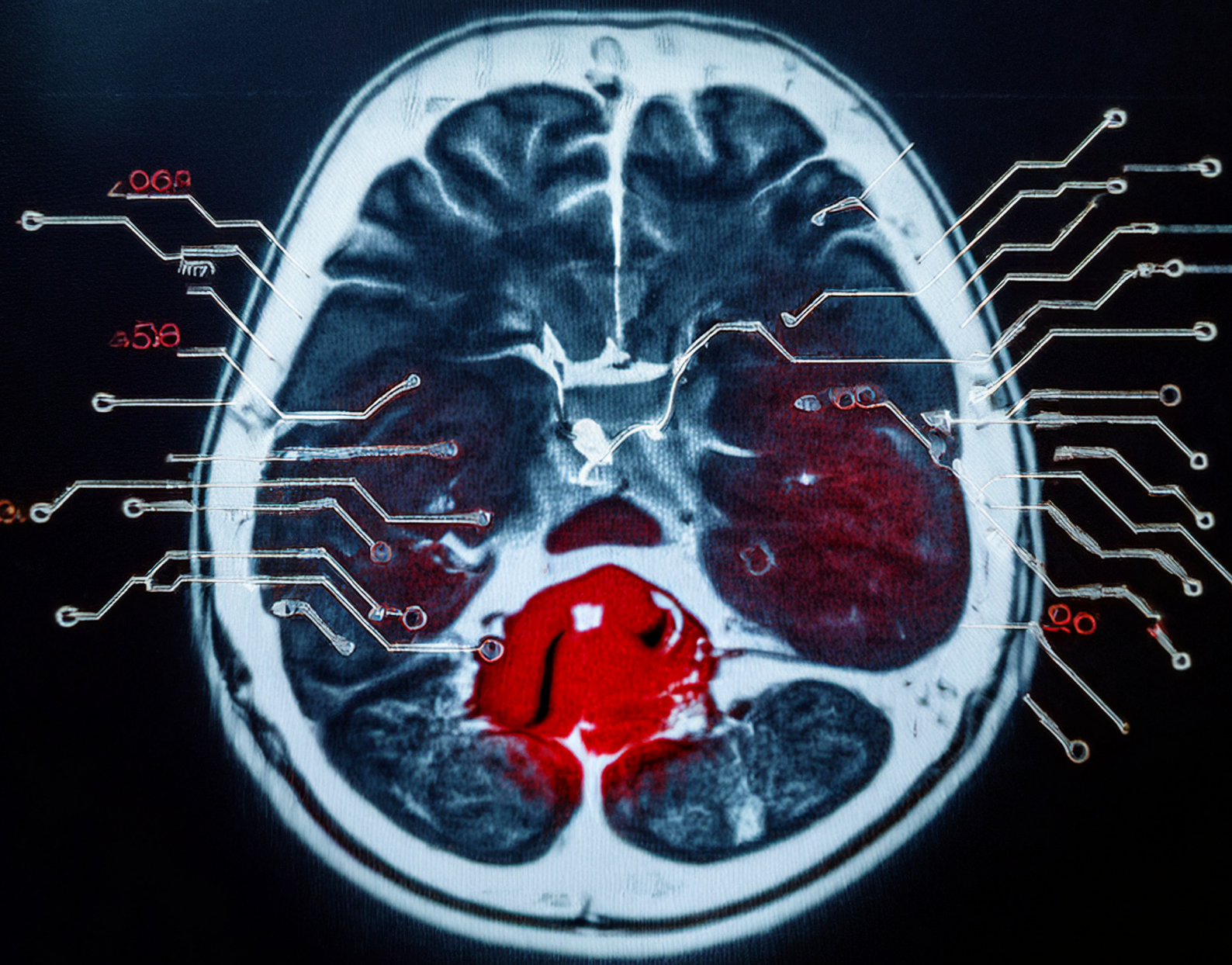
Matura Thesis

Alte Kantonsschule Aarau

AI-Driven Diagnostic Tool for Early Detection of Brain Tumors



October 2025

Mikkel Lüscher, G22K

Submitted to Martina Vàzquez

Contents

Abstract

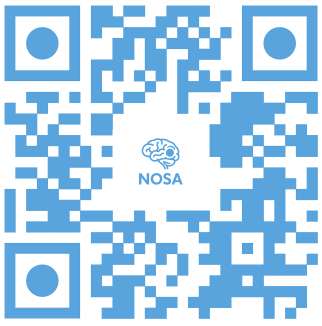
This Matura thesis addresses the need for improved diagnostic tools in brain tumor segmentation, which is a manual and time-consuming process subject to inter-observer variability. This thesis reviews existing approaches, evaluates the feasibility of artificial intelligence (AI)-driven diagnostic tools, and adapts a U-Net architecture for segmenting brain tumors in 2-dimensional magnetic resonance imaging (MRI) slices. This thesis offers a detailed theoretical introduction to MRI, brain tumors, and most importantly image segmentation using AI. The thesis discusses the methods for developing the proposed AI model, a methodical review of state-of-the-art models, and a systematic comparison of the proposed model and state-of-the-art models.

The proposed AI model, used in Neuro Oncology Segmentation Algorithm (NOSA)v2 employs an adapted U-Net trained on 3064 T1-weighted contrast enhanced MRI images. The model was trained using a combined dice and binary cross entropy (BCE) loss function and an adaptive moment estimation (Adam) optimizer. The model used in NOSAv2 features a user friendly graphical user interface (GUI) to facilitate practical implementation. The results show that NOSAv2 significantly improves recall at the price of precision, achieving a mean accuracy comparable to that of the state-of-the-art models. However, NOSAv2 shows a notable deficit in intersection over union (IoU) due to architectural simplifications, driven by computational restraints. Although, the product does not reach a clinical standard, it lays a solid foundation for further development.

The thesis further discusses the current and future use of AI-driven tools in radiology through an interview with a neuroradiologist. The interview confirmed the value of AI-driven tools, but only as an augmentation to human expertise.

Complex terms and jargon are defined in the glossary in Appendix B.

The GitHub with all mentioned code can be accessed via this link:



1. Introduction

1.1 Background and Context

Brain tumors are among the most serious and life-threatening neurological conditions that require an exact diagnosis and treatment planning. Segmenting a brain tumor is typically done manually by a radiologist, which is time-consuming. Manually segmenting a tumor can take 10 minutes per tumor for a single slice of an MRI image and is subject to inter-observer variability [1]. This has led to a growing interest in deep learning-driven segmentation tools, especially in convolutional neural networks (CNNs). CNNs are accurate and time efficient; in surgery, a CNN model can accurately diagnose a tumor in less than 150 seconds [2]. In this thesis, the focus is on segmenting brain tumors in MRI images using U-Net, a common CNN in segmentation. This is a slower, but non-invasive alternative. [3]

1.2 Objectives and Motivation

Early and accurate diagnosis is one of the most important factors for increasing the survival rates and quality of life during brain tumor treatment. The inspiration for this work is personal: in 2022, a family member was diagnosed with a brain tumor. Their illness and eventual loss showed the need for improved diagnostic tools to ease the battle for the patient and their loved ones.

This thesis pursues three main objectives. First, it reviews existing models to assess their methodologies and applications. Second, it analyzes the feasibility of AI-based diagnostic tools, considering both accuracy and practical implementation. Third, it adapts an existing U-Net structure to segment brain tumors, aiming for the highest accuracy given the available resources.

2. Theoretical Background

2.1 Magnetic Resonance Imaging and Its Properties

MRI images are the basis of this thesis, understanding how MRI works is essential to make an optimal algorithm. MRI operates by combining a strong magnetic field and a burst of radio frequency (RF) energy, which produces a detailed image of the body's soft tissues, organs, and bones. MRI can capture 3-dimensional images with details that go far beyond the capabilities of an X-ray machine. The 3-dimensional MRI images can be transformed to 2-dimensional slices, on either the axial, sagittal, or coronal plane, as can be seen in Figure 1.

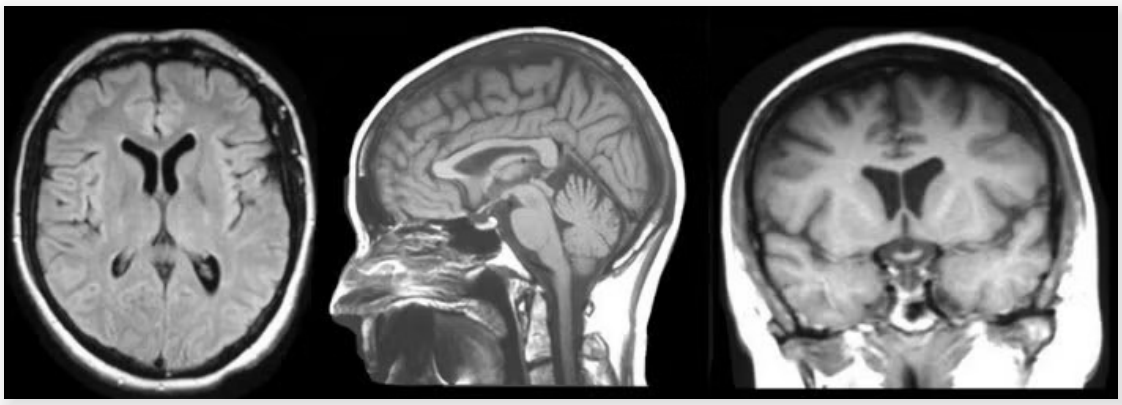


Figure 1: From left to right MRI images on the axial, coronal, and sagittal plane. [4]

2.1.1 Basics of Magnetic Resonance Imaging (MR Physics)

An MRI machine is made up of a set of primary magnets that are responsible for the main magnetic field. Then, there are gradient coils, which create a secondary magnetic field. There are three gradient coils, one for each axis in 3-dimensional space, which enables spatial encoding. These magnetic fields are disturbed when a patient is placed inside the machine, so an MRI technician will use various tools to improve the homogeneity of the field. This is important because an inconsistent magnetic field can lead to distortion, possibly rendering the image unusable. Finally, the machine has RF coils that send pulses toward the desired area. [5]

An MRI machine uses the fact that the body comprises approximately 70% water, which contains hydrogen. Hydrogen has a single proton carrying a positive charge. The proton, which is naturally randomly oriented, has its own small magnetic field. When a strong magnetic field (main magnetic

field, usually 1.5 Tesla) is applied by the MRI machine, the protons align themselves with the field lines of that magnetic field. They are now either parallel (low-energy state) or antiparallel (high-energy state), although slightly more are parallel. The RF coils then send a pulse toward the desired area, which causes all protons to flip to high-energy states, either 90° or 180° to the main magnetic field, depending on the strength and duration of the RF burst. The RF burst also causes the protons to spin in phase, creating a measurable transverse magnetization. Flipping a proton is only possible if the RF is exactly tuned to the precessional frequency of the electrons, and the proton needs to resonate with the frequency. This frequency is called the Larmor frequency and is directly proportional to the magnetic field strength. [5]

2.1.2 Image Contrasts and Tissue Types (T1, T2, Flair)

There are countless different MRI sequences designed to show different tissues of the body, only the most common MRI sequences will be explained. A sequence in the context of MRI means the settings applied to the machine to achieve a certain image.

After the RF burst is over, the protons slowly flip back to their low-energy states, which releases measurable energy. T1 relaxation time is the time it takes for 63% of the protons to relax back into their low-energy state. Different tissues have different T1 relaxation times. Water-based substances, such as cerebrospinal fluid (CSF) have a long T1 relaxation time and produce a dark signal, as can be seen in Figure 2 in the T1-weighted image, whereas fat has a short T1 relaxation time and produces a bright signal. T2 relaxation time is the time it takes for 37% of protons to no longer rotate in phase, meaning there is no longer a measurable transverse magnetization. For T2 relaxation time, it is the opposite, CSF has a short T2 relaxation time and produces a bright signal, whereas fat has a long T2 relaxation time and produces a dark signal. This can be seen in Figure 2 in the T2-weighted image. [4, 5]

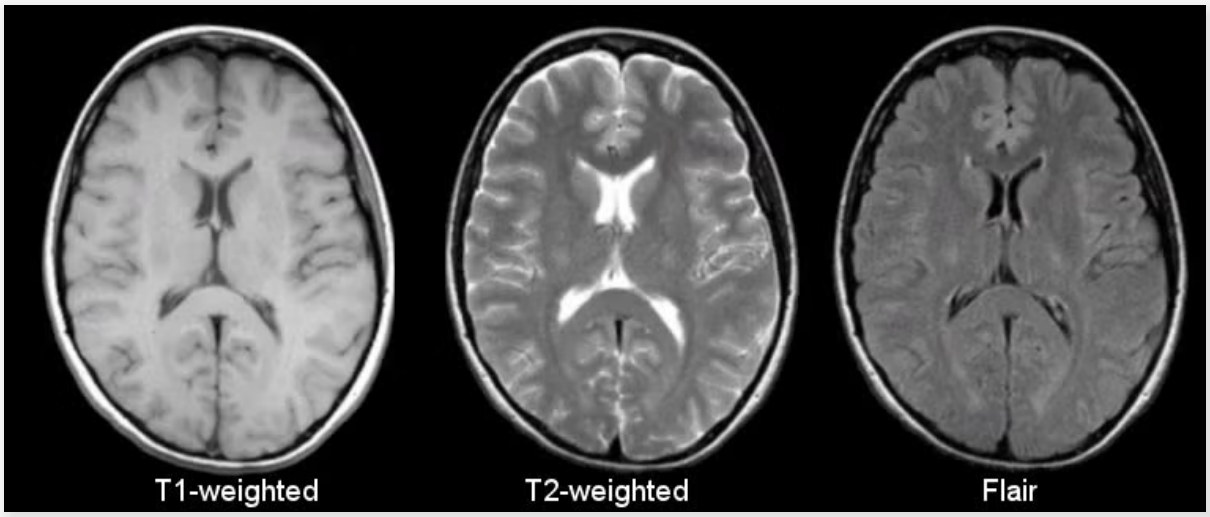


Figure 2: Comparison T1-weighted, T2-weighted and FLAIR MRI images. [4]

In the Spin Echo sequence, the measured signals can be changed to make a clearer contrast, depending on which tissue is important to look at. This is done by manipulating the repetition time (TR) and the

time to echo (TE). TR is the amount of time between successive RF bursts on the same slice of the brain, and TE is the amount of time between the peak of 90° RF burst and the echo (measuring the signal emitted by the protons flipping back into their low-energy state). A short TR and TE times will create a T1-weighted image since the T1 relaxation times of the tissue mostly determine its brightness. A T2-weighted image is the result of longer TR and TE times, with more time the signals created by T1 relaxation have passed and mainly the fading of the transverse magnetization (T2 relaxation) remains measurable. [4, 5]

Similar to a T2-weighted image, but having longer and most importantly precisely timed TR and TE times, the signal of water-based substances like CSF is suppressed, as seen in Figure 2 in the fluid attenuated inversion recovery (FLAIR) image. There is an initial 180 ° RF burst and the TR is timed to fire just when the T1 signal of CSF would be measured. This nullifies the signal of CSF, making it dark in the image. This sequence is particularly valuable in pathology because abnormalities such as tumors appear bright in sequences with long TR and TE times, and by suppressing the otherwise bright CSF, abnormalities become more clearly visible. [4, 5]

2.1.3 Common Artifacts and Their Effects on MRI Images

MRI images are commonly subject to artifacts, visual abnormalities that do not correspond to the object being scanned. The most common type of artifact are movement artifacts in which the patient moves. Willfully moving, breathing, or even the pulsing of veins can cause blurring and ghosting in the images. A small amount of blurring and ghosting is almost unavoidable and rarely a problem in diagnosis. Metal objects such as non-ferromagnetic implants can cause bigger problems because their magnetic field disturbs the magnetic fields of MRI machines and creates dark or bright spots. Also, a "Zipper" artifact can be created if there is an accidental RF wave which corrupts the image. For example, if the Faraday cage around the MRI machine, which is made to shield the machines from such waves, is not sealed properly. These artifacts, depending on the severity, make the diagnosis more difficult. [4]

2.2 Brain Tumors

2.2.1 Brief overview of Brain Tumors and the different Types of Brain Tumors

A brain tumor is an abnormal mass located in the brain or skull and mostly affects children between the ages of 3 and 12 and adults between the ages of 40 and 70 [6]. In 2019, there were 347,992 new cases of malignant brain tumors recorded worldwide, and the number of cases is growing each year. [7]

Brain tumors are classified as either a primary tumor or a secondary tumor, also known as a metastatic tumor. A primary brain tumor originates from the brain tissue, while a metastatic brain tumor is created from cancerous cells that spread from a cancerous tumor somewhere else in the body. A primary tumor is either benign, meaning it is a noncancerous tumor, or malignant, meaning it is a cancerous tumor. A benign tumor grows slowly and will not spread to other parts of the body. A malignant tumor will grow rapidly and aggressively; fortunately, it is unlikely for a cancerous tumor

that stems from brain tissue to create metastatic tumors. A tumor can evolve, a benign tumor can become a malignant tumor, and malignant tumors can progress through the stages. [6, 8]

Cancer is divided into four stages, depending on the severity of the cancer and how far it has progressed. Stage 0 is cancerous but has not spread to any nearby tissue. From stages I to III, the size and spread increases further. In stage IV cancer, the cancer has spread to distant parts of the body. [9]

Brain tumors derive their name from the tissue they originate from. Some of the most common brain tumors include:

- **Metastatic** brain tumors are the most common tumor in adults and are most commonly spread from skin cancer, lung cancer, breast cancer, or lymphatic cancer. The tumor cells in the brain resemble the cells from which the tumor originates [6]. Life expectancy depends on the location and severity of the tumor but has increased greatly in recent years. Before modern treatment, the life expectancy was less than 6 months, but today most people with metastatic tumors no longer die from them. [10]
- **Gliomas**, which stem from glial tissue in the brain, are the most common type of brain tumor. There are many types of gliomas, with different severity, the most renowned is the glioblastoma multiforme (GBM), a stage IV malignant tumor that can result in death in 6 months or less. [11]
- **Meningiomas**, a typically benign tumor that stems from the tissue surrounding the brain and spinal cord [6]. It has a very high survival rate, 80% to 90% of patients surviving 10 years or longer after their diagnosis. The chances for a meningioma to become cancerous are under 5%. [12]

2.2.2 Clinical Symptoms and the Importance of Early Tumor Detection

Symptoms of a brain tumor can be difficult to tie to a tumor because they often resemble other diseases. Patients do not feel a tumor; they only feel the symptoms caused by the growth of the tumor, which damages the healthy brain tissue. Initially, the symptoms may be mild or intermittent, but they will rapidly become increasingly severe, depending on the type of tumor it is. Many people do not deem these light symptoms severe enough to see a doctor since that can be very expensive in some parts of the world. However, if people only see a doctor when the symptoms become unbearable, the tumor has often progressed to later stages, drastically decreasing the chances of survival. These are some of the most common symptoms [6]:

- Severe headaches, which the tend to be worse in the morning and ease during the day
- Vomiting and nausea
- Seizures
- Weakness in one side of the body
- Trouble talking and slurred speech

- Loss of coordination
- Changes in vision or abnormal eye movements
- Changes in memory and personality
- Ringing and hearing loss in one ear

People who experience these symptoms most likely do not have a brain tumor, but in case it is one, it is important to act quickly and not underestimate these symptoms. Therefore, a quick and inexpensive first screening can save lives, literally and economically, especially in countries such as the United States, where there is no free healthcare. A proposed solution for this are AI-based segmentation methods, which are becoming increasingly popular and show great results. [13]

Other AI tools beside tumor segmentation are already being used by radiologists. At the Kantonsspital Aarau (KSA) an AI tool is used to detect acute brain bleeding, serving as a second opinion for a radiologist to make a decision if they are unsure and have little time. [14]

2.3 Automated Image Segmentation

Automated image segmentation is a broad term that refers to the identification of organ or lesion pixels from medical images using code. In the case of brain tumor segmentation, the identification of healthy brain tissue and tumors. There are countless segmentation methods, those mentioned are divided into 3 main categories: Traditional segmentation methods, deep learning (DL)-based segmentation methods , and hybrid-based segmentation methods. [15]

2.3.1 Traditional Segmentation Methods

Traditional segmentation methods were the beginning of segmentation, and they were rather primitive, only being capable of using few segmentation criteria. Traditional segmentation methods were just algorithms and nowhere near what is considered an AI today. Most traditional segmentation methods use intensity as their main criterion; an example would be threshold-based segmentation.

Threshold-based segmentation, uses the fact that abnormalities show up brighter in MRI images. These abnormalities can be found by using global thresholding, or in cases where global thresholding does not yield usable results, local thresholding can be used. $I(x, y)$ is the intensity of the input image, $f(x, y)$ is the input image with the applied threshold, T_0 is the global threshold value. [13] The mathematical equation for the global thresholding algorithm is as follows:

$$f(x,y) = \begin{cases} one & : \quad I(x,y) > T_0 \\ 0 & : \quad otherwise \end{cases} \tag{2.1}$$

In simple terms, this equation finds an intensity ratio over the entire image and then sets areas with a significantly higher intensity to one and the other areas to 0. In the resulting image, only the significantly brighter parts of the image remain. Local thresholding works similarly, but it only

compares pixels to nearby pixels instead of the entire image. This can be useful when the image has a similar intensity and only a small threshold can be created. The mathematical equation is the same as the global threshold equation (2.1), only that the global threshold value T_0 is replaced by the local threshold value p_0. [13]

$$f(x,y) = \begin{cases} 1 & : \quad I(x,y) > p_0 \\ 0 & : \quad otherwise \end{cases} \tag{2.2}$$

$$where\ p_0 = f\left(T \in Np(x_p, y_p)\right). \tag{2.3}$$

Using this equation will result in an increased computational cost, as the intensity has to be calculated for every pixel, instead of the entire picture. Also, because thresholds vary, identical tissue may be treated differently if the context is different.

2.3.2 Machine Learning, Neural Networks, and Deep Learning

DL-segmentation models are the current standard for precise and robust image segmentation. DL is a subcategory of machine learning (ML), which is a subcategory of AI. This is visualized in Figure 3.

