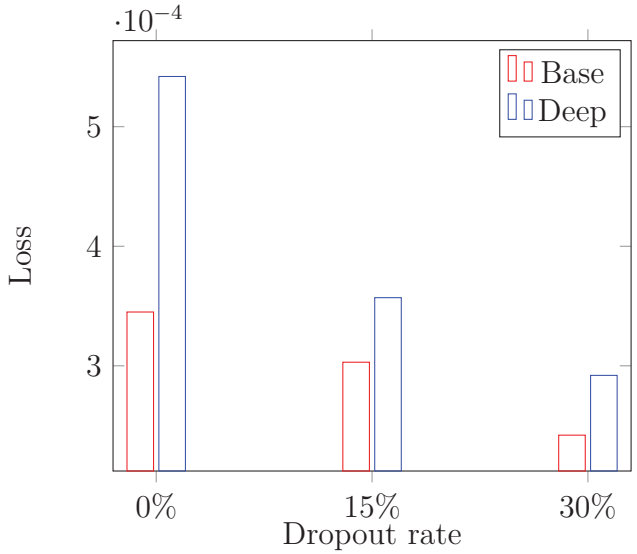


Figure 5.1: Testing loss on the Base and Deep U-Net model with varrying dropout rates

In the figure, the loss is lower as the dropout rate strengthen for both the base and deep model and the effect is more dramatic in the deeper network. One possibility is that deeper networks are overfitted resulting in less accurate prediction in the testing set. If this is the case, then the dropout works as it was intended to do as mentioned in the original paper [33].

With the lower loss, it should translate to better prediction performance. However, this is not evident in the metrics and therefore cannot fully support the claim by Liu and Deng [25] where a strong dropout rate reduces the error rate.

## 5.2 Loss Function

There is a clear improvement to the model when changing from the standard weighted binary cross-entropy loss to Dice loss. With Dice loss, the base model achieved the sensitivity of 11.4% and the deep model achieved the sensitivity of 14.7%. Similar results can be seen with

Focal Tversky loss as well. This improvement in prediction sensitivity could be an indication that the choice of loss function does bring out the benefit of deep models.

One reason why such improvement cannot be seen when using weighted binary cross-entropy could be due to the incorrect value of $\beta$. Unlike weighted binary cross-entropy where it is required to specify the value of $\beta$ despite not knowing how imbalanced the dataset is, Dice loss does not need such a constant. Therefore, Dice loss does not require the modeler to assume the distribution of the dataset and can be used as-is. Focal Tversky loss is also designed with the problem of imbalanced datasets in mind already [3].

There is an instance where the $\beta$ is set to be proportional to the negative prediction. More precisely, $\beta = \frac{N-\sum_i y_i}{\sum_i y_i}$ [35]. This variation of weighted binary cross-entropy remains to be tested. It may be possible the same improvement can be observed if the $\beta$ is set correctly. If this is the case, then it is not conclusive that the choice of a loss function contributes to the improvement. If little to no improvement is produced, then there may be benefits in using certain loss functions on the deep network. The reason is something that has to be explored in future research.

In this experiment, Dice loss outperforms Focal Tversky loss in terms of sensitivity which is not the case in the original paper [3]. This could be due to the choices of constants used in Focal Tversky loss.

But more importantly, what makes Focal Tversky loss stands out is its impact on the training. In Figure 4.5, both the training loss and validation loss are unstable from epoch to epoch. The losses also converge very fast. The losses start to become stable at around epoch 5 for Focal Tversky loss while the losses begin to stable at around epoch 10. With the right optimizer, Focal Tversky loss may be benefical to the deep model.

## 5.3 Optimizer

The model that used the Adam optimizer suffers heavily on the deep model both in terms of performance and loss. The AUC-ROC drops from 0.692 to 0.387, and the testing loss increases from 0.000335 to 0.004754. Dropout layers may be necessary to use together with the model that uses the Adam optimizer, as previously shown to reduce testing loss in the cases of models that use 15% and 30% dropout rates.

Another point of interest is how long the model that used Adam takes to train. The validation loss starts to become stable at around epoch 20 for the base model and around epoch 25 for the deep model. Almost twice as much when compared with Adadelta optimizer.

## 5.4 Weight Initialization

The choice of weight initialization doesn't seem to have any impact on the model performance. The convergence of the lost seems to be the same across all models. It is worth noting that the value of AUC-ROC decreases only when trained on the deep model with Xavier normal initialization method.