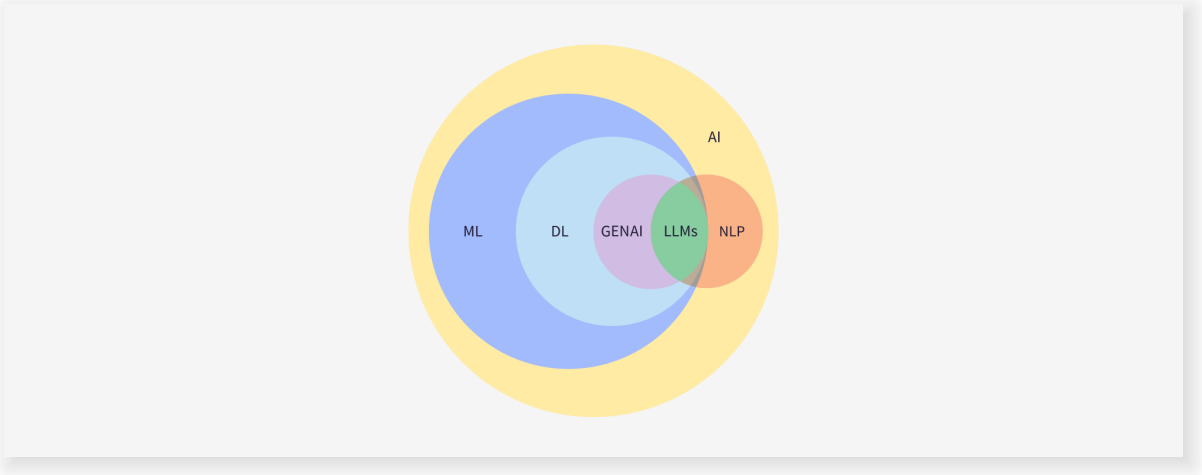


Figure 3: An overview of the connection between AI, machine learning, and deep learning. [16]

DL models are significantly more advanced than ML models, because of one huge advantage. While ML models have to be told which features to look for in an image, DL models will find those features themselves during training. This makes DL algorithms a lot more accurate and versatile. In training, DL models are given huge annotated datasets (solutions) and then they adjust countless internal parameters to obtain optimal results. In other words, they adjust the rules until the answers from the DL model match the given answers. [16]

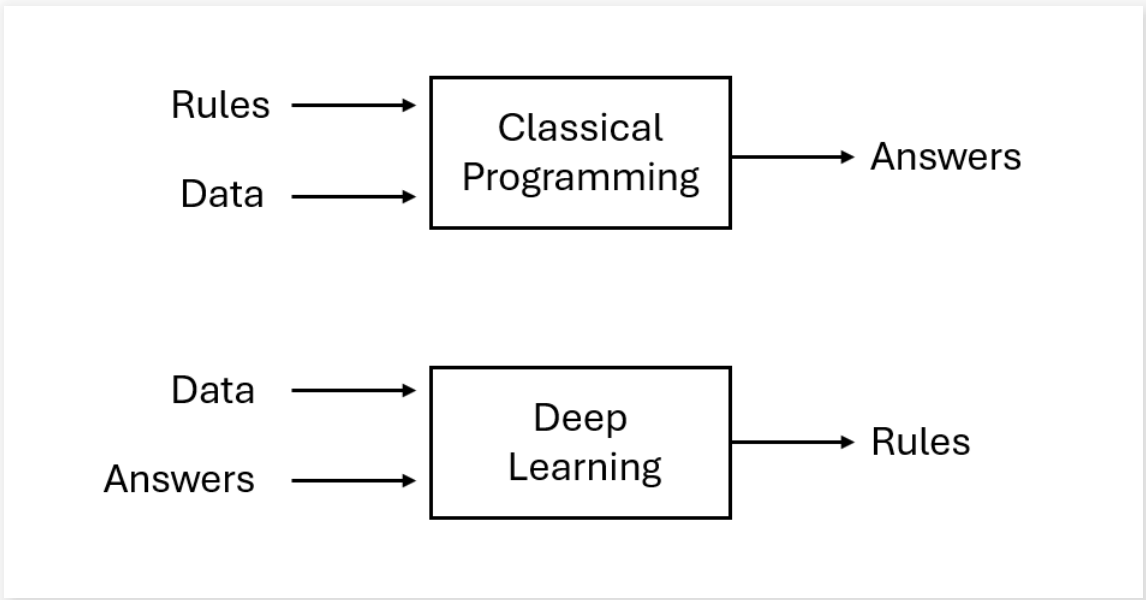


Figure 4: Deep learning vs. algorithms, altered. [17]

After the training is complete, the optimized internal parameters are saved as weights. These weights can be applied to the DL model and together they can accurately segment any image. To compare DL or any ML to traditional methods, which were just simple algorithms, Figure 4 is a good representation.

A **neural network** is an attempt to transforming the biological brain into code. A neural network has 3 layers, the input layer, the hidden layers, and the output layer. A common representation of neural networks is that of Figure 5.

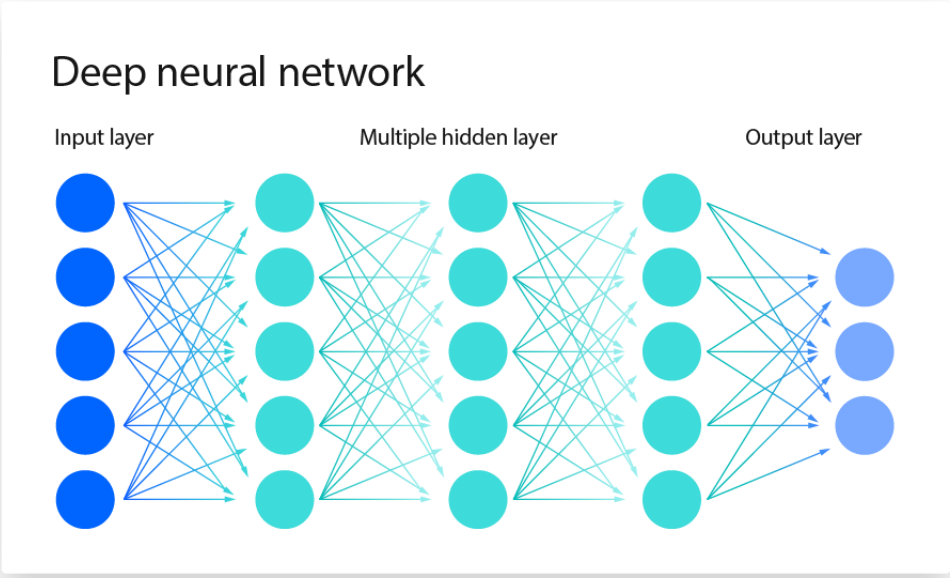


Figure 5: Structure of a neural network. [18]

Each dot in Figure 5 represents a neuron. Each neuron has an input, weights, and bias and then uses an activation function to determine if it gives an output. A neuron receives an input, it then multiplies the input by the weights, adds bias (bias is added to important signals resulting even in weak inputs to give a strong output), and then, after applying the activation function, sends the output to the next layer if. This is repeated until the signal reaches the output layer. The mathematical equation that each neuron computes is a follows [18]:

$$z = \sum_{i=1}^{n} w_i x_i + b \tag{2.4}$$

$$a = \sigma(z) \tag{2.5}$$

z is the result of each input for x_i being multiplied by its weight w_i, and then b is added to the sum of all inputs. This number can be anything, so the activation function $\sigma(z)$ is applied. A common activation function is the sigmoid function, which transforms the number for z into a number between zero and one, which can then be used as an input signal. This is important because you cannot know how strong a signal is if you have nothing to compare it to. [18]

2.3.3 Optimizers in Neural Networks

During training the neural network will learn the optimal values for each weight and bias using an optimizer. It learns these numbers by using a loss function. A loss function compares the loss with the value of the weights. Loss is calculated by measuring the difference between the model's prediction and the answers from the dataset. Depending on what the model is trained to do, a different loss function is applied. Common loss functions include BCE loss and dice loss [19]. A graph of a loss function will look like Figure 6. This graph contains only one single parameter. The graph for the overall loss of the model, which is used in training, is a polynomial function with multiple local minima and maxima. This makes finding the global minimum significantly harder.

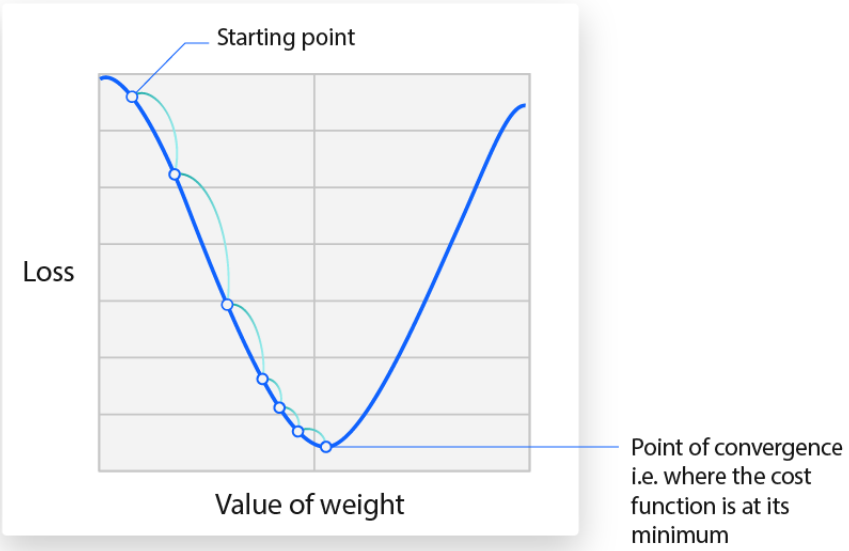


Figure 6: A loss function of a single neuron. [18]

The goal is to have as little loss as possible, so the minimum in the graph is chosen as the optimal value for the weights. Important to know is that there are no solutions for individual neurons, only for the output of the whole network. This means the network has to experiment with changing the parameters of each individual neuron to get as little loss as possible. [18] This optimization is what is done by an optimization algorithm during training, such as Adam. [18, 20]

Adam is a preferred optimizer as the algorithm intelligently adjusts the parameters based on the recent training history, leading to efficient training and great performance. The Adam optimizer remembers how much each parameter has changed and how the loss has changed as a result. Using this information, the algorithm updates the parameters more efficiently.

2.3.4 Modern Neural Networks

There are many versions of neural networks, but CNNs and especially FCNs are the most popular.

Both a CNN and a fully convolutional network (FCN) use **convolutional layers** to extract features from an image. A convolutional layer takes the input, either in the form of an image or in deeper layers, a feature map. These inputs have values; in colored images, there is one channel for every primary color, in MRIs grayscale images there is only one channel, and feature maps will have height, width and depth (depth is the value of a feature channel). Each convolutional layer has many filters (also called kernels) that convolve (slide) over the image simultaneously and take the dot product between the filter weights and the values of the input. This gives a single number for each position and filter, which is then combined to make a feature map. Each filter learns to detect features such as lines, curves etc. Essentially, the convolutional layer learns to filters all unnecessary information from an image or feature map. For example, if the goal is to find a bird, any features of a fish will not show up on the feature map. [21]

A CNN will have hierarchical layers consisting of a convolutional layer, an activation function, which decides if a neuron is to activate or not (in case of Figure 7 it is the rectified linear unit (ReLU) activation function), and then a pooling layer, which summarizes patches to lighten computation. Each deeper group of layers has a smaller window that convolves and learns to detect more detailed features. These layers are called an encoder, or as labeled in Figure 7, feature extraction. [15]

The output layer of a CNN, consists of one or more fully connected layers (meaning every neuron is connected to every neuron of the following layer), which flatten the feature map into a 1-dimensional vector and then interprets this value. There will be one value for each class the CNN has learned, in the case of Figure 7 the CNN has learned to detect horses, zebras and dogs. Whichever value is higher, is what the CNN decides is in the image, in the case of Figure 7 the CNN has decided that there is a zebra in the image. [15]

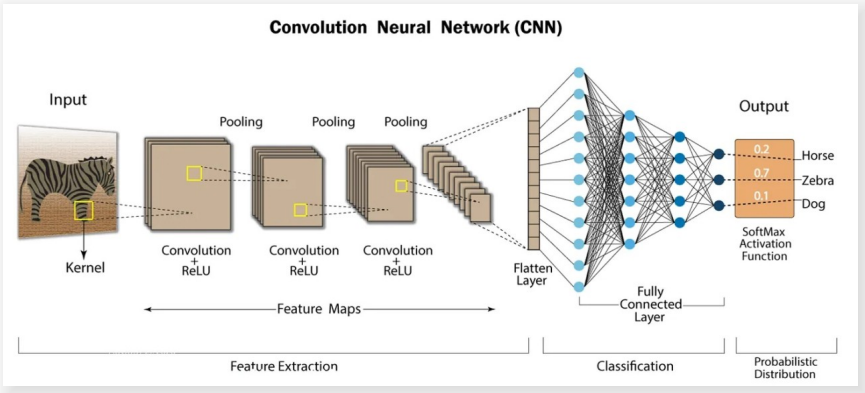


Figure 7: Structure of a convolutional neural network (CNN). [22]

An FCN is a type of CNN but has a major difference in its architecture, which allows it to do things a CNN cannot. A CNN cannot tell you where the thing you are looking for is. That is where an FCN comes in. An FCN also has an encoder, but once it reaches its lowest resolution (bottleneck), it will

start up sampling with a decoder, often using transpose convolutions. This sizes the miniature feature maps back up to the images original resolution. Encoding and decoding can be imagined as lenses. An encoder is a lens like a microscope, which allows it to zoom in to greater detail, while a decoder is a lens like a projector in a cinema, which increases the size of the image. Just like an encoder learns which features to recognize and give on, a decoder learns which features have to be reconstructed when zooming out. Now that the feature map is sized up to the original image size, the output layer will classify each pixel in the image, which gives the location of the desired object. In Figure 8 it is labeled as pixel-wise prediction. Unfortunately, the decoding loses detail and the location is not as accurate as one would hope. [15]

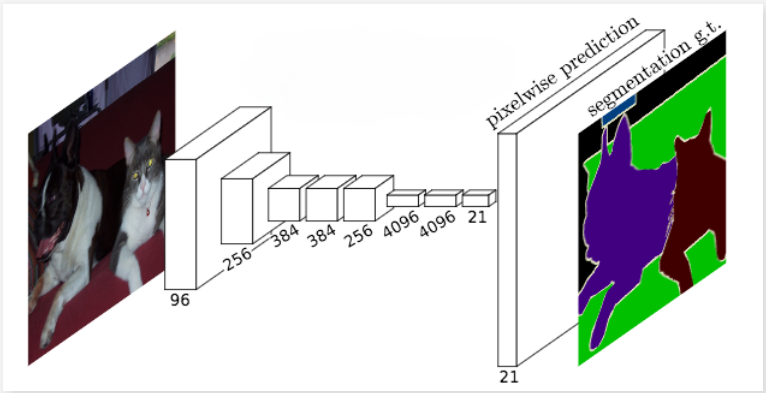


Figure 8: Structure of a fully convolutional network (FCN). [23]

2.3.5 Introduction to U-Net

U-Net is an extension of an FCN with a symmetrical U-shaped design, as seen in Figure 9. The encoder is a classic FCN, where in each encoded layer the convolved area is halved, and the number of channels is doubled (with that the number of feature maps is also doubled). This is indicated by the red arrows in Figure 9. The vital part that makes the U-Net better than a regular FCN, is that it uses concatenation[1]-based skip connections (in Figure 9 these are represented with gray arrows). Concatenation-based skip connections allow the U-Net to not lose any details during decoding. When the encoder sizes down by half, it saves a copy of the feature map, which it later combines with the feature map of the corresponding decoder layer. Because of skip connections the U-Net does not lose details when decoding, unlike a regular FCN, and with that can deliver a more precise segmentation. Concatenation-based skip connections are the most basic form of skip connections; they directly fuse the two feature maps without modifying the data. More complex skip connections will be introduced later. [15]

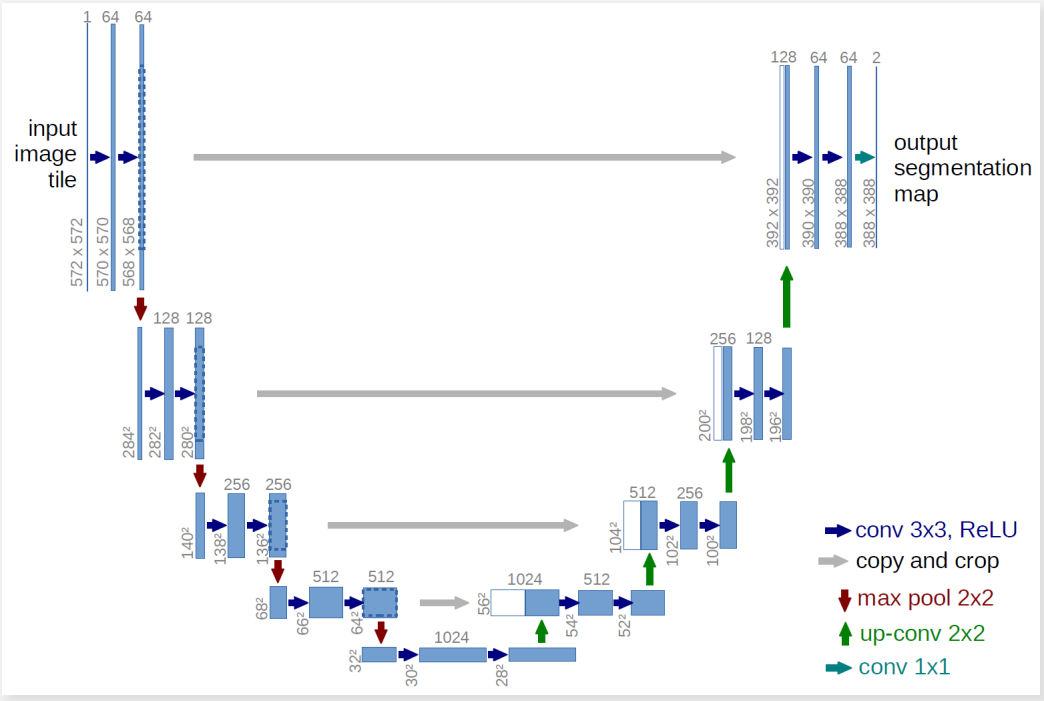


Figure 9: Structure of U-Net. [25]

In summary, a CNN can, for example, tell you if a bird is in the picture (it can classify an image), but it cannot tell you where the bird is. An FCN can tell you which pixels in the image belong to the bird (it can segment an image). However, because the resolution of the feature map is stretched from the size of the bottleneck to the size of the original image, many details are lost, and the delineation is not very accurate. The U-Net solves this issue of lost details by using skip connections, resulting in a very accurate delineation of the bird.

[1]Concatenate: to link something together. Definition by Merriam Webster [24]

2.4 Preprocessing

2.4.1 Importance of Preprocessing

Preprocessing is a crucial step in the training of any AI. It prepares the data in a manner that makes training efficient and effective. During preprocessing, noise is reduced, contrast and intensity is normalized, artifacts are minimized as much as possible, unusable images are discarded, and various other optional optimizations are Performed. Since MRI images are rarely perfect, these imperfections have to be reduced, because unlike a human, a machine cannot work around them, especially during training, when the weights are being adjusted. [26]

2.4.2 Significance of Skull and Eye Masking in MRI Images

A vital preprocessing step is to mask the skull and eyes to prevent the model from believing them to be tumors. As seen in Figure 10, the skull and the eye sockets have more characteristics of a tumor than healthy brain tissue; therefore, the model will likely label it as a tumor. This is particularly prominent in T2-weighted images where fluid-containing tissues appear bright, as seen in Figure 2)

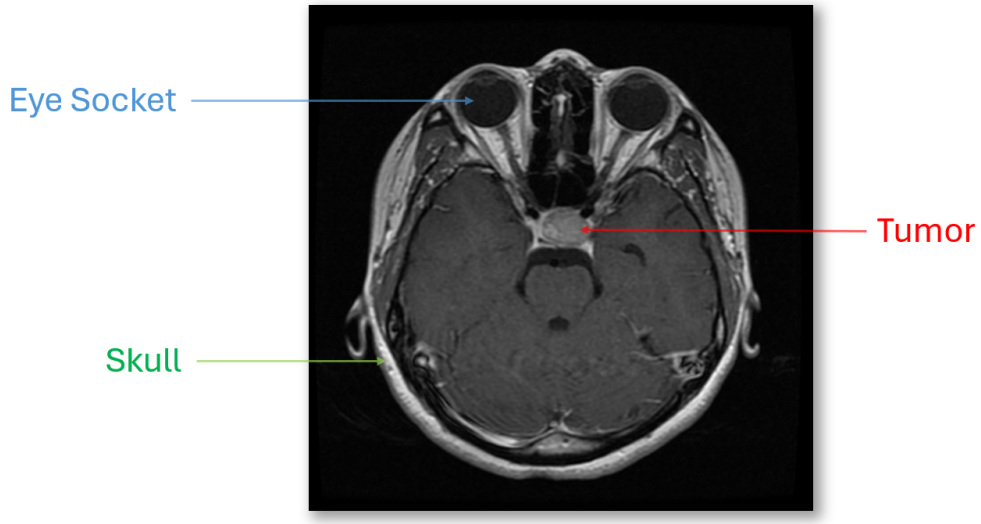


Figure 10: Axial MRI with tumor. [27]

2.5 Evaluation Metrics for Segmentation

The blackbox problem is a significant problem in the development of DL models . The blackbox problem means our inability to know 'how' and 'why' a DL model makes its choices. Only the final prediction from the output layers is visible, and what happens in the hidden layers is impossible to know. In theory, it is possible to obtain the data from each individual neuron and evaluate it, but that is impossible to do in a reasonable time frame. The solution to how developers can measure the performance of their model, and improve its weaknesses are evaluation metrics. [28]

2.5.1 Overview of Key Metrics (Dice, Sensitivity, Specificity)

Depending on which metric is chosen, the developers can gain insight into different aspects of their model. Most metrics will return a value between zero and one, where zero is the worst possible performance and one is a perfect performance. The most commonly used metrics are as follows:

- **IoU** (Jaccard score) compares the predicted mask and the ground truth mask. It works by dividing the overlapping area by their union. This is a common metric in general image segmentation and it shows how accurate the model's prediction is. With A being the area of the predicted mask and B being the ground truth mask, the mathematical equation is as follows [28]:

$$\text{IoU} = \frac{|A \cap B|}{|A \cup B|} \tag{2.6}$$

- **Dice Coefficient** is similar to the IoU but it is more forgiving if the model's prediction is not the same shape or size as the ground truth mask, as long as it is in the same area. The dice coefficient works by dividing double the overlapping area by the sum of the pixels in the prediction mask and the ground truth mask. With A being the area of the predicted mask and B being the ground truth mask, the mathematical equation is as follows [28]:

$$\text{Dice}(A, B) = \frac{2|A \cap B|}{|A| + |B|} \tag{2.7}$$

- **Precision** in itself is not a very interesting metric, but it can be used to calculate the F1 score, which will be addressed later. Precision is mathematically defined as the proportion between true positive pixels (pixels correctly predicted by the model) and the total number of predicted pixels. The mathematical formula is as follows [28]:

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} \tag{2.8}$$

- **Recall** is the other metric used to calculate the F1 score and measures the proportion of true positive pixels to the number of pixels in the ground truth mask. The mathematical formula is as follows [28]:

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \tag{2.9}$$

- The **F1 score** is the best metric to truly capture the model's performance. It demands accuracy because it penalizes false positives and false negatives by taking the harmonic mean of precision and recall. The F1 score is particularly useful in cases such as tumor segmentation, where the tumor class will have very few pixels compared to the non-tumor class. The mathematical formula is as follows [28]:

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{2.10}$$

- The **Hausdorff Distance** is another very useful metric in medical image segmentation, as it can be used to find the model's greatest segmentation error. It compares the predicted mask A and the ground truth mask B by finding the point in B that is farthest from any point in A, and then finds the point in A that is farthest from any point in B. The maximum of these two distances indicates the size the largest segmentation error is. In simple terms, the Hausdorff Distance indicates the largest distance between a pixel, which the model believes to be a tumor, and the nearest pixel that is actually the tumor. The Hausdorff Distance is used in models that need to have a very accurate bounding box, such as tumor segmentation models. The formula is as follows [28]:

$$d_H(A, B) = \max\left(\sup_{a \in A} \inf_{b \in B} d(a, b), \quad \sup_{b \in B} \inf_{a \in A} d(b, a)\right) \tag{2.11}$$

2.5.2 Meaning and Interpretation of Metrics for Medical Segmentation

In medical image segmentation, accuracy is crucial because false positives and false negatives can have severe clinical consequences. Using metrics, the developers can obtain a systematic and quantifiable assessment of the DL model's performance and how to optimize it. Although metrics do not solve the blackbox problem of DL models, they can reduce the prediction errors and improve the overall reliability of DL image segmentation models. [28]

3. Materials and Methods

3.1 Project Strategy

A clear strategy and plan were essential to completing a large project in the time frame of the Matura while balancing school and personal affairs. Therefore, a clear plan was established.

First, research was to be done. Analyzing existing studies to understand their methods and gaining foundational knowledge. Development of DL models is a very complex task; consequently, research was given a large time frame. The time frame had to be readjusted as the research showed itself to be even more complex than expected.

Once enough knowledge was gained, the first steps in development could begin. It was decided that first a dataset would be chosen and then a model would be developed to fit that dataset. After the dataset was found and the data refined, the development of the model could begin. Development was given a strict deadline, as there had to be sufficient time to evaluate the model and write the thesis.

3.2 Dataset and Data Preparation

Choosing the right dataset was a critical decision because the quality of the training dataset, is directly proportional to the quality of the model.

The first proposed idea for a dataset was to collect a series of available images from The Cancer Imaging Archive (TCIA) and label them myself. Labeling is the task of adding the classification masks, the ground truth mask from which the model learns. This can be done manually or using ML assisted tools. That idea was deemed infeasible as most images required extensive preprocessing steps to become viable training data, and manually labeling a dataset of thousands of images was not practical in the time frame of a Matura Thesis, especially when working alone.

Following a comprehensive online search, three candidate datasets were identified.

- The **Sartaj Bhuvaji** dataset, a collection of T1-weighted, T2-weighted and FLAIR images, combined there are 3264 .jpg files. They are divided into four classes. Gliomas, meningiomas, pituitary tumors, and no tumors. Additionally, they are divided into two folders, one for training and one for testing. The size of the images varies. This dataset is a well-maintained and widely used dataset, as of 04/10/2025, the dataset has 86.7 thousand downloads. [27]

- The **Masoud Nickparvar** dataset is an expansion of the Sartaj Bhuvaji dataset. Nickparvar removed images that he found to be incorrectly classified and replaced them with more images from figshare.com and the Br35H dataset. The dataset has a mix of T1-weighted, T2-weighted, and FLAIR images, combined there are 7022 .jpg files, which are divided into four classes of: Gliomas, Meningiomas, pituitary tumors, and no tumors. The four classes are divided into a training and testing folder. The size of the images varies. This is an even more widely used dataset, as of 04/10/2025 it has 145 thousand downloads. [29]
- The **Jun Cheng** dataset is a collection of 3064 T1-weighted contrast enhanced MRI images, stored as MATLAB (.mat) files. A MATLAB file from this dataset contains the class label (meningioma, glioma, or pituitary tumor) (there are no images that do not contain a tumor), the MRI image, a binary mask indicating the tumor region, and the tumor coordinates. The Cheng dataset has 140 thousand downloads as of 04/10/2025. [30]

All datasets were of good quality and a manageable size, however, the Cheng dataset stood out. Unlike the Nickparvar and Bhuvaji datasets, which provide only class labels for each image, the Cheng dataset provides a detailed binary mask indicating the exact location of the tumor. Ultimately, the Cheng dataset was chosen as the exact location of the tumor would allow more advanced metrics to be used.

3.2.1 Dataset Source and Description

The Cheng dataset contains 3064 T1-weighted contrast enhanced MRI images from 233 patients. The dataset contains three types of brain tumors: meningioma (708 MRI images), glioma (1426 MRI images), and pituitary tumor (930 MRI images). All images were acquired at Nanfang Hospital, Guangzhou, and General Hospital, Tianjin Medical University, China, between September 2005 and October 2010. The images were made using a Gd-DTPA contrast injection, which makes healthy tissue and abnormalities more distinct [31]. All images have a resolution of 512×512 pixels.

3.2.2 Data Format and Storage (MATLAB (.mat) files)

The dataset is stored as MATLAB files, a file type that allows images and associated data to be stored together. Each MATLAB file contains the MRI image, the tumor class label, patient ID (links all images to the same patient), the binary tumor mask, and the tumor boundary coordinates. [30]

There were no folders separating the training images from the testing images, this process was carried out by constructing a Python script. First 100 random images were moved to the testing folder, then the images in both folders were relabeled from 1-100 and 1-2964. This is to prevent any possible errors caused by missing numbers.

3.2.3 Preprocessing Steps

The Cheng dataset was already preprocessed; therefore, many preprocessing steps were no longer needed. Noise was corrected beforehand, and images with potentially hindering artifacts and duplicates were not found in the dataset. Due to the contrast-enhancing injection, artificial contrast enhancements using scripts were not necessary and would only reduce the details. All images were equal in size and

all labels were intact. However, some steps were still necessary. They were performed when loading each image in the training script.

After all relevant data is extracted from the MATLAB file (Lines 23-26 in Figure 11), the image is normalized (Lines 29-30 in Figure 11). Image normalization is used to adjust the intensity values of all images to a consistent scale. Having a common reference intensity range allows direct comparison by the model.This ensures that the same tissue type has a comparable intensity value across all images.

Then, a dimension is added to the image (Lines 33-34 in Figure 11), that shows the number of channels. The number of channels will be defined in the training loop, for grayscale images, it is one channel.

The image is then converted to a tensor (Lines 37-38 in Figure 11), which is the expected input format for the U-Net model. This transforms the image into computable numbers that the model can use.

Next, the image is resized to 256×256 (Lines 41-42, size defined in line 15 of Figure 11). This specific size is important because the dimensions of the image have to be a multiple of 2^n where n is the number of pooling layers in the model. [32] The used U-Net variation has four pooling layers; therefore, the number has to be a multiple of 16. 256 is a commonly used number as it balances low computational load and sufficient resolution for detailed learning, while being a multiple of 16.

Next, the image has a 50% chance to be horizontally flipped (lines 45-47 in Figure 11), this is to ensure the model does not overfit (Learning sections by heart instead of features) and to increase the variety of data without increasing the size of the dataset.

```
13  # Data preprocessing class for MATLAB files with basic normalization and resizing
14  class MatlabTumorDataset(Dataset):
15      def __init__(self, mat_dir, target_size=(256, 256)):
16          self.mat_files = [os.path.join(mat_dir, f) for f in os.listdir(mat_dir) if f.endswith('.mat')]
17          self.target_size = target_size
18
19      def __len__(self):
20          return len(self.mat_files)
21
22      def __getitem__(self, idx):
23          mat = mat73.loadmat(self.mat_files[idx])
24          cjdata = mat['cjdata']
25          image = np.array(cjdata['image'], dtype=np.float32)
26          mask = np.array(cjdata['tumorMask'], dtype=np.float32)
27
28          # Normalize image to 0-1
29          image = (image - image.min()) / (image.max() - image.min() + 1e-8)
30          mask = (mask > 0).astype(np.float32)  # Binary mask
31
32          # Add channel dimension to ensure data matches expected input shape of UNet [Channel, Height, Width]
33          image = np.expand_dims(image, axis=0)
34          mask = np.expand_dims(mask, axis=0)
35
36          # Convert to torch.Tensor before resizing
37          image = torch.from_numpy(image)
38          mask = torch.from_numpy(mask)
39
40          # Resize images and masks
41          image = TF.resize(image, self.target_size, interpolation=transforms.InterpolationMode.BILINEAR)
42          mask = TF.resize(mask, self.target_size, interpolation=transforms.InterpolationMode.NEAREST)
43
44          # Data augmentation (random horizontal flip to avoid overfitting and ge tmore data without increasing dataset size)
45          if torch.rand(1) < 0.5:
46              image = TF.hflip(image)
47              mask = TF.hflip(mask)
48
49          return image, mask
```

Figure 11: Basic data preprocessing class used in training.

3.2.4 Skull and Eye Region Masking

As mentioned in 2.4.2, masking the skull and eye region is an essential step in preprocessing, yet also a very difficult step. Many attempts were made to get a clean result, none were satisfactory.

The first attempt used inverted Otsu thresholding, which finds the largest connected mass and masks out anything that does not belong to it. This approach assumes a clear intensity separation between the brain and non-brain tissue, which is not true for MRI images. Also, the largest component may not be the brain; for example, it is possible that the skull and eyes are larger in certain slices.

Another attempt was to use High-Definition Brain Extraction Tool (HD-BET), a widely used DL-based skullstripping tool. However HD-BET depends on the Linux environment to function properly. Attempts to install and run it in a Windows environment resulted in compatibility issues, therefore HD-BET was not feasible. [33]

To avoid further delays and to conform to the tight time frame of the Matura, training was performed without applying skull stripping to the images.

3.3 Model Architecture and Rationale

3.3.1 Choice of U-Net Architecture for Segmentation

There were two candidate models, U-Net and You Only Look Once (YOLO). Both models have shown great results in medical image segmentation. YOLO and U-Net represent two distinct yet complementary approaches to image segmentation. [34]

U-Net excels at high -recision semantic segmentation by using its symmetrical encoder-decoder structure and skip connections to capture detail. The U-Net provides a very detailed segmentation, making it effective even for small datasets. The only downside is that it operates at a lower speed compared to YOLO. [34]

The YOLO model, originally designed for real-time object detection, delivers a rapid and accurate bounding box. However, in pixel wise segmentation YOLO is generally less detailed compared to U-Net. Therefore, there are hybrid models that attempt to combine the speed of YOLO with the accuracy of U-Net. One such model is the YOLO-Med, it combines both models by sharing an encoder for detailed feature extraction, but separate task-specific decoders. One decoder for quick detection with the YOLO structure and one decoder for the detailed U-Net-based segmentation. This allows the model to use the high accuracy of U-Net in real time. [34]

Performance comparisons in segmentation show that U-Net achieves better results in most metrics, such as accuracy, recall ,and precision, as seen in Figure 12 but YOLO, allows real-time application.

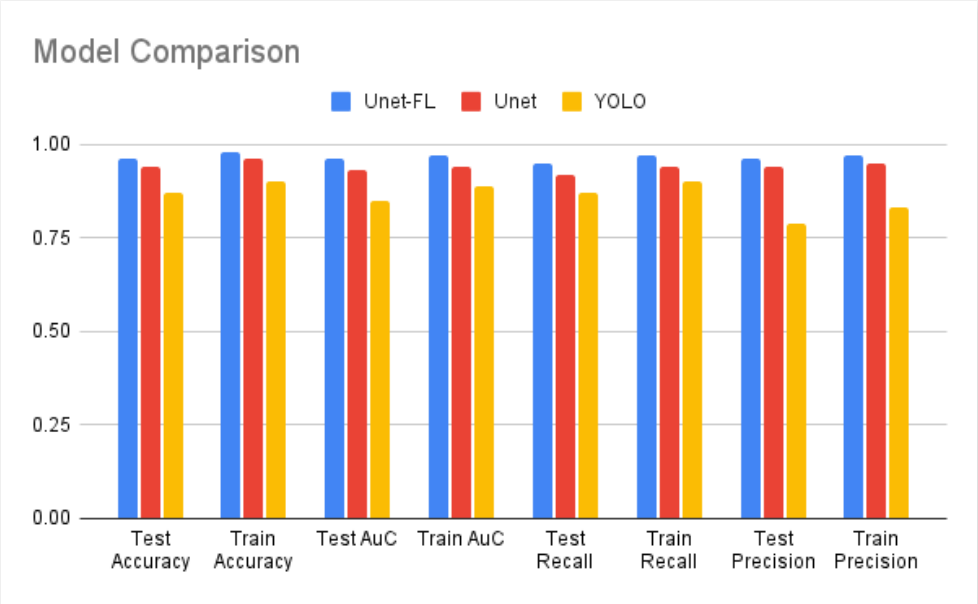


Figure 12: Comparison U-Net and YOLO. [35]

In medical image segmentation, achieving highly accurate and detailed segmentation is crucial for clinical relevance and patient safety. Therefore, model selection prioritized segmentation accuracy and boundary precision over segmentation speed, leading to the choice of U-Net over YOLO. Although more complex U-Net models and U-Net hybrid models were considered (such as YOLO-med, U-Net++, nnU-Net, U-Net 3+, and others), their integration and training complexity posed a risk of not completing development and training within the time frame of the Matura.

3.3.2 U-Net Structure and Hyperparameter Decisions

Milesial converted the original U-Net structure of Ronneberger et al. [25] into Python using the PyTorch library and uploaded it to GitHub. [36]. The PyTorch U-Net was then simplified to include only essential components and combined into one python script. This was done to avoid unnecessary complexity and more effective optimization. The adapted PyTorch U-Net model resembles the model in Figure 9, it has four downsampling layers, a bottleneck layer, and four upsampling layers. (All mentioned code can be found in Appendix E or on GitHub)

The full model version later used in NOSAv2 used 64 base filters; this is defined in Line 71 in Figure 29, and the filters were doubled at the bottleneck (Line 90 in Figure 29). This resulted in a maximum of 1024 layers in the bottleneck.

The first version later used in NOSAv1 was reduced in size to reduce the computational cost and training time. In Figure 29 in line 71, the base filters were reduced to 32 and in line 90, the base filters were not doubled from the previous layer (base_filters * 8 instead of base_filters * 16).

Each layer uses the double convolutional block, as defined in Figure 27. The block first applies a 2-dimensional convolution with a 3×3 kernel, and padding is added to preserve the spatial dimensions, ensuring that the output feature map has the same size as the input. Each batch of processed feature maps is then normalized and the ReLU activation function is applied. The ReLU activation function is different from the sigmoid activation function explained in section 2.3.2 in that ReLU outputs zero for all negative inputs and leaves positive values unchanged. This makes the ReLU function computationally simpler and more efficient. ReLU is considered to be the best activation function for DL models as it also mitigates the vanishing gradient problem, which activation functions such as the sigmoid have. The vanishing gradient problem is an issue during training where the gradient used to update the weights becomes increasingly small. This hinders the learning of shallow layers or might stop it altogether. [37]

This sequence of 2-dimensional convolution, batch normalization and the ReLU activation function is repeated twice, therefore the name double convolutional block, this allows the model to learn more sophisticated features.

Finally, dropout regularization is applied, which randomly disables channels (in this model the chance is set to 10% as can be seen in Line 54 of Figure 27) and forces the model to use different channels to reach a decision. Dropout is a common tool against overfitting and results in a more robust and versatile model.

3.3.3 Output Layer Design

The output layer in line 104 of Figure 29 shows how a standard 1×1 convolving output layer was used. This output layer was introduced in section 2.3.4 in the paragraph about FCNs.

3.4 Training Procedure

3.4.1 Loss Function Selection (Dice Loss vs. BCE Loss)

A combined loss function was used that integrates both dice loss and BCE loss. This loss function seen in Figure 28. The BCE loss is implemented via PyTorch's BCEWithLogitsLoss as seen in Line 153 of Figure 28. BCE loss measures the pixel-wise accuracy by comparing the model's prediction with the ground truth mask. This can be a problem when segmenting something small since BCE loss treats all pixels equally. In extreme cases, this might lead to the model learning to classify all pixels as no tumor, as overall, most pixels will be correctly classified. [19]

The first model used in NOSAv1 was trained using only BCE loss, dice loss was added to train the full model to gain a more accurate segmentation of the tumor. Dice loss does not include background in it's formula, it directly compares the predicted mask with the ground truth mask.

Dice loss is derived from the dice Coefficient, which was introduced in section 2.5.1. Dice loss uses the following formula: Dice Loss = 1 - Dice Coefficient, or:

$$\text{Dice loss} = 1 - \frac{2|A \cap B|}{|A| + |B|} \tag{3.1}$$

3.4.2 Optimization Algorithm and Learning Rate Settings

An Adam optimizer is used for training in both model versions (Line 182 in Figure 30), an algorithm introduced in section 2.3.3, to optimize the model's parameters (weights and bias) with the goal of minimizing loss.

The initial learning rate is set to 1×10^{-3} (Line 171 in Figure 30), a common rate, which is the amount the parameters are adjusted at a time. A cosine annealing scheduler was used to decrease the learning rate over time; this allows for more detailed adjustments during the later stages of training.

Automatic mixed precision is implemented, which increases the training speed by using the computers graphics processing unit (GPU) (Line 186 in Figure 30)

3.4.3 Batch Size Considerations, Number of Epochs, and Hardware Constraints

A batch size of 32 was chosen, balanced to maximize GPU memory usage. The batch size is the number of images processed before the weights are updated. The number of epochs (96) was chosen as an estimate when the loss would no longer improve per epoch (The loss / Epoch graph flattens out). One epoch is when the model has viewed each image in the training data set once. Additionally, the computational constraints would not allow for a training period longer than 5 hours (More epochs result in longer training time).

An NVidia Geforce RTX 3060 was used in training, a solid mid-range GPU, but it lacks memory for larger models. To further maximize the GPU's processing power, a data loader was used (Line 174 in Figure 30), to minimize idle time during training. The batch size and the number of epochs were the same for both model versions.

3.4.4 Early Stopping Criteria

With a greater number of epochs, a stopping criteria could be implemented based on performance metrics to prevent overfitting, but this was deemed unnecessary during development.

Overfitting occurs when the model learns the data too well instead of the general features that it can use to segment new data. This will become visible if the model performs significantly better in training than with test data. [38]

3.4.5 Use of Data Augmentation in Training and Validation

As mentioned in section 3.2.3, random horizontal flipping was applied, and as mentioned in 3.3.2, dropout regularization was applied. These tools prevent the model from relying only on a select few neurons, which makes the model train all available neurons, resulting in a more robust and versatile model that responds better to new data on which it was not trained.

3.5 Handling of False Positives and Post-Processing

3.5.1 Masking of Skull and Eyes During Training vs. Inference

After the training was completed faster than expected, the additional time allowed the skull stripping issue to be revisited. A heuristic approach was used to reduce false positives in the skull and eye area. By adding all ground truth masks of the dataset, a probabilistic map was created, which showed how likely it was for a tumor to be in a certain area. This mask, named heatmap, would be multiplied with the prediction mask in post-processing and with that remove predictions in areas outside the brain. The mask is a gradient mask (zero to one) and not a binary mask(zero or one). This is to make the mask deal in possibilities instead of absolutes.

3.6 Addition of a Graphical User Interface (GUI)

3.6.1 Purpose of a GUI

The purpose of a GUI is to allow a user to interact with programs or devices using graphical elements rather than command line inputs or manually altering the source code. Adding a GUI to the model facilitates practical implementation by providing a comprehensive interface, hiding the source codes complexity and presenting only relevant options to users. A well-designed GUI significantly improves usability and allows a visual representation of data. A GUI was deemed necessary to let users efficiently review, validate, and manage the segmentation tasks.

3.6.2 Design and Architecture

The goal was to create a GUI that allows the user to upload MRI images and use the trained model for brain tumor segmentation. Providing the possibility of changing all relevant parameters was essential to optimize segmentation and allow users to tailor the model to specific cases. Core features to enable this included:

- Uploading MRI images from a user's computer, visualizing said MRI image, and being able to replace the MRI image at any time.
- Visualizing the predicted mask over the original MRI image.
- A method to enable or disable the heatmap. Because only a single map is created for three different MRI perspectives, the map is not always aligned with the shape of each MRI image. The heatmap can be disabled, if necessary, to avoid misalignment or falsifying the true positives.

- A method to adjust the strength of the heatmap by modifying the threshold required to classify an area as a tumor after the heatmap is applied. Providing control over the sensitivity of the segmentation results. This is important because both model versions lack reliability. By adjusting the strength, false positives can be managed.
- The option to display the ground truth mask, if available, for optimization and performance analysis.

A blue and white color theme was chosen for the GUI, as blue evokes a feeling of trust, calmness, and professionalism, while white evokes a feeling of a clean and sterile environment. A custom tool bar was added for better interactions with the GUI.

The program (the model equipped with the GUI) was named NOSA. Candidate names that more accurately demonstrated the connection to U-Net were considered, such as U-NOSA, the concise and professional-sounding acronym NOSA was ultimately preferred for its memorability.

3.6.3 Implementation details

The program architecture follows a modular design with three core components: NOSA_GUI.py serves as the main interface, NOSA.py contains the inference engine (component that uses the U-Net (U-Net) model to reach a decision), and NOSA_def.py which defines the U-Net model's architecture. This separation allows a simple updating procedure if a new version is developed, as was done with the full model version. The program featuring the full version was named NOSAv2, the program featuring the prototype model is named NOSAv2.

In NOSA.py, a function named predict tumor is defined. First, the program ensures that if a cuda capable GPU is available, it is used instead of the central processing unit (CPU). Then, the U-Net model is loaded as well as the trained weights. The model is set to the evaluation mode, which disables dropout and batch normalization.

The loaded image is a variable that the user can change using the GUI, the image is preprocessed and transformed into a tensor, then the model decides on the probability for every area of being a tumor. The threshold of 70% is applied, meaning that any area with a probability of less than 70% will become zero, and any area with a probability of 70% or more will become 1. This creates a binary mask that is then returned to the GUI.