# Chapter 6

# Conclusions

With all the metrics used, there is no perfect indicator. Each hyperparameter has different effects. With the right configuration and by evaluating the right metrics, there is evidence to support the need of deep neural networks. In this experiment, a strong dropout rate helped to improve the loss and avoid overfitting. Dice loss improved the sensitivity in the testing set, however, had a lower F-1 score due to false positives. Compared to the results from Focal Tversky loss, the F-1 score is higher in the deep model than all other models. While the weight initialization and optimizers do not show any direct improvement, some changes to the parameters can be made to better fit the ADAM dataset.

More work is still needed to help refine and support the findings in this thesis. Future researchers can explore other configurations that might affect the performance of deep neural networks. For example, test on another hyperparameter like activation functions or pooling layers, or even a combination of these hyperparameters can be explored, as some show promise to work well together. For example, a quick-converging loss like Focal Tversky loss and adaptive gradient optimizer like Adam. Using different network architectures (e.g., with multiple branches or reinforcement learning) might give a different set of benefits or disadvantages. Other configurations that might be important include but are not necessarily limited to input resolutions, the number of output kernels in each convolutional layer, and the placement of the dropout layers. Other metrics such as the AUC of PR (precision-recall) curve may also be useful in evaluating datasets with small positive labels ratio.

# Chapter 7

# Summary

To better capture patterns in a dataset, sometimes it is necessary for the model to be complex. In deep learning, making a model too complex is not a good practice as it leads to undesirable side effects. Base on the recent research, however, well configured deep neural networks can reduce error rates in image classifications [25].

From the experiment presented in this thesis, there is evidence to support such a claim even when scaling up to image segmentation tasks. With the right choices of hyperparameters, the deep model surpassed the base model in terms of its ability to recognize and locate intracranial aneurysms from 3D TOF-MRA images in the ADAM dataset.

# Bibliography

[1] Tinkercliffs, arc's flagship resource. URL https://www.docs.arc.vt.edu/resources/compute/00tinkercliffs.html.

[2] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL https://www.tensorflow.org/. Software available from tensorflow.org.

[3] Nabila Abraham and Naimul Mefraz Khan. A novel focal tversky loss function with improved attention u-net for lesion segmentation. *CoRR*, abs/1810.07842, 2018. URL http://arxiv.org/abs/1810.07842.

[4] Michele Andreucci, Richard Solomon, and Adis Tasanarong. Side effects of radiographic contrast media: pathogenesis, risk factors, and prevention. *BioMed research international*, 2014:741018–741018, 2014. ISSN 2314-6141. doi: 10.1155/2014/741018. URL https://pubmed.ncbi.nlm.nih.gov/24895606. 24895606[pmid].

[5] Wadii Boulila, Maha Driss, Mohamed Al-Sarem, Faisal Saeed, and Moez Krichen.

Weight initialization techniques for deep learning algorithms in remote sensing: Recent trends and future perspectives, 2021.

[6] Matthew Brett, Christopher J. Markiewicz, Michael Hanke, Marc-Alexandre Côté, Ben Cipollini, Paul McCarthy, Dorota Jarecka, Christopher P. Cheng, Yaroslav O. Halchenko, Michiel Cottaar, Eric Larson, Satrajit Ghosh, Demian Wassermann, Stephan Gerhard, Gregory R. Lee, Hao-Ting Wang, Erik Kastman, Jakub Kaczmarzyk, Roberto Guidotti, Or Duek, Jonathan Daniel, Ariel Rokem, Cindee Madison, Brendan Moloney, Félix C. Morency, Mathias Goncalves, Ross Markello, Cameron Riddell, Christopher Burns, Jarrod Millman, Alexandre Gramfort, Jaakko Leppäkangas, Anibal Sólon, Jasper J.F. van den Bosch, Robert D. Vincent, Henry Braun, Krish Subramaniam, Krzysztof J. Gorgolewski, Pradeep Reddy Raamana, Julian Klug, B. Nolan Nichols, Eric M. Baker, Soichi Hayashi, Basile Pinsard, Christian Haselgrove, Mark Hymers, Oscar Esteban, Serge Koudoro, Fernando Pérez-García, Nikolaas N. Oosterhof, Bago Amirbekian, Ian Nimmo-Smith, Ly Nguyen, Samir Reddigari, Samuel St-Jean, Egor Panfilov, Eleftherios Garyfallidis, Gael Varoquaux, Jon Haitz Legarreta, Kevin S. Hahn, Oliver P. Hinds, Bennet Fauber, Jean-Baptiste Poline, Jon Stutters, Kesshi Jordan, Matthew Cieslak, Miguel Estevan Moreno, Valentin Haenel, Yannick Schwartz, Zvi Baratz, Benjamin C Darwin, Bertrand Thirion, Carl Gauthier, Dimitri Papadopoulos Orfanos, Igor Solovey, Ivan Gonzalez, Jath Palasubramaniam, Justin Lecher, Katrin Leinweber, Konstantinos Raktivan, Markéta Calábková, Peter Fischer, Philippe Gervais, Syam Gadde, Thomas Ballinger, Thomas Roos, Venkateswara Reddy Reddam, and freec84. nipy/nibabel: 3.2.1, November 2020. URL https://doi.org/10.5281/zenodo.4295521.