Advanced post-processing features can be enabled using the GUI, such as the heatmap, which multiplies the predicted mask by its probability gradient. This results in a predicted mask that does not show predictions in areas such as the eyes or skull. The strength of the heatmap can be adjusted using an interactive slider.

The system implements robust error handling for file loading and provides the user with error messages for invalid file types and missing components.

3.6.4 Software Frameworks and Libraries Used

The GUI was built using the **PyQt5** library, a versatile Python library that allows developers to efficiently build GUIs. PyQt5 provides a wide variety of widgets and tools, such as the implementation of a custom tool bar, featuring the name, NOSA and the current version. [39]

The U-Net architecture is made using the **PyTorch** library, a powerful and commonly used framework for deep learning, containing all necessary commands. The PyTorch library supports GPU acceleration, resulting in faster inference. [40]

The program also uses the **mat73** library to process MATLAB files and extract all relevant data. [41]

The **NumPy** library is used for efficient numerical operations and array manipulations, such as normalization and thresholding. NumPy is particularly used in pre- and post-processing. [42]

3.7 Comparison with State-of-the-Art Models

3.7.1 Baseline U-Net

The baseline U-Net that is used for comparison, is defined in the paper "Automated Brain Tumor MRI Segmentation Using Advanced Residual U-Net (ARU-Net) with Residual-Attention Modules" by Erdal and Feyza Özbay [43]. The baseline U-Net used a mostly identical architecture to the standard U-Net structure used by NOSAv2, but with 3-dimensional volumetric scans instead of 2-dimensional MRI slices. It used a standard encoder-decoder structure, basic concatenation-based skip connections, and a combination of dice loss and Cross-Entropy loss. The model was trained to not only distinguish healthy tissue from tumor tissue but also to determine the type of tumor it is. This model was chosen to show the potential of the standard U-Net model also used in NOSAv2, but with greater computational resources and a larger time frame. [43]

A notable difference was the dataset it was trained with. It was trained using the Brain Tumor Segmentation Challenge (BraTS) 2022 dataset and a custom hospital dataset. This dataset is significantly larger than the one used to train the proposed model, as it has more than 2000 cases, compared to 233 cases. The dataset used combined of T1-weighted images, T1-weighted contrast enhanced images, T2-weighted images, and FLAIR images. Another difference was that the dataset contained healthy cases, where there was no tumor to find. The BraTS datasets are up to the clinical standard, yet too large to process within the time frame of the Matura. [43]

The training methods are assumed to be the same as for the ARU-Net in Figure 13. The figure comparing the training methods of the state-of-the-art models can be found in section 3.7.5.

3.7.2 ARU-Net

The ARU-Net is an advanced model with good results, it was chosen to show the possibilities of U-Net based segmentation. The ARU-Net enhances the U-Net structure with Residual blocks throughout the networks, which allows for a deeper structure; the exact number of layers is not stated. Residual

blocks enable the model to learn the difference (called the residual) between the input and the desired output, allowing the model to learn small changes instead of entire transformations. [43]

The ARU-Net uses advanced skip connections (residual connections) that share the information from the residual blocks across layers, and with that allows the model to learn finer details. Residual connections stabilize DL models with many layers by solving the vanishing gradient problem and simplifying optimization. [43]

Additionally the ARU-Net uses residual attention-guided feature fusion, which combines important features from different layers to better learn which regions are relevant to tumors. Attention modules are used for various tasks to filter out unimportant information. When features related to tumors are found in shallow layers , the deeper layers will learn to pay more attention to these areas. Using attention modules results in more efficient training, as the model learns to pay less attention to unimportant data. [43]

The ARU-Net was trained on the same dataset as the Baseline U-Net, a combination of the BraTS 2020 dataset and a custom hospital dataset. [43]

The training methods can be seen in Figure 13 in section 3.7.5.

3.7.3 EfficientNetB4+Multi-Attention U-Net

The EfficientNetB4+Multi-Attention U-Net is a hybrid model that combines the pretrained CNN EfficientNetB4 as the encoder, with a Multi-Attention U-Net as the decoder. This hybrid model was chosen to show the possibilities of hybrid models and their complexity, as they weave together different approaches to achieve optimal results. This model operates with 3-dimensional MRI images. EfficientNetB4 was trained on millions images (ImageNet), not related to tumor segmentation. The EfficientNetB4 as a backbone uses this general knowledge to better extract important features from MRI images. This process is called transfer learning and gives the model a head start, as it does not have to begin learning from zero. [44]

The model uses a Multi-Attention U-Net as the decoder. This advanced U-Net decoder uses Multi-level attention mechanisms that allow the model to focus on relevant features. The multi-level attention mechanisms apply multiple attention mechanisms simultaneously, compared to only one in the residual attention mechanism. The used attention mechanisms include a spatial attention module (SAM) (where important features are) and a channel attention module (CAM) (which features are important), and they are applied across layers. The model then effectively combines the information from upsampling and the feature maps through skip connections, which were refined by the multi-level attention modules to more accurately segment tumors. [44]

The EfficientNetB4+Multi-Attention U-Net model was trained on a combined dataset of the BraTS 2018-2022 datasets, data from The Cancer Genome Atlas (TCGA)-GBM, and data from TCGA-lower grade gliomas (LGG). This combination creates a large, diverse dataset with over 1500 cases. The dataset contains T1-weighted images, T1-weighted contrast enhanced images, T2-weighted images,

and FLAIR images collected from countless different institutions. This dataset also includes healthy cases, where there is no tumor to segment. [44]

The training methods can be seen in Figure 13 in section 3.7.5.

3.7.4 Enhanced U-Net

The enhanced U-Net improves the U-Net architecture for better MRI segmentation, by enhancing feature extraction and feature fusion. This model was chosen as an alternative to ARU-Net, as the model uses a similar architecture without residual learning. [45]

This model utilizes dense connections within blocks to enable feature reuse and with that learn detailed features more effectively. Multiple attention mechanisms were implemented to refine feature masks in upsampling and in skip connections, such as a SAM, a CAM, and a positional attention model (PAM) (where important features are considered in context). Multilayer feature fusion is used to combine feature maps across layers, resulting in more detailed feature maps. [45]

The Enhanced U-Net was trained using a combination of BraTS 2020/2021, with a total of 494 cases. The dataset used combined of T1-weighted images, contrast enhanced T1-weighted images, T2-weighted images, and FLAIR images. The dataset contained healthy cases, where there was no tumor to find. The multiclass annotation of the BraTS datasets allows the models to differentiate between the tumor core, the whole tumor, and the enhancing tumor (the most malignant part of the tumor) and classify each area as such. [45]

The training methods can be seen in Figure 13 in section 3.7.5.

3.7.5 Training Methods for State-of-the-Art Models

The state-of-the-art models use significantly more sophisticated training methods. This causes significantly higher training time, computational costs, and increased development complexity. However, the quality of training is greatly increased, which results in better performance, as can be seen in the results in section 4.4. In NOSAv2 the simplest version of each aspect is applied due to the limited time frame, limited computational resources, and limited experience working with highly complex models.

Aspect	NOSAv2	ARU-Net	EfficientNetB4+MA	Enhanced U-Net
Loss Function	Dice + BCE	Dice + Focal	Weighted BCE + Dice + IoU	Dice + BCE + Boundary
Optimizer	Adam (1e-3)	AdamW with weight decay	Adam with scheduling	AdamW with momentum
Scheduler	Cosine Annealing	Step decay	Polynomial decay	Cosine restart
Data Augmentation	Horizontal flip	Rotation, scaling, elastic	comprehensive augmentation suite	Multi-scale augmentation
Batch Size	32	16-24	8-16	16-32
Epochs	96	200+	150-300	200+

Figure 13: A table showing the training methods used for the different Models. [43, 44, 45]

4. Presentation of Results

4.1 Showcasing of NOSAv2

Upon starting NOSAv2, a window is opened, as can be seen in Figure 14. A message lets the user know that there is no image currently loaded. To begin segmentation, the user must upload an image from their computer. This can be done with the dark blue button labeled "Load Image (.mat)". The added (.mat) is to let the user know which format is required. To ensure the correct file type is chosen, only .mat files will be shown in the file explorer.

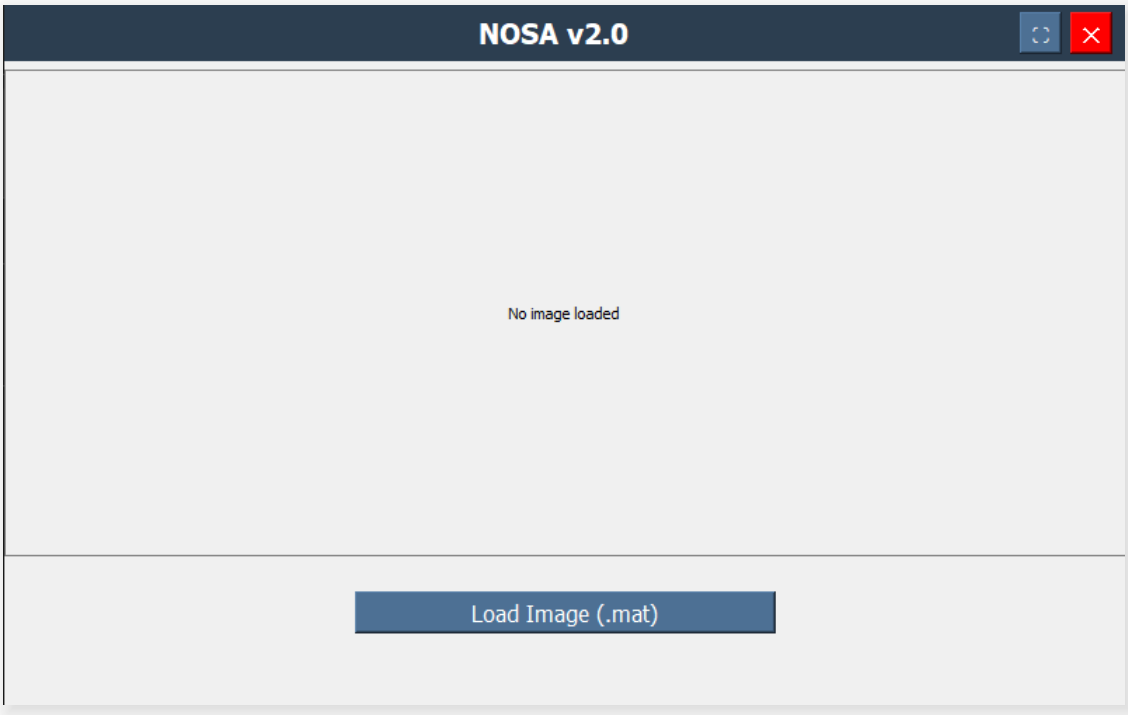


Figure 14: NOSA v2.0 starting page.

Once the user has loaded an image, it is displayed where the no image loaded message was located before, as can be seen in Figure 15. Underneath the image, the file path of the image is displayed to ensure that the user has loaded the right image. Two new buttons are available under the "Load Image" button. A burgundy button, labeled "Find Tumor", and a teal button, labeled "Show Ground Truth Mask". Under the buttons, a checkbox allows the user to enable or disable the heatmap to reduce false positives in the skull and eye regions.

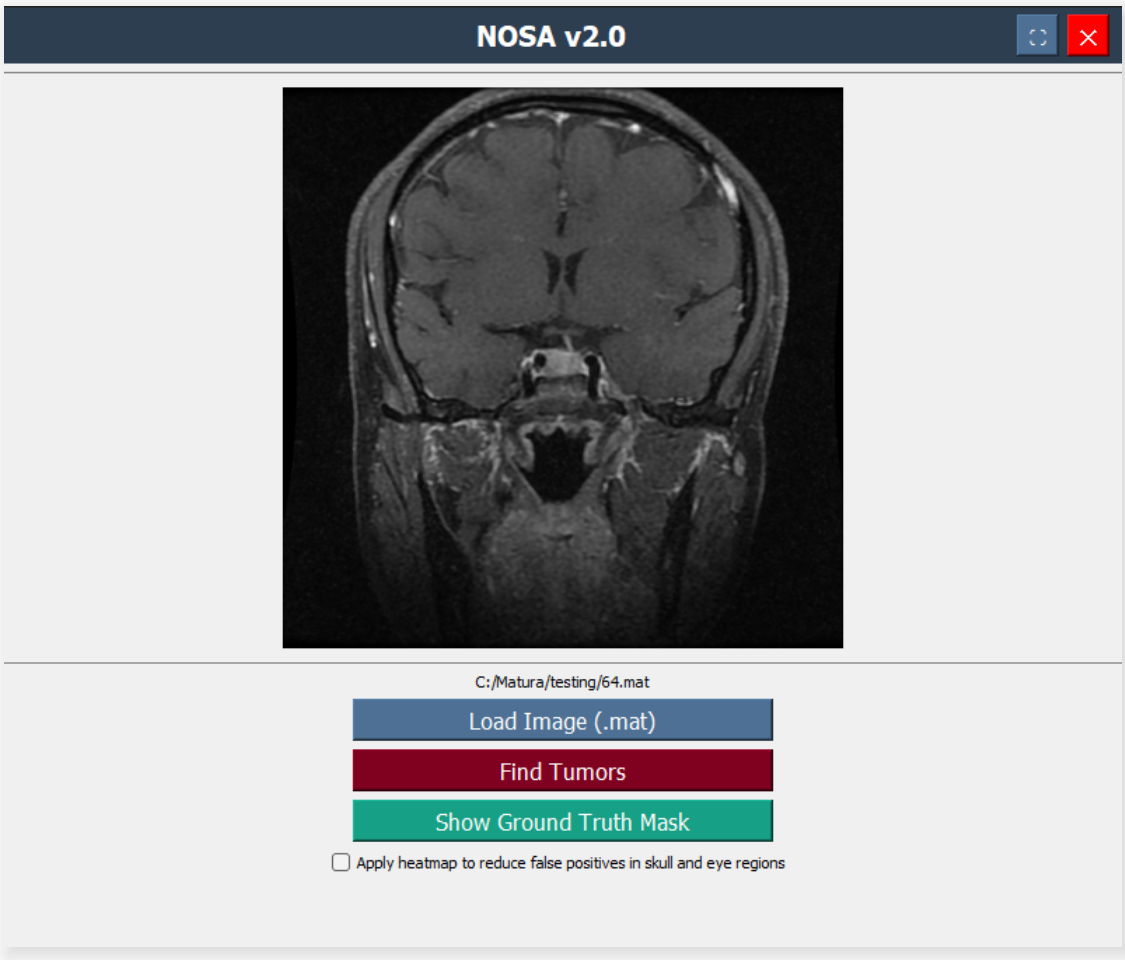


Figure 15: NOSA v2.0 with an image loaded.

Once the user presses the "Find Tumors" button, the model will compute a prediction mask, which will be loaded and visualized over the image. The red area in Figure 16 is the area which the inference engine in NOSAv2 predicts a tumor to be. To visually assess the model's performance, the user is presented with the option to show the ground truth mask. If the user presses the button to show it, the ground truth mask will be visualized in green, as can be seen in Figure 17.

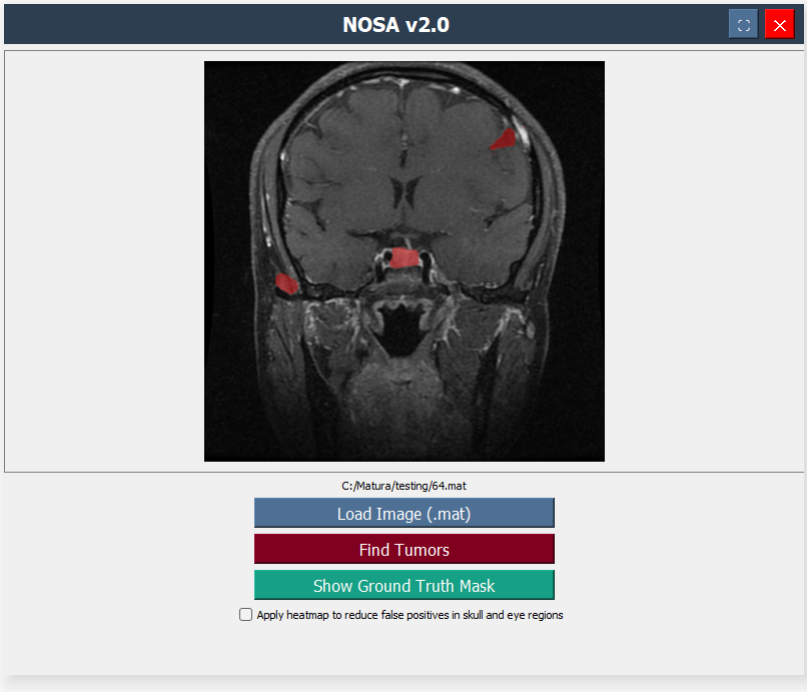


Figure 16: NOSA v2.0 with the predicted tumor mask visualized in red.

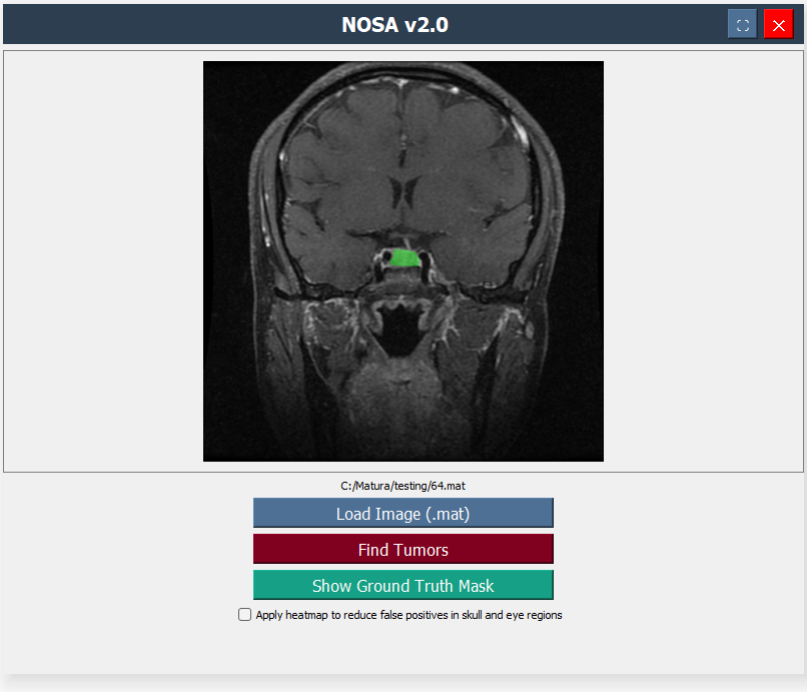


Figure 17: NOSA v2.0 with the ground truth mask visualized in green.

If the user decides to use the heatmap by marking the checkbox (as done in Figure 18), a slider will appear, labeled "strength". By default it is set to 70% but the user can adjust the value by interacting with the slider. By increasing the strength, only the predicted tumors in common locations will be shown. Furthermore, by decreasing the strength, all predicted tumors will be shown. To apply the changes to the heatmap, the user has to press the "Find Tumor" button again. Out of the original prediction mask in Figure 16, only predicted tumors in areas where tumors commonly occur are visible in Figure 18.

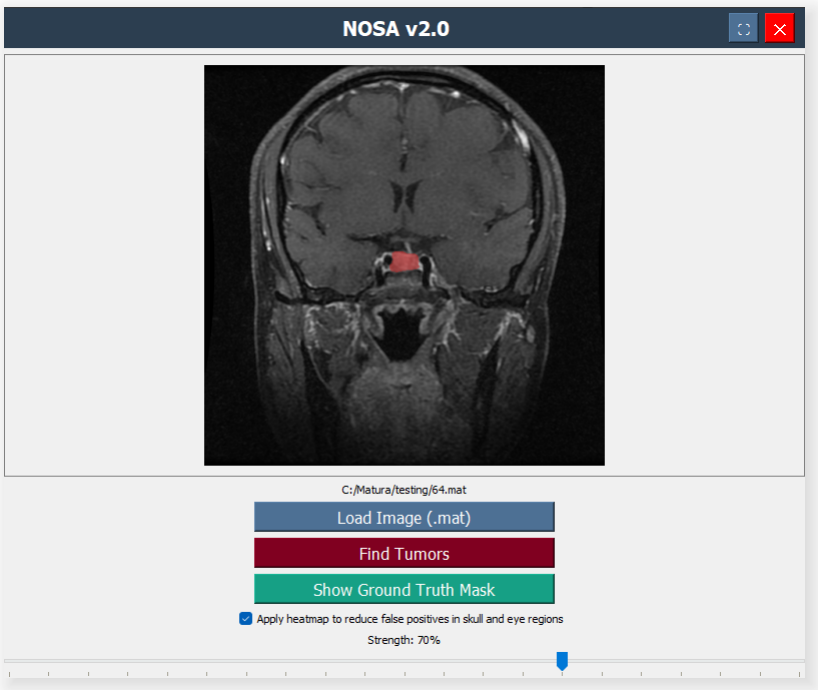


Figure 18: NOSA v2.0 with the heatmap enabled. The modified prediction mask is visualized in red.

4.2 Visual Results

As can be seen when comparing the red predicted mask in Figure 16 and the green ground truth mask in Figure 17, NOSAv2 finds the tumor with near no visible errors, except for the false positives in the skull area. However, with the heatmap applied, the false positives are no longer shown, as can be seen in Figure 18.

4.3 Quantitative Results (Metrics such as Dice, Sensitivity, etc.)

Technically the programs NOSAv1 and NOSAv2 include everything ranging from GUI to inference engine, but for simplicity reasons, the model versions used in NOSAv1 and NOSAv2 will be addressed as NOSAv1 and NOSAv2.

4.3.1 Training Performance

The following graphs compare the training performance of NOSAv1 and NOSAv2. The first graph labeled Training Loss Trajectories shows the learning curve of both models by using loss per epoch. A lower score is better. NOSAv1 has a 13.4% higher loss in the first epoch compared to NOSAv2, then both have a steep initial drop and flatten out. NOSAv2 achieves a 22.3% lower final loss and a 0.9% better total improvement (total improvement is the percentage the model improved compared to the first epoch) than NOSAv1, as can be seen in Figure 19.

The next graph labeled Learning Dynamics shows the change (delta) in loss per epoch. A high score means a large reduction in loss. NOSAv1 and NOSAv2 have a mostly overlapping graph, with an exception at the beginning, where NOSAv1 has a significantly higher maximum. When the change in loss is no longer significant per epoch, the model is considered to have converged (defined as less than 0.005 change over 15 epochs was used in Figure 19). NOSAv2 converges 10 epochs later than NOSAv1.