[7] Jason Brownlee. How to configure the learning rate when training deep learning neural networks, 2019. URL https://machinelearningmastery.com/learning-rate-for-deep-learning-neural-networks/.

[8] Rubén Cardenes, José María Pozo, Hrvoje Bogunovic, Ignacio Larrabide, and Alejandro F. Frangi. Automatic aneurysm neck detection using surface voronoi diagrams. *IEEE Transactions on Medical Imaging*, 30(10):1863–1876, 2011. doi: 10.1109/TMI.2011.2157698.

[9] Geng Chen, Xia Wei, Huang Lei, Yang Liqin, Li Yuxin, Dai Yakang, and Geng Daoying. Automated computer-assisted detection system for cerebral aneurysms in time-of-flight magnetic resonance angiography using fully convolutional network. *BioMedical Engineering OnLine*, 19(1):38, May 2020. ISSN 1475-925X. doi: 10.1186/s12938-020-00770-7. URL https://doi.org/10.1186/s12938-020-00770-7.

[10] François Chollet et al. Keras. https://keras.io, 2015.

[11] DP Harrington, LM Boxt, and PD Murray. Digital subtraction angiography: overview of technical principles. *American Journal of Roentgenology*, 139(4):781–786, 1982. doi: 10.2214/ajr.139.4.781. URL https://doi.org/10.2214/ajr.139.4.781. PMID: 6751053.

[12] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020. doi: 10.1038/s41586-020-2649-2. URL https://doi.org/10.1038/s41586-020-2649-2.

[13] Tao Hu, Heng Yang, Wei Ni, Yu Lei, Zhuoyun Jiang, Keke Shi, Jinhua Yu, Yuxiang Gu, and Yuanyuan Wang. Automatic detection of intracranial aneurysms in 3d-dsa based on a bayesian optimized filter. *BioMedical Engineering OnLine*, 19(1):73, Sep 2020.

ISSN 1475-925X. doi: 10.1186/s12938-020-00817-9. URL https://doi.org/10.1186/s12938-020-00817-9.

[14] Emrah Irmak. A novel deep convolutional neural network model for covid-19 disease detection. In *2020 Medical Technologies Congress (TIPTEKNO)*, pages 1–4, 2020. doi: 10.1109/TIPTEKNO50054.2020.9299286.

[15] Shruti Jadon. A survey of loss functions for semantic segmentation. *2020 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB)*, Oct 2020. doi: 10.1109/cibcb48159.2020.9277638. URL http://dx.doi.org/10.1109/CIBCB48159.2020.9277638.

[16] Tim Jerman, Franjo Pernuš, Boštjan Likar, and Žiga Špiclin. Computer-aided detection and quantification of intracranial aneurysms. In Nassir Navab, Joachim Hornegger, William M. Wells, and Alejandro Frangi, editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pages 3–10, Cham, 2015. Springer International Publishing. ISBN 978-3-319-24571-3.

[17] Tim Jerman, Franjo Pernus, Boštjan Likar, and Žiga Špiclin. Aneurysm detection in 3d cerebral angiograms based on intra-vascular distance mapping and convolutional neural networks. In *2017 IEEE 14th International Symposium on Biomedical Imaging (ISBI 2017)*, pages 612–615, 2017. doi: 10.1109/ISBI.2017.7950595.

[18] Jaber Juntu, Jan Sijbers, Dirk Van Dyck, and Jan Gielen. Bias field correction for mri images. In Marek Kurzyński, Edward Puchała, Michał Woźniak, and Andrzej żołnierek, editors, *Computer Recognition Systems*, pages 543–551, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg. ISBN 978-3-540-32390-7.

[19] Eftychia Z. Kapsalaki, Christos D. Rountas, and Kostas N. Fountas. The role of 3 tesla mra in the detection of intracranial aneurysms. *International Journal of Vascular*

*Medicine*, 2012:792834, Jan 2012. ISSN 2090-2824. doi: 10.1155/2012/792834. URL https://doi.org/10.1155/2012/792834.

[20] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

[21] Ignacio Larrabide, Maria Cruz Villa-Uriol, Rubén Cárdenes, Jose Maria Pozo, Juan Macho, Luis San Roman, Jordi Blasco, Elio Vivas, Alberto Marzo, D. Rod Hose, and Alejandro F. Frangi. Three-dimensional morphological analysis of intracranial aneurysms: A fully automated method for aneurysm sac isolation and quantification. *Medical Physics*, 38(5):2439–2449, 2011. doi: https://doi.org/10.1118/1.3575417. URL https://aapm.onlinelibrary.wiley.com/doi/abs/10.1118/1.3575417.

[22] Alexandra Lauric, Eric L. Miller, Sarah F. Frisken, and Adel M. Malek. Automated detection of intracranial aneurysms based on parent vessel 3d analysis. *Medical image analysis*, 14 2:149–59, 2010.

[23] Lisha Li, Kevin Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, and Ameet Talwalkar. Hyperband: A novel bandit-based approach to hyperparameter optimization, 2018.

[24] Ruirui Li, Wenjie Liu, Lei Yang, Shihao Sun, Wei Hu, Fan Zhang, and Wei Li. Deepunet: A deep fully convolutional network for pixel-level sea-land segmentation. *CoRR*, abs/1709.00201, 2017. URL http://arxiv.org/abs/1709.00201.

[25] Shuying Liu and Weihong Deng. Very deep convolutional neural network based image classification using small training sample size. In *2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR)*, pages 730–734, 2015. doi: 10.1109/ACPR.2015.7486599.

[26] Kevin P. Murphy. *Machine Learning A Probabilistic Perspective*. The MIT Press, 2012.

[27] Tom O'Malley, Elie Bursztein, James Long, François Chollet, Haifeng Jin, Luca Invernizzi, et al. Kerastuner. https://github.com/keras-team/keras-tuner, 2019.

[28] M.E. Paoletti, J.M. Haut, J. Plaza, and A. Plaza. A new deep convolutional neural network for fast hyperspectral image classification. *ISPRS Journal of Photogrammetry and Remote Sensing*, 145:120–147, 2018. ISSN 0924-2716. doi: https://doi.org/10.1016/j.isprsjprs.2017.11.021. URL https://www.sciencedirect.com/science/article/pii/S0924271617303660. Deep Learning RS Data.

[29] Alexander R Podgorsak, Ryan A Rava, Mohammad Mahdi Shiraz Bhurwani, Anusha R Chandra, Jason M Davies, Adnan H Siddiqui, and Ciprian N Ionita. Automatic radiomic feature extraction using deep learning for angiographic parametric imaging of intracranial aneurysms. *Journal of NeuroInterventional Surgery*, 12(4):417–421, 2020. ISSN 1759-8478. doi: 10.1136/neurintsurg-2019-015214. URL https://jnis.bmj.com/content/12/4/417.

[30] Ines Rahmany, Mohamed El Arbi Nemmala, Nawres Khlifa, and Houda Megdiche. Automatic detection of intracranial aneurysm using lbp and fourier descriptor in angiographic images. *International Journal of Computer Assisted Radiology and Surgery*, 14(8):1353–1364, Aug 2019. ISSN 1861-6429. doi: 10.1007/s11548-019-01996-0. URL https://doi.org/10.1007/s11548-019-01996-0.

[31] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation, 2015.

[32] D Saloner. The aapm/rsna physics tutorial for residents. an introduction to mr angiography. *RadioGraphics*, 15(2):453–465, 1995. doi: 10.1148/radiographics.15.2.7761648. URL https://doi.org/10.1148/radiographics.15.2.7761648. PMID: 7761648.

[33] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfit-

ting. *Journal of Machine Learning Research*, 15(56):1929–1958, 2014. URL http://jmlr.org/papers/v15/srivastava14a.html.