The Performance at Key Training Milestones shows the same as the Training Loss Trajectory, but enables a quantitative comparison by showing exact numbers.

Finally, there is a table that shows key statistics that summarizes all key data. Learning efficiency is the average reduction in loss per epoch, a high efficiency means the model learns quickly, but not necessarily better. This can be seen at the bottom right of Figure 19 in the following page.

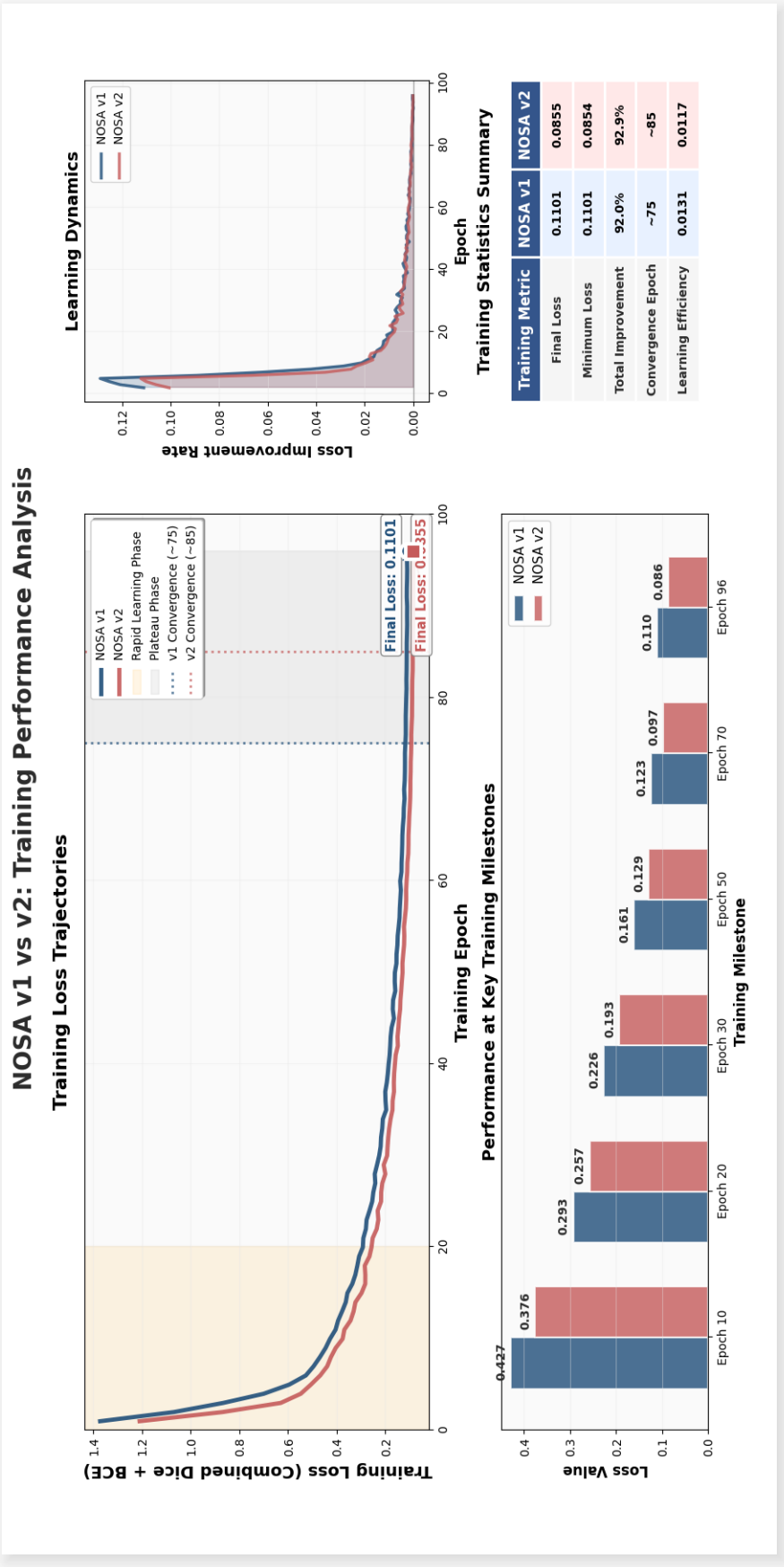


Figure 19: Comparison of NOSAv1's and NOSAv2's loss per epoch. This graph was created using a script written by Microsoft's Copilot.

4.3.2 Performance on the Test Dataset

The line graph in Figure 20 and the boxplot in Figure 21 (in the following page) compare the performance of NOSAv1 and NOSAv2 using common metrics on 100 test images. The python script used to compare both model versions was written by Microsoft's Copilot, and can be found in the GitHub repository. The test images are images on which the model has not been trained on. The F1 and Jaccard scores are slightly better for NOSAv2 (+2.7% and +2.1% as can be seen in Figure 20), but the interquartile ranges largely overlap and both models have a wide range of scores (this can be seen in Figure 21 in the following page). Precision in NOSAv2 is significantly lower compared to NOSAv1 (-8.2% in Figure 20), but the recall score showed a significant improvement (+9.1% in Figure 20). Overall the average increase in performance is 1.45% on test data.

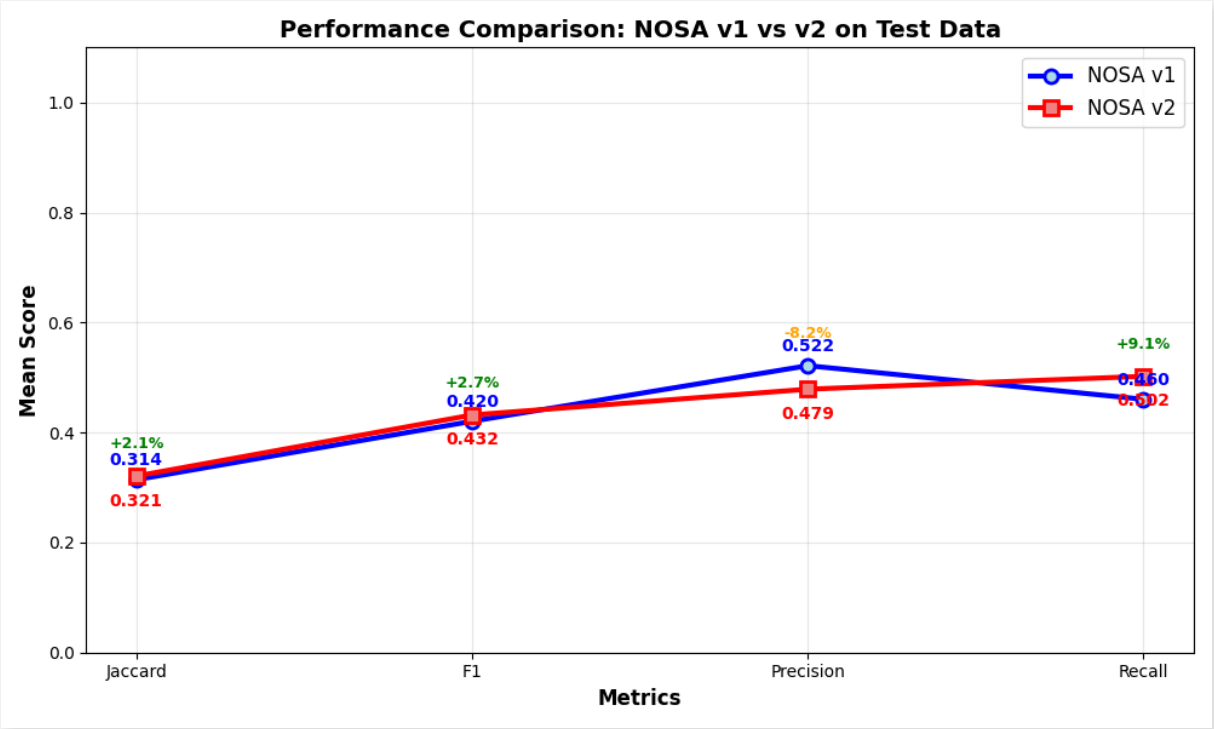


Figure 20: Comparison of NOSAv1's and NOSAv2's mean score in common metrics on test data. This graph was created using a performance comparison script written by Microsoft's Copilot.

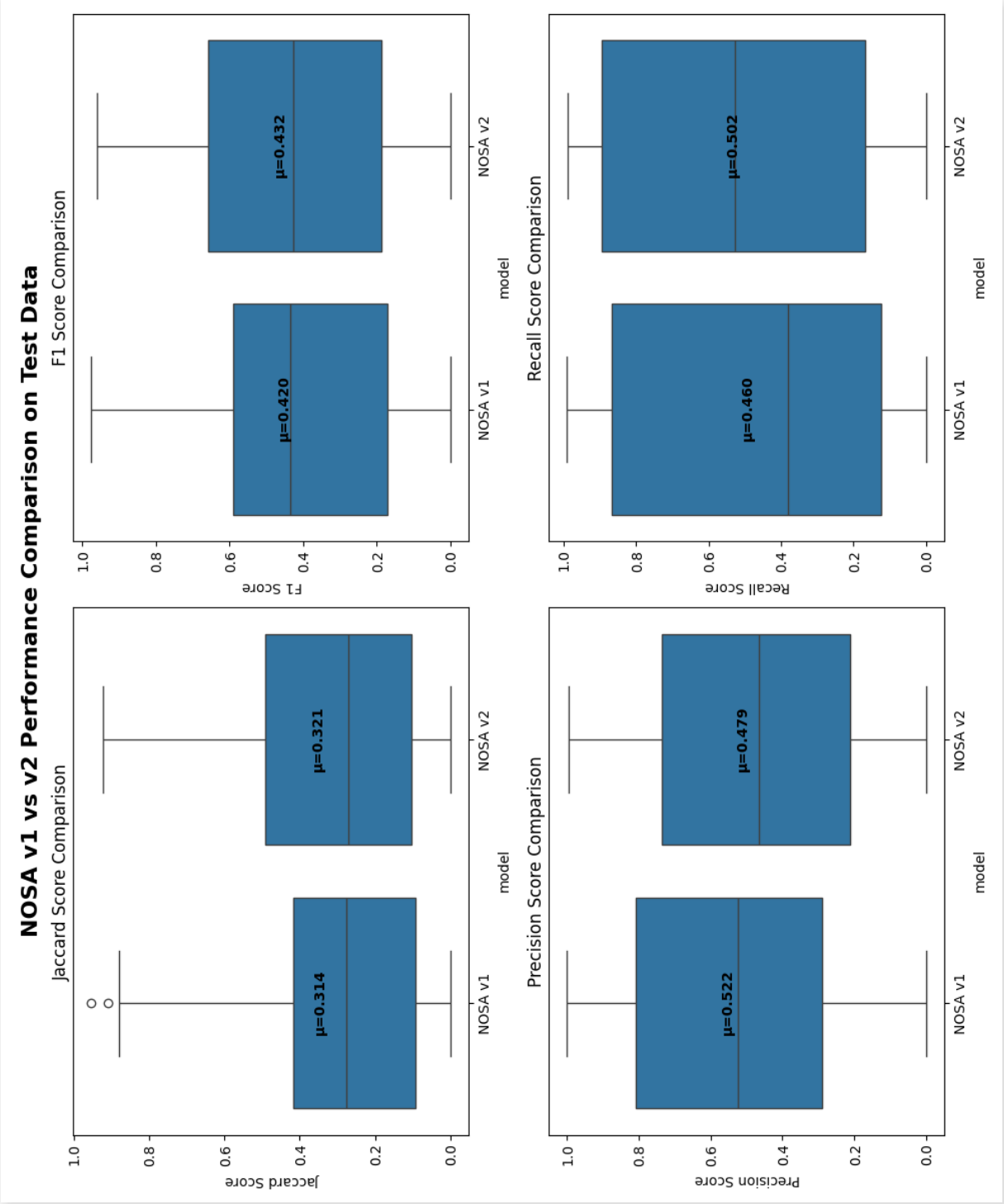


Figure 21: Boxplot comparison of NOSAv1 and NOSAv2 on test data using common metrics. This graph was created using a performance comparison script written by Microsoft's Copilot.

4.3.3 Performance on the Training Dataset

The line graph in Figure 22 and the boxplot in Figure 23 (in the following page) compare the performance of NOSAv1 and NOSAv2 using the images on which they were trained. Collecting performance data using the training dataset was done to investigate if a model would overfit. An indicator of overfitting is if the model performs significantly better on the training data than on the test data. The versions are not compared with each other, but with themselves on test data.

Across all metrics NOSAv1 shows a marginal average increase in the mean performance of 1.27% on the training data compared to the test data.

Across all metrics NOSAv2 shows a larger increase in the mean performance of 4.53% on the training data compared to the test data than NOSAv1, but NOSAv2 performs 4.65% better than NOSAv1 on average.

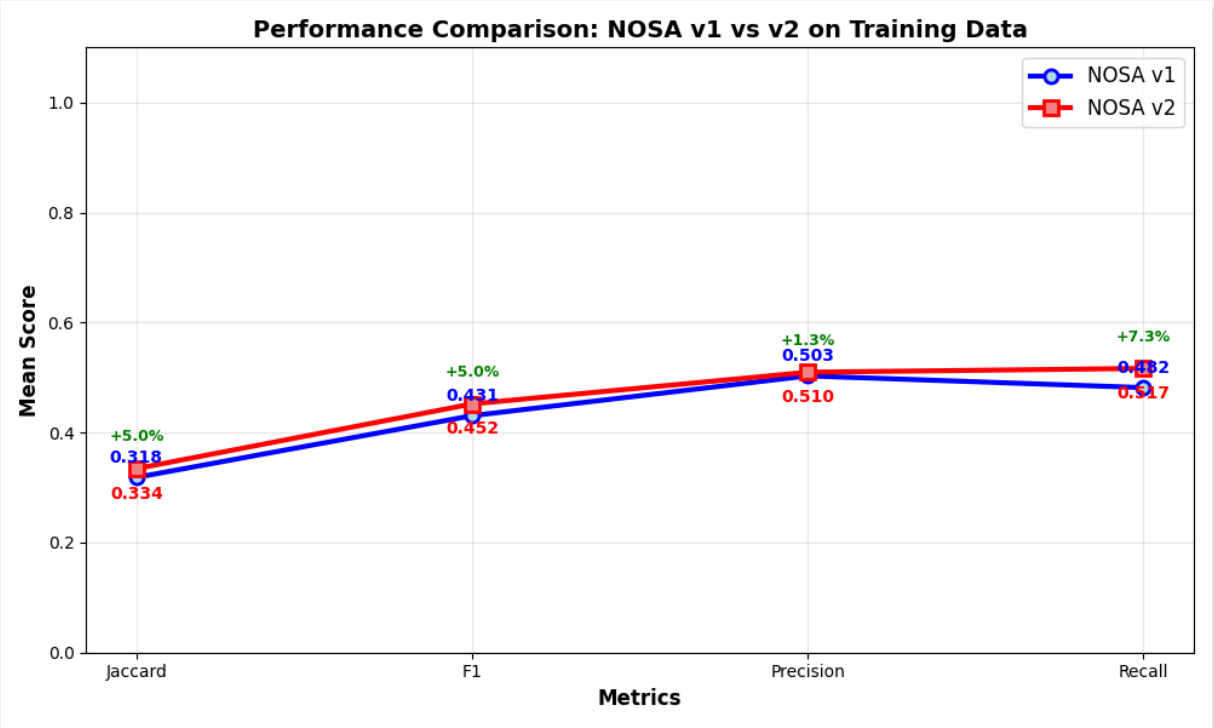


Figure 22: Comparison of NOSAv1's and NOSAv2's mean score in common metrics on training data. This graph was created using a performance comparison script written by Microsoft's Copilot.

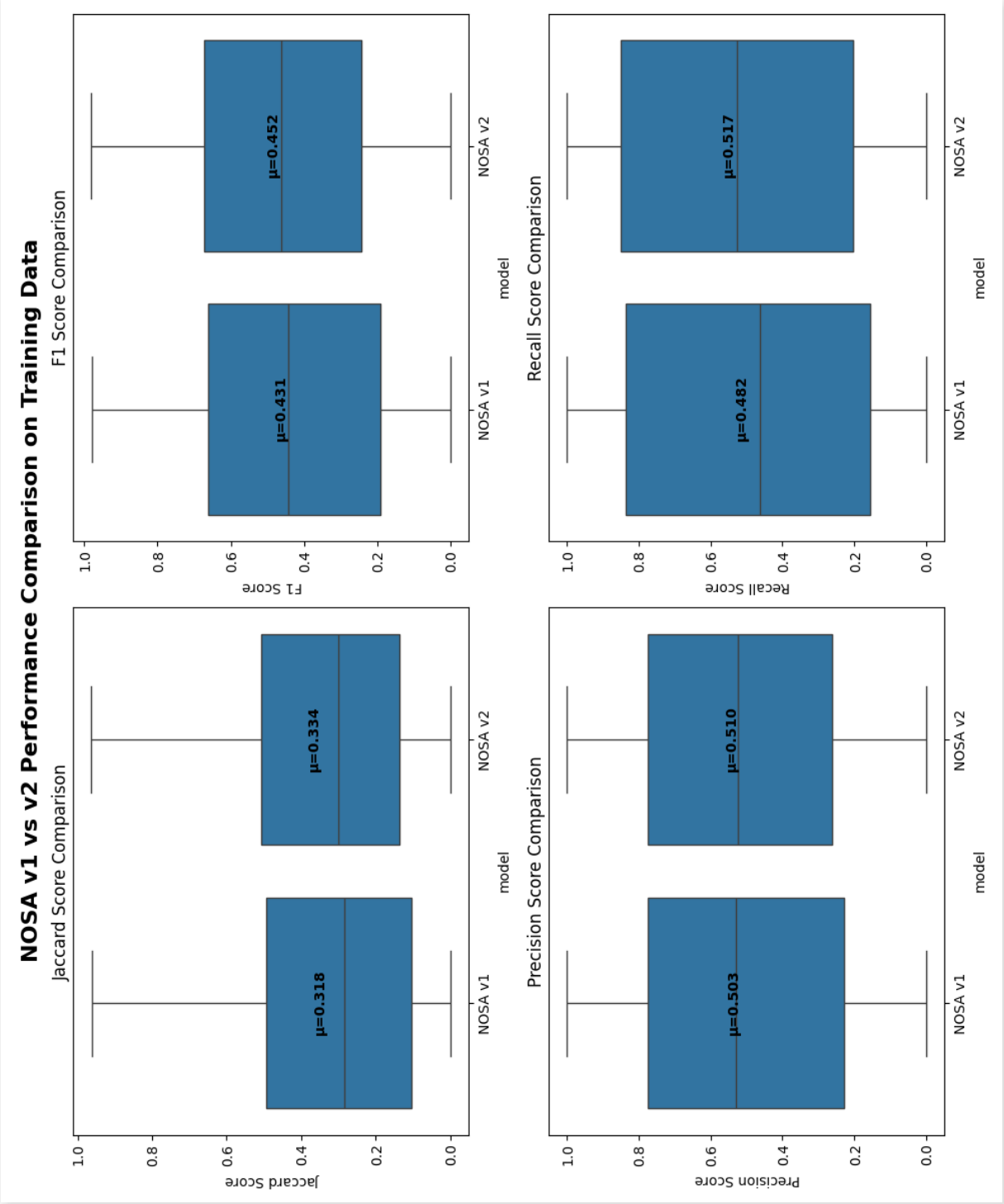


Figure 23: Boxplot comparison of NOSAv1 and NOSAv2 on training data using common metrics. This graph was created using a performance comparison script written by Microsoft's Copilot.

4.3.4 Performance of the Heatmap

The line graph in Figure 24 and the boxplot in Figure 25 (in the following page) compare the performance of NOSAv2 with and without the heatmap. The comparison was done using the test data and a python script written by Microsoft's Copilot. The Jaccard and F1 scores showed substantial decreases (-41.3% and -38.1% change in mean scores, Figure 24) with the heatmap applied. The Precision score, on the other hand, shows a significant improvement (+20.4% change in the median score, Figure 24). Precision shows an incredibly wide interquartile range with the heatmap applied, ranging from zero to near perfect, compared to the stand-alone model. The recall score showed the greatest decrease (-59.5% change in the median score, Figure 25)

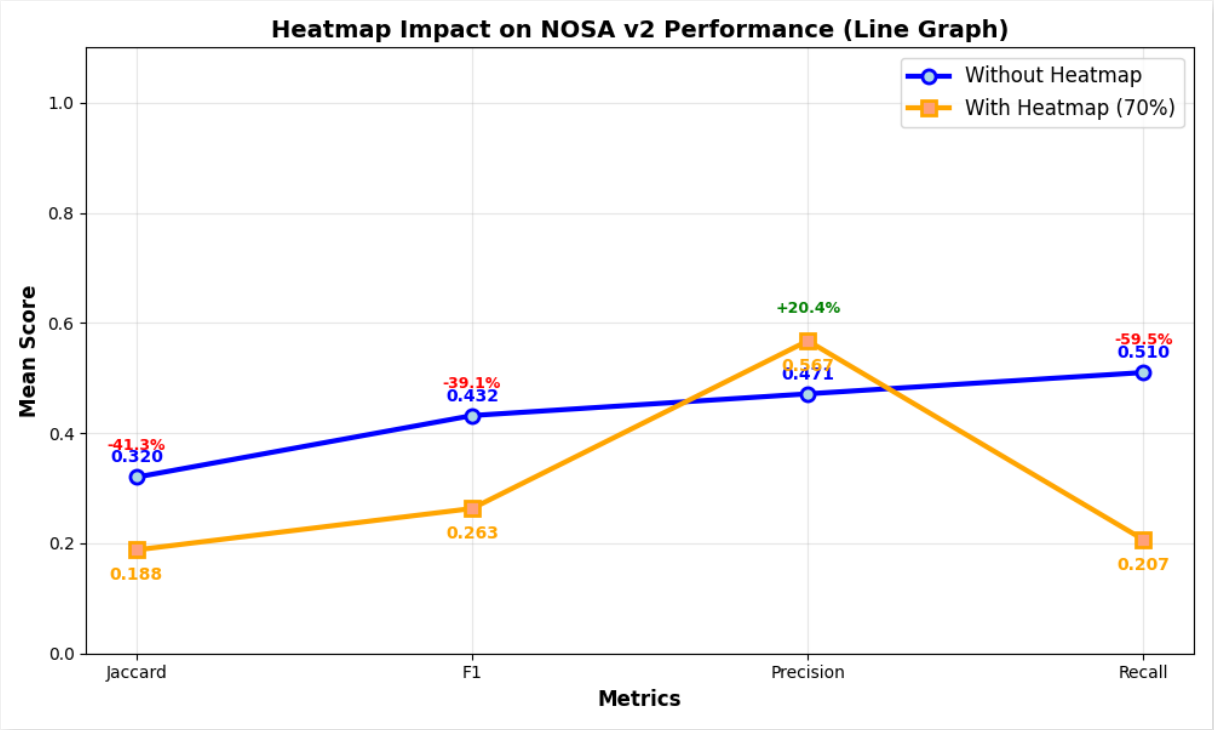


Figure 24: Comparison of NOSAv2's mean score in common metric with and without the heatmap on training data. This graph was created using a performance comparison script written by Microsoft's Copilot.