[34] Joseph N. Stember, Peter Chang, Danielle M. Stember, Michael Liu, Jack Grinband, Christopher G. Filippi, Philip Meyers, and Sachin Jambawalikar. Convolutional neural networks for the detection and measurement of cerebral aneurysms on magnetic resonance angiography. *Journal of Digital Imaging*, 32(5):808–815, Oct 2019. ISSN 1618-727X. doi: 10.1007/s10278-018-0162-z. URL https://doi.org/10.1007/s10278-018-0162-z.

[35] Carole H. Sudre, Wenqi Li, Tom Vercauteren, Sebastien Ourselin, and M. Jorge Cardoso. Generalised dice overlap as a deep learning loss function for highly unbalanced segmentations. In M. Jorge Cardoso, Tal Arbel, Gustavo Carneiro, Tanveer Syeda-Mahmood, João Manuel R.S. Tavares, Mehdi Moradi, Andrew Bradley, Hayit Greenspan, João Paulo Papa, Anant Madabhushi, Jacinto C. Nascimento, Jaime S. Cardoso, Vasileios Belagiannis, and Zhi Lu, editors, *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*, pages 240–248, Cham, 2017. Springer International Publishing. ISBN 978-3-319-67558-9.

[36] Kimberley Timmins. Adam aneurysm detection and segmentation challenge: Data. URL https://adam.isi.uu.nl/data/.

[37] Stéfan van der Walt, Johannes L. Schönberger, Juan Nunez-Iglesias, François Boulogne, Joshua D. Warner, Neil Yager, Emmanuelle Gouillart, and Tony Yu. scikit-image: Image processing in python. *CoRR*, abs/1407.6245, 2014. URL http://arxiv.org/abs/1407.6245.

[38] C. Vega, J. V. Kwoon, and S. D. Lavine. Intracranial aneurysms: current evidence and clinical practice. *Am Fam Physician*, 66(4):601–608, Aug 2002.

[39] Wei Yuan and Wenbo Xu. Neighborloss: A loss function considering spatial correlation for semantic segmentation of remote sensing image. *IEEE Access*, 9:75641–75649, 2021. doi: 10.1109/ACCESS.2021.3082076.

[40] Matthew D. Zeiler. Adadelta: An adaptive learning rate method, 2012.

[41] Junjie Zhao, Hao Lin, Richard Summers, Mingmin Yang, Brian G. Cousins, and Janice Tsui. Current treatment strategies for intracranial aneurysms: An overview. *Angiology*, 69(1):17–30, 2018. doi: 10.1177/0003319717700503. URL https://doi.org/10.1177/0003319717700503. PMID: 28355880.

[42] Özgün Çiçek, Ahmed Abdulkadir, Soeren S. Lienkamp, Thomas Brox, and Olaf Ronneberger. 3d u-net: Learning dense volumetric segmentation from sparse annotation, 2016.