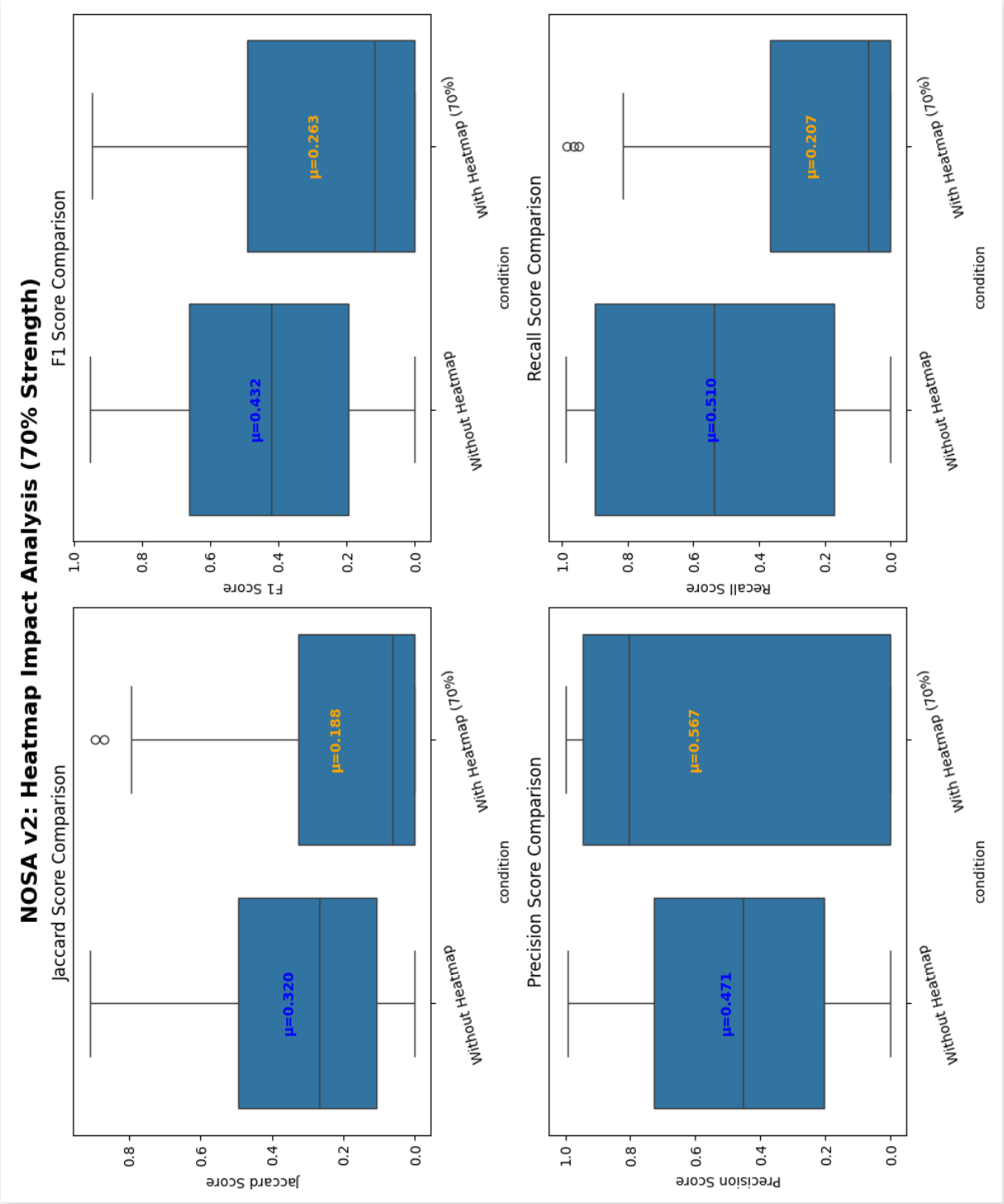


Figure 25: Boxplot comparison of NOSAv2's performance with and without the heatmap on training data using common metrics. This graph was created using a performance comparison script written by Microsoft's Copilot.

4.4 Comparison with State-of-the-Art Models

NOSAv2 shows a competitive accuracy compared to state-of-the-art U-Net models in Figure 26. NOSA even supersedes the baseline model used in the "Automated Brain Tumor MRI Segmentation Using ARU-Net with Residual-Attention Modules" by Erdal and Feyza Özbay [43]. Note that the accuracy metric is not a reliable metric in segmentation with unbalanced classes, such as the segmentation of brain tumors. Accuracy is the percentage of correctly classified pixels, including the background pixels. If a model were to classify all pixels in an image with 1000 pixels as background, and there were 50 pixels that were not background, the model would still achieve an accuracy of 95% . [28]

The NOSAv2 model shows a significant deficit in IoU / Jaccard index score. This means the predicted mask of NOSA overlaps significantly less compared to other models.

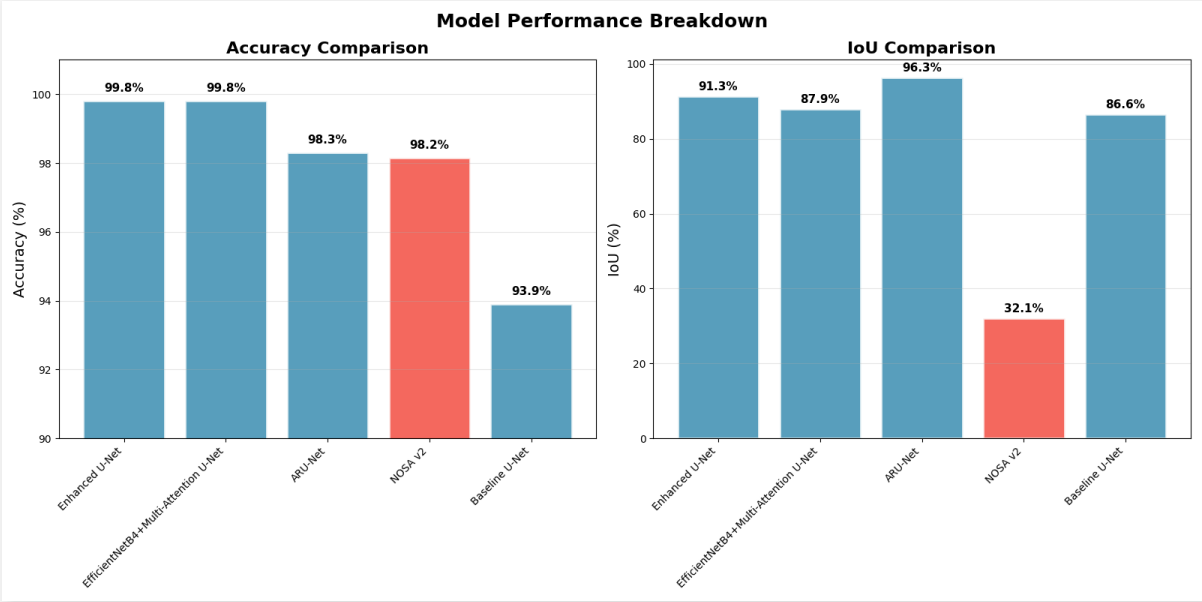


Figure 26: Comparison of NOSAv2's meanIoU score and mean accuracy with state-of-the-art models. This graph was created using a performance comparison script written by Microsoft's Copilot. [43] [44][45]

5. Discussion

5.1 Interpretation of the Results

5.1.1 Training Performance

The graphs in Figure 19 show that NOSAv1 converges earlier (≈ 75), due to its limited model capacity. (32 initial filters, maximum of 256 filters), resulting in a higher loss (0.1101). This suggest that the architecture and training methods used in NOSAv1 produce a model suitable for prototyping in development, as it can be quickly adjusted and retrained, even when computation resources are limited.

NOSAv2 converges 10 layers later (≈ 85), but achieves a slightly better final loss (0.0855). The later convergence is due to its larger capacity (62 initial filters, and 1024 = 4x more Parameters), which requires more epochs to optimize every parameter. This shows that increasing the model parameters results in greater learning potential.

5.1.2 Evaluation of the Visual Results

Because there are no great visible errors, the NOSAv2 works well for finding brain tumors. It shows great performance compared with what the average person is capable of, but to gain clinical relevance and be useful to a radiologist, delineation must be absolutely perfect. This is why metrics are used to assess a model's performance beyond what is visible to the human eye.

5.1.3 Performance on the Test Dataset

As seen in Figure 21, the improvements in NOSAv2 achieve a minimal increase in performance, comparable to the increase in training performance, in the F1 score and the IoU score (Jaccard score). However NOSAv2 shows a common precision-recall trade-off. A significantly higher recall (+9.1%) shows that NOSAv2 is better at finding true positive pixels, but at the cost of significantly lower precision (-8.2%), which means the model also produces more false positive predictions. This shows that NOSAv2 labels more pixels as a tumor, but achieves a better and more detailed result over all.

The wide spread across all metrics shows that neither version achieves consistent stability. Stability is a crucial aspect of a model, as it shows how reliable it is. However, both NOSAv1 and NOSAv2 achieve near perfect results in some images, in all metrics, which shows that the approach shows promising potential if more sophisticated methods are implemented.

The marginal increase in results (1.45%) shows that solely increasing the number of filters (information)

and adding a more sophisticated loss function is not enough to drastically improve the segmentation capabilities of the model.

5.1.4 Performance on the Training Dataset

Most models will show slightly better performance on the training data, as their parameters were optimized to segment these images. However, the amount by which the performance increases is critical.

NOSAv1 shows no signs of overfitting with a marginal increase in performance on testing data (1.27%).

NOSAv2 has an increase of 4.5%, which is significantly larger than NOSAv1, which suggests that the model fits the data more strongly and may overfit slightly. There is no definitive number by which can be decided whether a model overfits or not, this performance comparison would have to be continued with following models to be able to judge it correctly. However, the NOSAv2 shows an increase in performance compared to NOSAv1 of 4.65% on training data, which suggests genuine improvement in model capacity and improvement, rather than overfitting.

5.1.5 Dataset Split

A major issue in the validation of the results is that the size of the test dataset was significantly smaller than that of the training dataset. A common split of 80/20 would require a minimum of 600 images in the test dataset. The small test dataset of 100 images drastically reduces the statistical validity of the model performance. However, while in future research the 80/20 split will be implemented, it can be assumed due to the other data available and the consistency of performance increase in NOSAv1 and NOSAv2 across training and test data that the results are valid.

5.2 Comparison of Results with State-of-the-Art Models

5.2.1 Impact of Training Differences

The results from Figure 26 show that NOSAv2 notably under performs compared to the other models. Except for their remarkably more refined feature extraction methods, a significant difference in epochs (96 in NOSA and 200+ in the others) might suggest that NOSAv2 is not allowed to fully optimize because training is cut short. However, as can be seen in Figure 19, the model converges before training is finished, which means it would not be able to reach a better result with more time.

What might happen during training is that the model settles for a local minima and is not capable of escaping that local minima due to the learning rate being too small. The other models use more advanced scheduling approaches to reach the global minimum, such as cosine restart (Enhanced U-Net).

Additionally, the state-of-the-art models use specific loss functions such as dice loss, BCE loss, and boundary loss, which directly address segmentation edge precision (delineation). This means that the loss function used in training directly teaches the model how to excel at the most important metrics.

The advanced scheduling, more epochs, and boundary-specific loss functions allow the models to better learn important, fine features and with that achieve better performance (90+% compared to 32.1%). This may also be the reason why NOSA significantly under performs compared to the baseline U-Net.

5.2.2 Impact of Architectural Differences

The main difference compared to the architecture of the state-of-the-art models, which massively increases the model efficiency, is the implemented attention modules. The attention models allow the model to focus mainly on tumor-relevant features, which means the model learns finer features of that which is relevant (the tumor) and does not waste its time and resources on less relevant areas (the rest of the image). This resulted in a significantly better performance compared to NOSAv2 which does not have attention modules.

NOSAv2 uses concatenation-based skip connections, which directly merge feature maps across layers to avoid any information during upsampling. The other models use different more advanced skip connections to enhance the data that is transferred through the connections.

The significantly more advanced architecture of the state-of-the-art models allow them to process information more effectively, only focusing on that which is relevant. The relevant area is very small, which is why a model like NOSAv2 will reach a competitive accuracy (98.2%). However, the model will not be able to learn the finest features of a tumor which are required to accurately delineate it. This explains the large gap in the IoU comparison of Figure 26.

5.2.3 Impact of Dataset Differences

The datasets on which the state-of-the-art models were trained were significantly larger and more diverse, which produces a more reliable model that can accurately segment a tumor in a wide variety of images, including imperfect images.

Another notable difference is that the state-of-the-art models were trained using 3-dimensional images. This allows for richer spatial information and context to be processed compared to only 2-dimensional outlines. A 3-dimensional representation can portray the complex structure of a tumor more accurately, which allows the model to learn to better distinguish between healthy tissue and a tumor. The model can use the 3-dimensional scans to process the volumetric data of differently weighted MRI images simultaneously, which allows the model to learn which features correspond to which across the differently weighted MRI images.

5.3 Practical Relevance

5.3.1 NOSA

Unfortunately, neither the model in NOSAv1 nor its successor NOSAv2 achieve a standard close to what is needed to be clinically relevant for delineation. However, once a model is developed that achieves the clinical standard, it can easily be implemented into NOSA, as mentioned in section 3.6.3.

5.3.2 AI-driven Tools in Radiology

The state-of-the-art models, on the other hand, reach the standard of clinical relevance and are used to segment brain tumors, which reduces manual effort. The models can be used to gain quantitative data to track the progress of tumors, which supports better planning and monitoring of treatments. Additionally, the precise and quick delineation of a tumor may enable a very early diagnosis and treatment can begin before the tumor progresses to the late stages of cancer. A more in-depth examination of this topic is done in the interview with Dr. Diepers in section 5.5.

While it is my personal belief that AI-driven tools should never work unsupervised in diagnosis, I find the implementation of purely AI-driven first screenings a feasible, cost-effective alternative, especially in countries where free healthcare is not provided. A cheap first opinion can then aid the decision whether a real hospital should be visited or not.

5.4 Technical limitations and challenges

The primary limitations that prevented the development of a more advanced model, perhaps even comparable to state-of-the-art approaches, were the limited computational resources and a small project time frame.

Computational Constraints

The computer that was used to train and develop the models had some major issues such as overheating when computing challenging tasks such as model training over a long period of time. Another concern was that the computer could not be occupied with training for long periods, as it had to be used daily for other tasks. Upgrading the computer was not an option due to budget issues.

5.4.1 Time Frame Constraints and Limited Experience

The Matura's limited time frame of 6 months did not allow time for the development of a better version. This is because the required knowledge to build an advanced model had to be acquired before development could begin, which left limited time to develop, train and optimize a model. It was decided not to start the development of an advanced model due to the risk of not finishing the development within the given time frame. Instead, the time was used to develop a simple but fully functional model, with the addition of a user friendly GUI.

5.5 Interview with a Neuroradiologist on AI-driven Diagnostic Tools

To gain a professional opinion on the use of AI-driven tools, an interview was conducted with Dr. med. Diepers, Deputy Head of Neuroradiology, and Senior Physician in Neuroradiology at the KSA. The interview was conducted over a recorded phone call in German. Unless otherwise cited, all information in section 5.5 is derived from this interview, and most content is directly paraphrased. The information provided reflects his personal perspectives and experiences. As such, the information is subjective and should be interpreted accordingly. The complete transcript can be found in Appendix F

5.5.1 Current Use of AI-driven Tools

The use of AI-driven tools is increasing for different tasks, mostly for analyzing, detecting, and classifying pathologies to assist diagnosis. AI is used in MRI to analyze tissue change and help pathologists distinguish between healthy tissue that appears abnormal and truly dangerous abnormalities. AI is also used to visualize data such as functional data in a way that is not possible with traditional methods or to create topographical maps of the brains perfusion, highlighting deficits or surpluses.

AI-tools can also be used for organization tasks, they can help improve the efficiency of a doctor's work and optimize schedules, and equipment usage to meet demand peaks

5.5.2 Impact of AI-Driven Tools on Radiologists

According to Dr. Diepers, the integration of AI-driven tools already has a significant impact on radiology by increasing throughput and efficiency. The AI tools allow radiologists to accurately analyze and process a much larger volume of data in the same time frame. This increased efficiency is a welcomed gain.

However, the increased efficiency comes at the cost of more pressure. From an outside perspective, the argument is that radiologists can use AI now, they can work faster now. It is expected of a radiologist to deliver more in less time while the quality should remain consistently high. While radiologists can work faster using AI, the expectations rise proportionally.

5.5.3 Limitations of AI-driven Tools

Despite these advances, Dr. Diepers highlights critical limitations. While AI is extremely efficient and accurate at processing information, the communication with patients cannot be replaced by AI. A doctor can empathize with patients and notice how a patient is feeling without having to be told. A doctor may even notice things the patient is trying to hide. Noticing these things is incredibly important to determine what a patient lacks.

Also, the doctor-patient relationship is not only about finding out what is wrong with the patient and then fixing it. Often the doctor must proceed carefully, considering factors such as how the patient feels and their personality. As an example where the patient should not be confronted with everything at once, Dr. Diepers uses the following:

> "For example, if a patient has a tumor but hasn't noticed any symptoms yet, and the tumor is discovered incidentally during an examination for a completely different complaint, it is not always appropriate to bluntly tell the patient about it. The information must be carefully introduced and tailored to the patient's personality when explaining the facts. This does not mean withholding information, but understanding how to communicate it, whether provocatively or gently, and choosing the right words."

In the case of radiology, it is not enough to just let an AI process all data and draw conclusions. A radiologist always has to supervise the AI and judge whether the AI's decisions were correct. This can

be difficult to do because of the blackbox problem (it can be nearly impossible to comprehend the decisions made by an AI). AI may be more precise and thorough than humans, but placing what is seen in the context of patient care is something a human must do.

In summary, the empathy, experience, and knowledge a doctor has means that in our current time, even the most advanced AI cannot replace the interpersonal aspects of being a doctor.

5.5.4 Future Use of AI-driven Tools

There is an increasing rise in popularity of AI-tools which offer a statistical prognosis, as an example MIRAI[1] AI was chosen, an AI-tool that offers a forecast of a patients year-by-year risk of developing breast cancer over the next five years. [46]

Dr. Diepers' personal opinion is that predictive AI tools are dangerously misleading as they give a false impression that conditions such as cancer are predictable solely on a statistical evaluation of image data. These models cannot consider the countless additional factors that determine whether an inflammatory focus degenerates into a malignancy.

A prognosis by such an AI is only a rough estimate, but it is very convenient for doctors to be able to say: "There is a 15% chance that cancer will develop in the next five years.". Gaining a comprehensive understanding of what your future might look like is very tempting knowledge, which is why the demand of such models will only increase, even if it is a pseudo accuracy.

While Dr. Diepers finds it impossible for AI to completely replace radiologists, he does see a possibility that the field of radiology shifts to where a radiologist primarily organizes incoming data and supervises that the machines function properly.

[1] MIRAI is not an acronym, the name was chosen from the Japanese word mirai, which means future. [46]

5.5.5 Ethics of using AI-driven tools

A major problem in the use of AI-driven tools is the privacy of patient data. Currently, there is no definitive solution to this problem. Common approaches for solving this problem include to anonymize data before feeding it to the AI-tools or separating personal data in a way that cannot be accessed externally. However, based on his experience, Dr. Diepers can say that these solutions do not always work.

An example of this is that the decisions of the AI, the information they provide, are not restricted. This means that any doctor with access to the patient's imaging files can view the AI's output, not just the doctor that used to AI to aid their decision-making. This lack of control can create problems if the AI's recommendation conflicts with the decision made by the doctor, other doctors viewing the report may incorrectly assume the doctor missed something, and send the patient for another screening. Not being able to restrict the AI's output can waste resources and time.

Another problem is who takes responsibility for the decision made by the AI. There is no general solution for this, the decision who takes responsibility is currently made on a case-by-case basis. Predominately, the user takes responsibility, as they have to supervise the decisions made by the AI.

6. Reflection

Work Process

The process began with extensive research, and with limited prior knowledge, almost everything had to be learned. This took almost three months out of the available six, leaving only two months for development and one month for writing. This allocation of time proved to be a good choice, as Abraham Lincoln once said:

> "Give me six hours to chop down a tree and I'll spend the first four sharpening the axe."
>
> — Abraham Lincoln

Once enough knowledge was gained, development could begin. The early stages were spent carefully planning methodologies, including dataset collection, preprocessing, and model choice. Throughout the project, there were various challenges, such as acquiring a labeled dataset. This issue was resolved in a week so no significant time was lost. Another challenge was the limited computational power, which drastically restricted the quality of the product. The implementation of a GUI was not necessary, yet the remaining time did not allow for a new version to be developed, so it was deemed a valid use of the remaining time.

The analysis using an AI-developed script was conducted because investing weeks in developing a custom script seemed unnecessary, since only the data gained from the script is relevant to the thesis.

Overall, the process has significantly deepened my understanding of AI in general but also specific to brain tumor segmentation.

Dataset Split

A major issue in this project was the incorrect split of training data and test data, which led to an unreliable depiction of the model performance on the testing data. Unfortunately, there was not enough time to retrain the models on a better split, since besides training time, all diagnostics, comparisons, and interpretation of results would have to be redone. Additionally, in future research, the patient ID can be used to ensure an even spread of cases and tumor types in both datasets.

Adaptation of Advanced U-Net Architecture

Adapting an advanced version of the U-Net was heavily considered, but it was decided that the risk of not completing development, training, and evaluation within the given time frame was too great. A very basic approach was chosen, with minor enhancements made to NOSAv2. Although technical knowledge to implement advanced features such as attention mechanisms was present due to the extensive research done, the complexity of integrating such features to work properly was not possible in the given time frame. The choice of using a simple U-Net structure was the right choice for a project on the scale of a Matura thesis.

Technical Skills developed: PyTorch and MATLAB

Learning to use PyTorch was essential to implement and train the NOSA models, however it posed a great challenge. Understanding the structure of a DL model is simple in theory, but applying that knowledge in the form of code is significantly more complex. Another challenge was coding in a manner that did not demand too much computing power.

Learning to use MATLAB was a small challenge compared to PyTorch, but learning to properly process and convert data from MATLAB files allowed for the use of a valuable dataset.

The challenges were rewarding as the gained skills can be used in future research to develop more complex architectures, as no time will have to be spent on building foundational knowledge.

Writing with LaTeX

Using LaTeX to write the thesis was challenging and time consuming. Learning to understand document structure, commands, and bibliography management required a great effort, which people using writing programs like Word did not have. Additionally, grammar is not corrected in LaTeX beyond the syntax of words. This had to be done manually.

However, the gained control and professionalism were worth the invested time. The experience working with LaTeX improved my technical writing skills and provided a powerful tool set that I plan on using in future scholarly work.

Workload and Effort

The workload associated with this thesis was substantially larger than that required. Careful time management to balance school, personal commitments, and the project was necessary; otherwise, I could have not have finished the project. The early stages of development and research took significantly longer than expected due to the complexity of the project, but sufficient motivation and a disciplined work ethic allowed me to reach a good result.

Weekly deadlines were a great asset, as they allowed me to stick to my tight schedule. If a problem arose, I gave myself a week to find possible solutions. If solutions were found, they were implemented.

If no solutions were found within a week, a heuristic approach was used (such as the heatmap), and I moved on. Balancing ambitions and achievable results was another key to a good result and motivation. This experience taught me valuable lessons about perseverance, self-discipline, and the realistic planning of large-scale projects.

7. Conclusion

The initial goals of reviewing the methods of existing models, analyzing feasibility of AI-driven diagnostic tools, and adapting an existing U-Net structure to segment brain tumors have been reached. However, the proposed model lacks specific components that allow it to segment tumors as accurately as state-of-the-art models. I believe that given the time frame, the limited computational resources, having little prior experience, and the fact that the model shows great performance in classification, a satisfactory result has been reached.

I believe that in future research it would be possible to develop a model capable of competing with the state-of-the-art models; in fact, it is the plan to compete in BraTS 2026.

While the philosophical and ethical questions behind the use of AI may never be fully resolved, I believe that as long as humans regard AI as a tool to use and not a replacement for people, AI will allow for wonderful things. The journey of AI in medicine has just begun, and its legacy will be written saving lives.

"The future of AI is not about replacing humans, it's about augmenting human capabilities."

— Sundar Pichai, CEO of Google

Thank you to Martin and Didi for proofreading this thesis. Thank you to Jacob for introducing me to Dr. Diepers and making the interview possible. Thank you to Dr. Diepers for the very informative interview. Thank you to Ms. Vàzquez for mentoring this thesis.

A. List of Abbreviations

A

Adam adaptive moment estimation. , 10, 23, XIV, XVIII, *Glossary:* adaptive moment estimation

AI artificial intelligence. , 6, 7, 14, 45–49, 52, *Glossary:* artificial intelligence

ARU-Net Advanced Residual U-Net. 26–28, 41, *Glossary:* Advanced Residual U-Net

B

BCE binary cross entropy. , 10, 22, 43, XIV, XVI, *Glossary:* binary cross entropy

BraTS Brain Tumor Segmentation Challenge. 26–28, 52, *Glossary:* Brain Tumor Segmentation Challenge

C

CAM channel attention module. 27, 28, *Glossary:* channel attention module

CNN convolutional neural network. 1, 11, 13, 27, *Glossary:* convolutional neural network

CPU central processing unit. 25, *Glossary:* central processing unit

CSF cerebrospinal fluid. 3, 4, *Glossary:* cerebrospinal fluid

D

DL deep learning. 6–8, 14, 16, 17, 20, 22, 27, 50, *Glossary:* deep learning

F

FCN fully convolutional network. 11, 13, 22, *Glossary:* fully convolutional network

FLAIR fluid attenuated inversion recovery. 4, 17, 18, 26, 28, *Glossary:* fluid attenuated inversion recovery

G

GBM glioblastoma multiforme. 5, 27, *Glossary:* glioblastoma multiforme

GPU graphics processing unit. 23, 25, 26, *Glossary:* graphics processing unit

GUI graphical user interface. , 24–26, 32, 45, 49, *Glossary:* graphical user interface

H

HD-BET High-Definition Brain Extraction Tool. 20, *Glossary:* High-Definition Brain Extraction Tool

I

IoU intersection over union. , 15, 41, 42, 44, XIV, *Glossary:* intersection over union

K

KSA Kantonsspital Aarau. 6, 45, *Glossary:* Kantonsspital Aarau

L

LGG lower grade gliomas. 27, *Glossary:* lower grade gliomas

M

ML machine learning. 7, 8, 17, *Glossary:* machine learning

MRI magnetic resonance imaging. , 1–4, 6, 11, 14, 18, 20, 24, 26–28, 41, 44, 46, *Glossary:* magnetic resonance imaging

N

neural network neural network. 8, 10, 11, *Glossary:* neural network

NOSA Neuro Oncology Segmentation Algorithm. , 21, 22, 25, 26, 28–44, 50, XIII, XIV, *Glossary:* Neuro Oncology Segmentation Algorithm

NumPy NumPy. 26, *Glossary:* NumPy

P

PAM positional attention model. 28, *Glossary:* positional attention model

PyTorch PyTorch. 21, 22, 26, 50, *Glossary:* PyTorch

R

ReLU rectified linear unit. 11, 22, *Glossary:* rectified linear unit

RF radio frequency. 2–4, *Glossary:* radio frequency

S

SAM spatial attention module. 27, 28, *Glossary:* spatial attention module

T

TCGA The Cancer Genome Atlas. 27, *Glossary:* The Cancer Genome Atlas

TCIA The Cancer Imaging Archive. 17, *Glossary:* The Cancer Imaging Archive

TE time to echo. 4, *Glossary:* time to echo

TR repetition time. 3, 4, *Glossary:* repetition time

U

U-Net U-Net. , 1, 13, 19–21, 25–28, 41, 43, 44, 50, 52, *Glossary:* U-Net

W

WHO World Health Organization. *Glossary:* World Health Organization

Y

YOLO You Only Look Once. 20, 21, *Glossary:* You Only Look Once

B. Glossary

A

Adaptive moment estimation An optimization algorithm combining momentum and adaptive learning rates to improve neural network training. [20]. , I

Advanced Residual U-Net A U-Net variant incorporating residual connections for improved segmentation accuracy in medical imaging. [43]. 26, I

Artificial intelligence A branch of computer science focused on creating systems capable of performing tasks that normally require human intelligence. [18]. , I

B

Binary cross entropy A loss function measuring the difference between predicted probabilities and actual binary class labels. [19]. , I

Brain Tumor Segmentation Challenge An international competition providing standardized MRI datasets for benchmarking brain tumor segmentation algorithms. [47]. 26, I

C

Central processing unit The component responsible for all major processing tasks in a computer.. 25, I

Cerebrospinal fluid A clear liquid surrounding the brain and spinal cord, providing protection and metabolic waste removal. [4]. 3, I

Channel attention module Mechanisms in neural networks that emphasize important feature channels to enhance representation learning. [44]. 27, I

Convolutional neural network A type of artificial neural network used primarily for image recognition and processing, due to its ability to recognize patterns in images. [48]. 1, I

D

Deep learning A subfield of AI that uses neural networks with multiple layers to learn representations from large datasets. [18]. 6, I

Dice A metric used to measure the similarity between two sets of data. [28]. , 10, 15, 22, 23, 26, 43, XIV, XVI

F

Fluid attenuated inversion recovery An MRI sequence that suppresses cerebrospinal fluid signals to highlight lesions in brain tissue. [4]. 4, I

Fully convolutional network Neural networks that apply convolutional operations throughout all layers, enabling pixel-wise prediction tasks such as image segmentation. [15]. 11, I

G

Glioblastoma multiforme An aggressive and highly malignant form of brain tumor arising from glial cells. [11]. 5, I

Graphical user interface Visual interfaces that allow users to interact with software through graphical elements rather than text commands. [49]. , II

Graphics processing unit Specialized processors designed for parallel computations of images and videos.. 23, II

H

High-Definition Brain Extraction Tool A deep learning-based method for precise brain extraction from MRI scans. [33]. 20, II

I

Inference engine An inference engine is a core component of expert systems and rule-based systems. It is designed to simulate human reasoning by applying logical rules to a set of facts or data within a knowledge base. It serves as the decision-making mechanism by analyzing input data and deriving conclusions or recommendations based on predefined rules. [50]. 25, 31, 32

Intersection over union A metric used to evaluate the accuracy of object detection or segmentation models by comparing predicted and ground truth areas. [28]. , II

K

Kantonsspital Aarau A regional hospital in Aarau, Switzerland.. 6, II

L

Lower grade gliomas A group of slower-growing primary brain tumors that may progress to higher-grade malignancies. [6]. 27, II

M

Machine learning A subset of AI involving algorithms that learn patterns from data to make predictions or decisions without explicit programming. [18]. 7, II

Magnetic resonance imaging A non-invasive imaging technology that produces three dimensional detailed anatomical images. It is based on sophisticated technology that excites and detects the change in the direction of the rotational axis of protons found in the water that makes up living tissues. [51]. , II

Mat73 A Python library used to process MATLAB files. [41]. 26

N

Neural network A structure used in Machine Learning which mimics the human brain. Neural networks learn patterns to recognize patterns from data. [18]. 8, 10, 11, II

Neuro Oncology Segmentation Algorithm A modified U-Net for automated brain tumor segmentation, developed for neuro-oncology applications.. , II

NumPy A Python library used to handle large, multi-dimensional arrays and performing fast computations efficiently. [42]. 26, II

P

Positional attention model Attention mechanisms that capture long-range dependencies by modeling positional relationships in feature maps. [44]. 28, II

PyQt5 A python library used to build engaging GUIs. [39]. 26

PyTorch A powerful python library essential to building, training, and developing deep learning models. [40]. 21, 22, 26, 50, II

R

Radio frequency An electromagnetic wave frequency used in MRI to excite atomic nuclei and produce measurable signals. [5]. 2, II

Rectified linear unit An activation function that introduces non-linearity by outputting the input if positive, and zero otherwise. [37]. 11, II

Repetition time The time interval between successive pulse sequences in MRI, affecting image contrast and signal intensity. [5]. 3, III

S

Spatial attention module Components that help neural networks focus on relevant spatial regions in feature maps. [44]. 27, III

T

Tensor Tensors in computer science are multidimensional arrays that store a specific type of value. [52]. 19, 25

The Cancer Genome Atlas A project offering genomic datasets for multiple cancer types, including gliomas, to advance molecular research.. 27, III

The Cancer Imaging Archive A large repository of publicly available medical images supporting cancer research and algorithm validation.. 17, III

Time to echo The time between the application of the RF excitation pulse and the peak of the echo signal in MRI, influencing image contrast. [5]. 4, III

U

U-Net A convolutional neural network designed for precise image segmentation. It has a U-shaped architecture, hence its name. [15]. , 1, 13, 19–21, 25–28, 41, 43, 44, 50, 52, III

W

World Health Organization A specialized agency of the United Nations responsible for international public health.. III

Y

You Only Look Once A real-time object detection algorithm that predicts bounding boxes and classifications in a single network pass. [34]. 20, III

C. Sources

C.1 Bibliography

1. MENZE, Bjoern H.; JAKAB, Andras; BAUER, Stefan; KALPATHY-CRAMER, Jayashree; FARAHANI, Keyvan; KIRBY, Justin; BURREN, Steve; PORZ, Niko; SLOTBOOM, Johannes; WIEST, Roland; LANCZI, Lajos; KOVACS, Gyorgy; GLOCKER, Ben; FICHTINGER, Gabor; HOLMES, Simon; DEEPAK, Bhanu Prakash; SUMMERS, Ronald M.; PAMIR, Mehmet N.; BIRELLO, Marta; FEMIA, Paolo; CRIMI, Alessandro; SHBOUL, Zaid; SAID, Nabil; REIS, Guilherme; PRASTAWA, Marcel; THOMAS, Omar; WARD, Chad; TIRSCHWELL, David L.; AZAD, Tara D.; SAHA, Apresh; KIKINIS, Ron; MEIER, Reinhard; REYES, Mauricio; VAN LEEMPUT, Koen. The Multimodal Brain Tumor Image Segmentation Benchmark (BraTS). *Scientific Reports* [Online]. 2013, vol. 3, pp. 1–6. Available from DOI: 10.1038/srep01364.
2. HOLLON, Todd C.; PANDIAN, Balaji; ADAPA, Arjun R.; URIAS, Esteban; SAVE, Akshay V.; KHALSA, Siri Sahib S.; EICHBERG, Daniel G.; D'AMICO, Randy S.; FAROOQ, Zia U.; LEWIS, Spencer; PETRIDIS, Petros D.; MARIE, Tamara; SHAH, Ashish H.; GARTON, Hugh J. L.; MAHER, Cormac O.; HETH, Jason A.; MCKEAN, Erin L.; SULLIVAN, Stephen E.; HERVEY-JUMPER, Shawn L.; PATIL, Parag G.; THOMPSON, B. Gregory; SAGHER, Oren; MCKHANN, Guy M.; KOMOTAR, Ricardo J.; IVAN, Michael E.; SNUDERL, Matija; OTTEN, Marc L.; JOHNSON, Timothy D.; SISTI, Michael B.; BRUCE, Jeffrey N.; MURASZKO, Karin M.; TRAUTMAN, Jay; FREUDIGER, Christian W.; CANOLL, Peter; LEE, Honglak; CAMELO-PIRAGUA, Sandra; ORRINGER, Daniel A. Near real-time intraoperative brain tumor diagnosis using stimulated Raman histology and deep neural networks. *Nature Medicine* [Online]. 2020, vol. 26, no. 1, pp. 52–58. Available from DOI: 10.1038/s41591-019-0715-9.
3. VEIGA-CANUTO, D.; CERDÀ-ALBERICH, L.; NEBOT, C. S.; DE LAS HERAS, B. M.; PÖTSCHGER, U.; GABELLONI, M.; SIERRA, J. M. C.; TASCHNER-MANDL, S.; DÜSTER, V.; CAÑETE, A.; LADENSTEIN, R.; NERI, E.; MARTÍ-BONMATÍ, L. Comparative multicentric evaluation of Inter-Observer variability in manual and automatic segmentation of neuroblastic tumors in magnetic resonance images. *Cancers* [Online]. 2022, vol. 14, no. 15. Available from DOI: 10.3390/cancers14153648.
4. PRESTON, David C. *Magnetic Resonance Imaging (MRI) Basics* [Online]. 2016. Available also from: https://case.edu/med/neurology/NR/MRI%20Basics.htm. Accessed: 18-09-2025.

5. PAI, Aparna; SHETTY, Rohil; HODIS, Brendan; CHOWDHURY, Yuvraj S. *StatPearls*. Magnetic Resonance Imaging Physics [Online]. StatPearls Publishing, 2023. Available also from: https://www.ncbi.nlm.nih.gov/books/NBK564320/. Accessed: 18-09-2025.
6. LEWINE, Howard E. *Brain Tumor Overview* [Online]. 2023. Available also from: https://www.health.harvard.edu/a_to_z/brain-tumor-overview-a-to-z. Reviewed by Howard E. LeWine, MD. Accessed: 20-09-2025.
7. ILIC, Irena; ILIC, Milena. International patterns and trends in the brain cancer incidence and mortality: An observational study based on the global burden of disease. *Heliyon* [Online]. 2023, vol. 9, no. 7. Available from DOI: 10.1016/j.heliyon.2023.e18222.
8. PHOENIX CYBERKNIFE RADIATION & ONCOLOGY CENTER. *How does our benign tumor treatment differ from malignant tumor treatment?* [Online]. 2024. Available also from: https://phoenixcyberknifecenter.com/how-does-our-benign-tumor-treatment-differ-from-malignant-tumor-treatment/. Accessed: 20-09-2025.
9. NATIONAL CANCER INSTITUTE. *Cancer Staging* [Online]. 2022. Available also from: https://www.cancer.gov/about-cancer/diagnosis-staging/staging. Accessed: 12-10-2025.
10. CLEVELAND CLINIC. *Brain metastases* [Online]. Cleveland Clinic, 2025. Available also from: https://my.clevelandclinic.org/health/diseases/17225-metastatic-brain-tumors. Accessed: 27-09-2025.
11. AMERICAN ASSOCIATION OF NEUROLOGICAL SURGEONS. *Glioblastoma multiforme - AANS* [Online]. 2024. Available also from: https://www.aans.org/patients/conditions-treatments/glioblastoma-multiforme/. Accessed: 20-09-2025.
12. CANCER THERAPY ADVISOR. *Benign meningioma: Can you die from it? - Cancer Therapy Advisor* [Online]. Cancer Therapy Advisor, 2025. Available also from: https://www.cancertherapyadvisor.com/factsheets/benign-meningioma-can-you-die-from-it/. Accessed: 27-09-2025.
13. BIBI, Khadija; NAWAZ, Mehmood; KHAN, Sheheryar; DAUD, Muhammad; MASOOD, Anum; ABDELGAWAD, Muhammad Ashraf; ABBASI, Syed Muhammad Tariq; RIZWAN, Muhammad; KHAN, Ahsan; YUAN, Wu. Artificial Intelligence–Based Approaches for Brain Tumor Segmentation in MRI: A Review. *NMR in Biomedicine* [Online]. 2025, vol. 38, no. 11. Available from DOI: 10.1002/nbm.70141. Accessed: 27-09-2025.
14. DIEPERS, Michael. *Interview About the Use and Ethics of AI in Radiology* [Interviewed by Mikkel Lüscher in German]. Phone call, 2025. Unpublished interview, complete transcript in Appendix F.
15. YAO, Wenjian; BAI, Jiajun; LIAO, Wei; CHEN, Yuheng; LIU, Mengjuan; XIE, Yao. From CNN to Transformer: A review of Medical Image Segmentation models. *Journal of Digital Imaging* [Online]. 2024, vol. 37, pp. 1529–1547. Available from DOI: 10.1007/s10278-024-00981-7.
16. DATAIKU. *A Deeper Understanding of Deep Learning | Dataiku* [Online]. 2025. Available also from: https://www.dataiku.com/stories/detail/a-deeper-understanding-of-deep-learning/?utm_campaign=GLO%20CONTENT%20Golden%20Pages%20FY26&utm_source=emea-adwords&utm_medium=paid-search&gad_source=1&gad_campaignid=22259713347&gbraid=0AAAAAC1WRxmRZWEiAEEvM1bKxYkxJPZrC&

gclid = CjwKCAjw_ - 3GBhAYEiwAjh9fUFCe3C4O2CN0HZn2gS2w23jiHGXuzn - zjq2KVzzatENTT1ndBfw2WxoCYtQQAvD_BwE. Accessed: 25-07-2025.

17. CHOLLET, François. *Deep Learning with Python*. Chapter 1. What is deep learning? [Online]. Manning Publications, 2019. Available also from: https://livebook.manning.com/book/deep-learning-with-python/chapter-1#15. Accessed: 30-09-2025.
18. LEE, Fangfang. *What Is a Neural Network?* [Online]. 2025. Available also from: https://www.ibm.com/think/topics/neural-networks. Accessed: 27-09-2025.
19. CORALOGIX. *A Practical Guide to Binary Cross-Entropy and Log Loss* [Online]. Coralogix, 2025. Available also from: https://coralogix.com/ai-blog/understanding-binary-cross-entropy-and-log-loss-for-effective-model-monitoring/. Accessed: 13-10-2025.
20. GEEKSFORGEEKS. *What is Adam Optimizer?* [Online]. GeeksforGeeks, 2025. Available also from: https://www.geeksforgeeks.org/deep-learning/adam-optimizer/. Accessed: 07-10-2025.
21. DATABRICKS. *What is a Convolutional Layer?* [Online]. 2023. Available also from: https://www.databricks.com/glossary/convolutional-layer. Accessed: 30-09-2025.
22. NANDI, Sushmita. *Understanding Convolutional Neural Networks (CNNs) for Beginners* [Online]. 2025. Available also from: https://medium.com/@sushmita2310/understanding-convolutional-neural-networks-cnns-for-beginners-e85ad21fe432. Accessed: 01-10-2025.
23. GAIKWAD, Mohit. *Overview: Fully Convolutional Network for Semantic Segmentation* [Online]. 2022. Available also from: https://medium.com/@mohit_gaikwad/overview-fully-convolutional-network-for-semantic-segmentation-b4ef92eeb8c4. Online. Accessed: 01-10-2025.
24. MERRIAM–WEBSTER. *Concatenate* [Online]. 2025. Available also from: https://www.merriam-webster.com/dictionary/concatenate. Accessed: 01-10-2025.
25. RONNEBERGER, Olaf; FISCHER, Philipp; BROX, Thomas. *U-Net: Convolutional Networks for Biomedical Image Segmentation* [University of Freiburg]. 2015. Available also from: https://lmb.informatik.uni-freiburg.de/people/ronneber/u-net/. Online. Available from: https://lmb.informatik.uni-freiburg.de/people/ronneber/u-net/. Accessed: 01-10-2025.
26. KRAGSTERMAN, Pär. *The Ultimate Guide to Preprocessing Medical Images: Techniques, tools, and best practices for enhanced diagnosis* [Online]. 2024. Available also from: https://collectiveminds.health/articles/the-ultimate-guide-to-preprocessing-medical-images-techniques-tools-and-best-practices-for-enhanced-diagnosis. Online. Available from: https://collectiveminds.health/articles/the-ultimate-guide-to-preprocessing-medical-images-techniques-tools-and-best-practices-for-enhanced-diagnosis. Accessed: 01-10-2025.
27. BHUVAJI, Sartaj. *Brain Tumor Classification (MRI)*. Kaggle, 2025. Available also from: https://www.kaggle.com/datasets/sartajbhuvaji/brain-tumor-classification-mri. Online dataset, Accessed: 01-10-2025.