# Appendices

# Appendix A

# Architecture Details

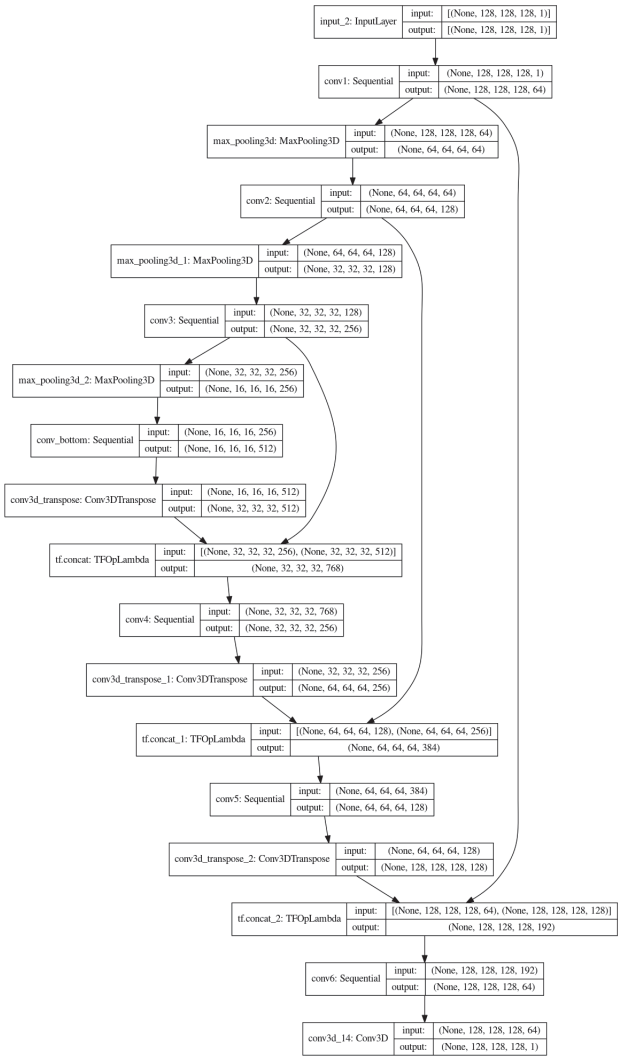


Figure A.1: Tensorflow plot of the Base U-Net model

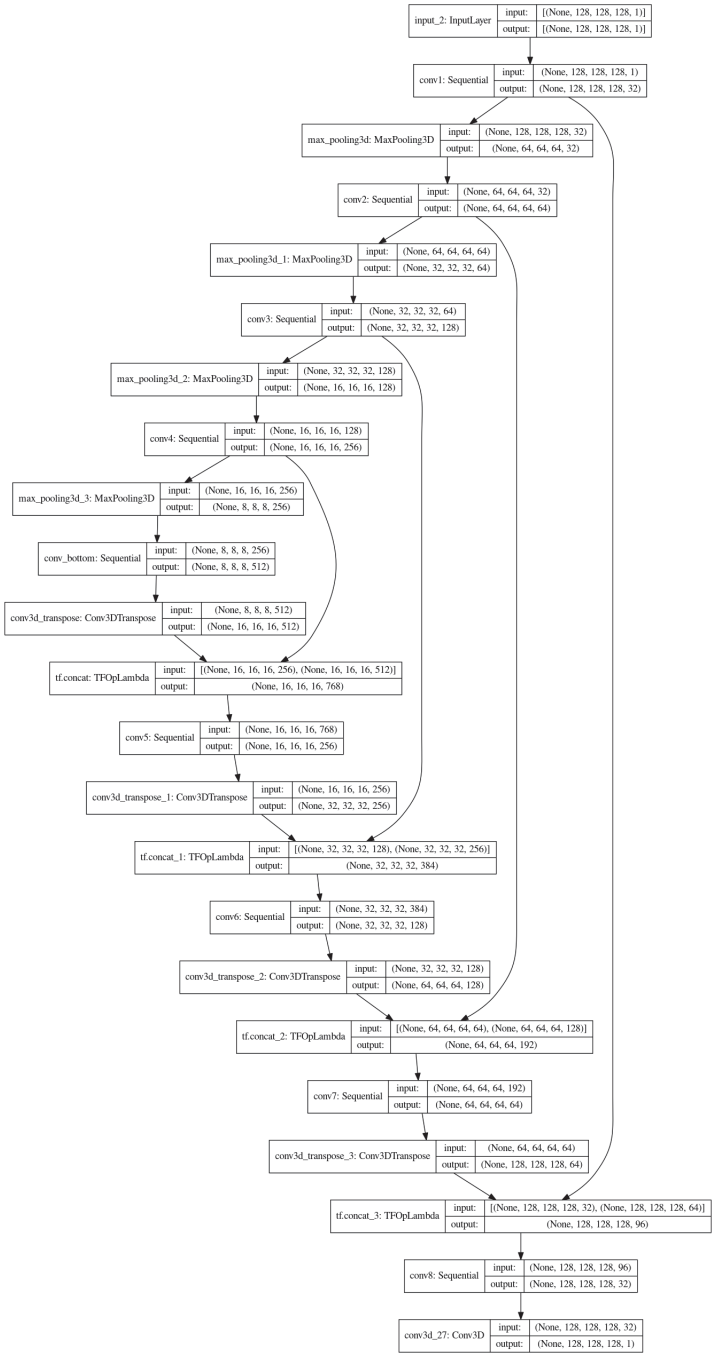


Figure A.2: Tensorflow plot of the Deep U-Net model

# Appendix B

# TinkerCliffs Specification

An A100 GPU node can be requested at a maximum of 8 NVIDIA A100-80G GPUs, 128 cores of an AMD EPYC 7742 CPU, 2048 GB of memory. [1]