28. DORIANCRAY13. *What are different evaluation metrics used to evaluate image segmentation models?* [Online]. 2025. Available also from: https://www.geeksforgeeks.org/computer-vision/what-are-different-evaluation-metrics-used-to-evaluate-image-segmentation-models/. Accessed: 01-10-2025.
29. NICKPARVAR, Masoud. *Brain Tumor MRI Dataset*. Kaggle, 2020. Available also from: https://www.kaggle.com/datasets/masoudnickparvar/brain-tumor-mri-dataset. Accessed: 06-06-2025.
30. CHENG, Jun. *Brain tumor dataset*. figshare, 2017. Version 8. Available from DOI: 10.6084/m9.figshare.1512427.v8. Accessed: 06-10-2025.
31. CARR, D.H.; BROWN, J.; BYDDER, G.M.; STEINER, R.E.; WEINMANN, H.J.; SPECK, U.; HALL, A.S.; YOUNG, I.R. Gadolinium-DTPA as a contrast agent in MRI: initial clinical experience in 20 patients. *American Journal of Roentgenology* [Online]. 1984, vol. 143, no. 2. Available from DOI: 10.2214/ajr.143.2.215. Online.
32. BHARATH, K. *U-Net architecture for image segmentation* [Online]. DigitalOcean, 2025. Available also from: https://www.digitalocean.com/community/tutorials/unet-architecture-image-segmentation. Accessed: 27-08-2025.
33. ISENSEE, Fabian; SCHELL, Michael; TURSUNOVA, Irina; BRUGNARA, Gabriele; BONEKAMP, David; NEUBERGER, Ulf; WICK, Anna; SCHLEMMER, Heinz-Peter; HEILAND, Sabine; WICK, Wolfgang; BENDSZUS, Martin; MAIER-HEIN, Klaus H.; KICKINGEREDER, Philipp. Automated Brain Extraction of Multi-Sequence MRI Using Artificial Neural Networks. *Human Brain Mapping*. 2019, pp. 1–13. Available from DOI: 10.1002/hbm.24750.
34. ZHANG, Yifan; WANG, Minghui; LIU, Rui; ZHAO, Chen; HUANG, Xin. YOLO-MED: Multi-Task Interaction Network for Biomedical Images. *arXiv preprint arXiv:2403.00245*. 2024. Available also from: https://arxiv.org/html/2403.00245v1. Accessed: 06-10-2025.
35. ANUVAB, Md. Shahriar Rahman; SULTANA, Mishkat; HOSSAIN, Md. Atif; DAS, Shashwata; CHOWDHURY, Suvarthi; RAHMAN, Rafeed; DOFADAR, Dibyo Fabian; RANA, Shahriar Rahman. *Figure 6: Model Comparison of FL U-Net, U-Net and YOLO* [Online]. ResearchGate, 2024. Available also from: https://www.researchgate.net/figure/Model-Comparison-of-FL-U-Net-U-Net-and-YOLO_fig5_379652483. Accessed: 06-10-2025.
36. MILESIAL. *PyTorch-UNet: PyTorch Implementation of the U-Net for Image Semantic Segmentation* [Online]. GitHub, 2025. Available also from: https://github.com/milesial/Pytorch-UNet. Accessed: 07-10-2025.
37. KRISHNAMURTHY, Bharath. *An Introduction to the ReLU Activation Function* [Online]. Built In, 2024. Available also from: https://builtin.com/machine-learning/relu-activation-function. Accessed: 10-07-2025.
38. MAILFORRIYA2708. *Training Data vs Testing Data* [Online]. 2023. Available also from: https://www.geeksforgeeks.org/python/training-data-vs-testing-data/. Accessed: 03-10-2025.
39. PYTHON GUIS. *PyQt5 Tutorial 2025: Create Python GUIs with Qt* [Online]. Python GUIs, 2025. Available also from: https://www.pythonguis.com/pyqt5-tutorial/. Accessed: 13-10-2025.

40. PYTORCH CONTRIBUTORS. *Welcome to PyTorch Tutorials* [Online]. PyTorch, 2023. Available also from: https://docs.pytorch.org/tutorials/. Accessed: 13-10-2025.
41. OSL DYNAMICS. *Loading Data* [Online]. OSL Dynamics, 2025. Available also from: https://osl-dynamics.readthedocs.io/en/v1.4.4/tutorials_build/data_loading.html. Accessed: 13-10-2025.
42. NUMPY DEVELOPERS. *NumPy Quickstart — NumPy v2.4.dev0 Manual* [Online]. NumPy, 2025. Available also from: https://numpy.org/devdocs/user/quickstart.html. Accessed: 13-10-2025.
43. ÖZBAY, Erdal; ÖZBAY, Feyza Altunbey. Automated Brain Tumor MRI Segmentation Using ARU-Net with Residual-Attention Modules. *Diagnostics*. 2025, vol. 15, no. 18. Available from DOI: 10.3390/diagnostics15182326.
44. R, Preetha; M, Jasmine Pemeena Priyadarsini; S, Nisha J. Brain Tumor Segmentation Using Multi-Scale Attention U-Net with EfficientNetB4 Encoder for Enhanced MRI Analysis. *Scientific Reports*. 2025, vol. 15, no. 1. Available from DOI: 10.1038/s41598-025-94267-9. Online.
45. NASIM, MD Abdullah Al; MUNEM, Abdullah Al; ISLAM, Maksuda; PALASH, Md Aminul Haque; HAQUE, MD. Mahim Anjum; SHAH, Faisal Muhammad. Brain Tumor Segmentation using Enhanced U-Net Model with Empirical Analysis. *arXiv* [Preprint at arXiv]. 2022. Available from eprint: arXiv:2210.13336. Accessed: 08-10-2025.
46. YALA, Adam; LEHMAN, Constance D.; SCHUSTER, Tal; PORTNOI, Tamar; BARZILAY, Regina. Toward Robust Mammography-Based Models for Breast Cancer Risk. *Science Translational Medicine* [Online]. 2021, vol. 13, no. 578, pp. 1–2. Available from DOI: 10.1126/scitranslmed.aba4373.
47. CENTER FOR BIOMEDICAL IMAGE COMPUTING AND ANALYTICS (CBICA). *Brain Tumor Segmentation (BRATS) Challenge* [Online]. Perelman School of Medicine, University of Pennsylvania, 2025. Available also from: https://www.med.upenn.edu/cbica/brats/. Accessed: 13-10-2025.
48. ARM LTD. *What is a Convolutional Neural Network (CNN)?* [Online]. Arm | The Architecture for the Digital World, 2024. Available also from: https://www.arm.com/glossary/convolutional-neural-network. Accessed: 13-10-2025.
49. INDEED EDITORIAL TEAM. *What is a GUI (Graphical User Interface)? Definition, elements and benefits* [Online]. Indeed Career Guide, 2025. Available also from: https://www.indeed.com/career-advice/career-development/gui-meaning. Accessed: 07-10-2025.
50. BHATI, Mukul. *What is Inference Engine* [Online]. Nected, 2025. Available also from: https://new.nected.ai/blog/what-is-inference-engine. Accessed: 14-10-2025.
51. NATIONAL INSTITUTE OF BIOMEDICAL IMAGING AND BIOENGINEERING. *Magnetic Resonance Imaging (MRI)* [Online]. -. Available also from: https://www.nibib.nih.gov/science-education/science-topics/magnetic-resonance-imaging-mri. Accessed: 13-10-2025.
52. LABONNE, Maxime. *What is a Tensor in Deep Learning?* [Online]. 2022. Available also from: https://medium.com/data-science/what-is-a-tensor-in-deep-learning-6dedd95d6507. Accessed: 13-10-2025.

C.2 List of Figures

The title page was created using Adobe Photoshop. The background was created with Photshop's included generative AI, using the following prompt: "Place a small MRI scan of a brain at the center bottom third of the image. In the MRI, there should be a tumor highlighted in red, with a bounding box labeled 'Tumor.' Include effects to indicate that this tumor was detected using AI."

D. Materials

D.1 Software and Tools Used

- Perplexity AI was used as a language tool to find the fitting jargon.
- Perplexity AI was used to transform citations into .bib format which is used in BibTeX.
- Microsoft's built in Copilot in Visual Studio code was used to assist debugging and to write performance evaluation scripts.
- Trinka was used to proofread the complete thesis and correct grammatical mistakes.
- DeepL was used to accurately translate the interview transcript to english.

D.2 Hardware Used

Development and training was performed on a custom built computer with the following specifications:

- ASUS ROG Strix B550-F
- AMD Ryzen 5 5600X
- Gigabyte GeForce RTX 3060
- 16 GB of RAM

E. Example Code / Scripts

All source code can be found in this GitHub repositry: https://github.com/milkman-dk/NOSAv2

```python
# Double convolution block
class DoubleConv(nn.Module):
    def __init__(self, in_ch, out_ch, dropout=0.1):
        super(DoubleConv, self).__init__()
        self.conv = nn.Sequential(
            nn.Conv2d(in_ch, out_ch, 3, padding=1),
            nn.BatchNorm2d(out_ch),
            nn.ReLU(inplace=True),
            nn.Conv2d(out_ch, out_ch, 3, padding=1),
            nn.BatchNorm2d(out_ch),
            nn.ReLU(inplace=True),
            nn.Dropout2d(dropout)
        )

    def forward(self, x):
        return self.conv(x)
```

Figure 27: Double convolutional block used in all versions.

```python
# Combined Dice + BCE Loss Function
class DiceBCELoss(nn.Module):
    def __init__(self):
        super(DiceBCELoss, self).__init__()
        # Initialization: define BCE loss with logits for numerical stability
        self.bce = nn.BCEWithLogitsLoss()

    def forward(self, inputs, targets, smooth=1):
        # Flatten inputs and targets to 1D vectors for element-wise operations
        inputs = inputs.view(-1)
        targets = targets.view(-1)
        # Compute BCE loss between raw network outputs (logits) and ground truth masks
        bce_loss = self.bce(inputs, targets)
        # Convert logits to probabilities for Dice calculation
        probs = torch.sigmoid(inputs)
        # Calculate the intersection as the sum of element-wise multiplication
        intersection = (probs * targets).sum()
        # Compute Dice coefficient; the loss is 1 minus this coefficient
        dice_loss = 1 - (2. * intersection + smooth) / (probs.sum() + targets.sum() + smooth)
        # Return the combined loss
        return bce_loss + dice_loss
```

Figure 28: Custom loss function that combines dice loss and BCE loss

```python
class UNet(nn.Module):
    def __init__(self, n_channels=1, n_classes=1, base_filters=64):  # NOSAv2: 64 base filters (vs 32 in v1)
        super(UNet, self).__init__()

        # ENCODER (Contracting Path)
        self.inc = DoubleConv(n_channels, base_filters)  # Encoder Block 1
        self.down1 = nn.Sequential(
            nn.MaxPool2d(2),
            DoubleConv(base_filters, base_filters * 2)  # 128 filters
        )  # Encoder Block 2
        self.down2 = nn.Sequential(
            nn.MaxPool2d(2),
            DoubleConv(base_filters * 2, base_filters * 4)  # 256 filters
        )  # Encoder Block 3
        self.down3 = nn.Sequential(
            nn.MaxPool2d(2),
            DoubleConv(base_filters * 4, base_filters * 8)  # 512 filters
        )  # Encoder Block 4
        self.down4 = nn.Sequential(
            nn.MaxPool2d(2),
            DoubleConv(base_filters * 8, base_filters * 16)  # Bottleneck: 1024 filters (vs 256 in v1)
        )  # Bottleneck

        # DECODER (Expanding Path)
        self.up1 = nn.ConvTranspose2d(base_filters * 16, base_filters * 8, 2, stride=2)  # UpSampling 1
        self.conv1 = DoubleConv(base_filters * 8 + base_filters * 8, base_filters * 4)  # Decoder Block 1
        self.up2 = nn.ConvTranspose2d(base_filters * 4, base_filters * 4, 2, stride=2)  # UpSampling 2
        self.conv2 = DoubleConv(base_filters * 4 + base_filters * 4, base_filters * 2)  # Decoder Block 2
        self.up3 = nn.ConvTranspose2d(base_filters * 2, base_filters * 2, 2, stride=2)  # UpSampling 3
        self.conv3 = DoubleConv(base_filters * 2 + base_filters * 2, base_filters)      # Decoder Block 3
        self.up4 = nn.ConvTranspose2d(base_filters, base_filters, 2, stride=2)          # UpSampling 4
        self.conv4 = DoubleConv(base_filters + base_filters, base_filters)              # Decoder Block 4

        # OUTPUT
        self.outc = nn.Conv2d(base_filters, n_classes, 1)  # Final classification (1x1 conv)

    def forward(self, x):
    # Encoder
        x1 = self.inc(x)        # 64 filters
        x2 = self.down1(x1)     # 128 filters
        x3 = self.down2(x2)     # 256 filters
        x4 = self.down3(x3)     # 512 filters
        x5 = self.down4(x4)     # 1024 filters (bottleneck)
        # Decoder
        x = self.up1(x5)        # upsampled to 512 filters
        x4_resized = x4
        x = self.conv1(torch.cat([x, x4_resized], dim=1))  # 1024 channels -> 256 filters
        x = self.up2(x)         # upsampled to 256 filters
        x3_resized = x3
        x = self.conv2(torch.cat([x, x3_resized], dim=1))  # 512 channels -> 128 filters
        x = self.up3(x)         # 128 filters
        x2_resized = x2
        x = self.conv3(torch.cat([x, x2_resized], dim=1))  # 256 channels -> 64 filters
        x = self.up4(x)         # 64 filters
        x1_resized = x1
        x = self.conv4(torch.cat([x, x1_resized], dim=1))  # 128 channels -> 64 filters
        x = self.outc(x)

        return x  # NOSAv2: Returns raw logits (no sigmoid)
```

Figure 29: Adapted U-Net structure version 2, with the changes from version 1 highlighted.

```python
# Training loop with mixed precision, scheduler, and optimized data loading
def train_unet(mat_dir, epochs=96, batch_size=32, lr=1e-3, target_size=(256, 256)):
    # Prepare dataset and dataloader for batch processing
    dataset = MatlabTumorDataset(mat_dir, target_size=target_size)
    loader = DataLoader(dataset, batch_size=batch_size, shuffle=True, num_workers=4, pin_memory=True)
    # Use available GPU or default to CPU
    device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
    # Instantiate the 3D U-Net model and transfer to device
    model = UNet().to(device)
    # Initialize the combined loss function
    criterion = DiceBCELoss()
    # Use Adam optimizer with specified learning rate
    optimizer = optim.Adam(model.parameters(), lr=lr)
    # Apply cosine annealing scheduler for learning rate decay over epochs
    scheduler = optim.lr_scheduler.CosineAnnealingLR(optimizer, T_max=epochs)
    # Enable mixed precision training for faster computation
    scaler = torch.cuda.amp.GradScaler()

    # Enable cuDNN auto-tuning for optimal performance
    torch.backends.cudnn.benchmark = True

    # Main training loop over epochs
    for epoch in range(epochs):
        model.train()
        running_loss = 0.0
        for images, masks in loader:
            # Transfer batch to GPU if available
            images, masks = images.to(device, non_blocking=True), masks.to(device, non_blocking=True)
            optimizer.zero_grad()

            # Automatic Mixed Precision (AMP) context: speeds up training while maintaining accuracy
            with torch.cuda.amp.autocast(device_type='cuda'):
                outputs = model(images)  # Forward pass: output logits
                loss = criterion(outputs, masks)  # Compute combined loss

            # Backpropagation with scaled loss to avoid underflow in float16
            scaler.scale(loss).backward()
            # Step optimizer: unscale and perform parameter update
            scaler.step(optimizer)
            # Update the scale factor for next iteration
            scaler.update()

            # Accumulate loss for monitoring
            running_loss += loss.item() * images.size(0)

        # Update learning rate according to schedule
        scheduler.step()
        print(f"Epoch {epoch + 1}/{epochs}, Loss: {running_loss / len(dataset):.4f}")

    # Save trained model weights
    torch.save(model.state_dict(), "unet_brain_tumor_v1.3.pth")
    print("Optimized model saved as unet_brain_tumor_v1.3.pth")
```

Figure 30: Custom training loop using an Adam optimizer

F. Complete Transcript of the Interview with Dr. Diepers

Lüscher: I already sent you most of the questions, and you commented on them in the email. I will simply ask the questions once more, and if you have a bit more detail or perhaps some additions, I can ask a few additional questions. So first: In which areas and to what extent is AI currently used in radiology, at the KSA?

Diepers: It is being used increasingly and already quite extensively. The focus, of course, is on diagnostics, that is, on image analysis – recognizing and defining pathologies. But there are also many other areas of application. It starts with the evaluation of referrals. There are applications that record how frequently certain diagnoses lead to certain examinations. These applications then develop suggestions on how to coordinate staff planning and equipment utilization so that one can best cover peak demands.

There are also organizational AI systems. I don't even know exactly what the technical term is – perhaps process analytical AI – that can give tips on how to organize work more efficiently.

Another example is not directly about diagnostics, but about the evaluation of very complex so called functional examinations. That means, it's not simply a matter of creating an imaging representation, but rather making bodily functions or organ functions visible – ones that otherwise could not be depicted at all.

An example would be the assessment of the brain's blood flow quality – the so called perfusion. One can measure the speed and the blood volume with which a certain amount of brain tissue is perfused per unit of time and then process that graphically. An AI like application then produces colorful topographical maps that show where deficits or over perfusion are occurring. A similar application exists in magnetic resonance imaging for analyzing tissue changes, in order to recognize the type of pathology better – for example, distinguishing dangerous from conspicuous but harmless findings.

When one speaks of AI in radiology, most people – and this is indeed the main area in which AI applications are currently used – think of image assessment, image analysis, and diagnosis. That is certainly the main focus, yes.

Lüscher: Where do you see the limits of AI, and what, in your opinion, should not be left to AI in radiology?

Diepers: Patient care, communication with patients, the interpersonal aspects – that can by no means be left to AI. AI can capture, process, and analyze information extremely quickly, but it does not have

the ability to react to moods or to sense a patient's condition. If a patient is unsettled or anxious, they would have to state that explicitly for the AI to respond. As a doctor, however, you sense it even when the patient tries to hide it, and you can draw very important conclusions from that.

The doctor patient relationship does not only consist of recognizing disturbances and deciding how to treat them, but it often requires a careful approach. One must be capable of recognizing things that the patient isn't even consciously aware of. This relationship is something that – as of today – no algorithm, not even a highly developed AI, can replace.

Even in radiology – even if it is "only" about image interpretation – it is never sufficient for image data simply to be analyzed by AI. There must always be someone who reviews the results and judges whether they are correct and appropriate. AI may be more precise than humans, but integrating what is seen into the context of patient care is a human task.

For example, if a patient has a tumor that is discovered incidentally, it is often wrong to tell them bluntly right away. One must build up and prepare the conversation and tailor it to the personality of the patient. It's not about withholding information, but about conveying it with sensitivity and the right choice of words.

Lüscher: So again, the interpersonal aspect?

Diepers: Exactly.

Lüscher: What I have read is that there's often a problem with responsibility – so, if one lets an AI work, who then takes responsibility for its decisions or mistakes?

Diepers: Yes, that is, to my knowledge, a completely unsolved problem. It must not be the case that responsibility for malfunctions is pushed solely onto the user – that would be a massive obstacle to development and progress.

I think there is still no universally valid regulation. At present, the user of an AI bears responsibility for whether it works correctly. One is obliged to check the results before they are implemented in decisions.

But if an AI makes an error that cannot be detected through normal supervision techniques, then it is unclear who is liable. To my knowledge, these are currently case by case decisions. And the EU Commission is discussing how this should be regulated by law in the future – especially in cases where AI errors lead to personal injury or material damage.

Lüscher: Yes, I have also read a lot about that – especially about the black box problem, meaning that one can hardly correct AI errors because one doesn't know "what it is thinking."

Diepers: Yes.

Lüscher: And how about data protection? How is that solved or attempted to be solved?

Diepers: That is often not properly solved. The attempt is made by using only anonymized data or by separating personal information so that it can no longer be assigned to a person from the outside.

How well that works, I cannot tell you – but that it sometimes doesn't, I know from daily practice. For example, we have an AI in use that checks in head CTs whether an acute hemorrhage is present. It runs quietly in the background; many don't even use it, because one usually recognizes hemorrhages oneself. It's meant for cases when one is uncertain.

The AI can then help to make a decision: Is it a real hemorrhage or merely an artifact? The reporting doctor uses the AI as a tool but writes the findings independently. However, the AI notification becomes part of the stored image data. If the referring physician later looks at the examination, they can see the AI alert "Attention – hemorrhage," although the report correctly says "no hemorrhage."

That creates a contradiction and can lead to misunderstandings – the colleague might think the radiologists overlooked something, inform the patient, and cause unnecessary anxiety and effort. That's essentially a data protection problem.

Actually, the reporting doctor should have the option to block the AI's additional information from others – but that is not the case.

Another data protection issue concerns the training data itself. An AI that interprets images must have seen tens or hundreds of thousands of cases in order to learn. No manufacturer can genuinely guarantee that all of those cases were fully anonymized. This means that at the developers' sites, large amounts of personal data inevitably exist – and that, to my knowledge, is likewise an unsolved problem.

Lüscher: You've already touched on it – the question of how to communicate information to patients. For example, in preventive diagnostics: MIRAI AI is an AI that calculates, based on mammograms, whether a patient may develop breast cancer within the next five years. What is your stance on that?

Diepers: Critical. That, of course, is a personal assessment. In medicine, such statistical, prognostic statements are very popular – among doctors as well as among patients. People like to work with concrete, seemingly tangible predictions about the future, such as "You have a fifteen percent risk that this change will one day turn into cancer."

But I consider that dangerous, because those calculations are based on statistical models that depict biological systems only inadequately. The development of a disease depends on countless individual factors – this complexity can only be very roughly captured by AI.

I therefore fear that such systems create pseudo accuracy and promote a misleading sense of certainty. An experienced doctor can often, through conversation and intuition alone, assess much better what is likely to happen than an AI that simply evaluates data.

So I am personally skeptical, even though the desire for such AI functions will certainly continue to grow.

Lüscher: How is your role as a radiologist changing with AI, and do you think it will change greatly in the future?

Diepers: Yes, it is already changing – towards more throughput. AI is helpful, and every help is used

to increase efficiency. Examinations and reporting proceed faster; more can be accomplished. But that leads to less patient contact. Radiology already has limited contact, and with AI it becomes even less. Especially in neuroradiology, where we have more direct contact than in general radiology, this is felt strongly. These patient contacts cannot be accelerated by AI.

At the same time, it is expected that we achieve more in less time. Pressure increases, quality is expected to remain consistently high – like with machines.

There is hardly any acceptance now for doctors having individual strengths and weaknesses. In summary: We can work faster with AI, but the pressure of expectations rises disproportionately, and the doctor patient relationship, as a central element of medical work, fades into the background.

Lüscher: Do you think that will get worse in the future – even more pressure as AIs become more advanced?

Diepers: Yes, I think so. At some point, such concerns will perhaps just be shrugged off. I consider it quite conceivable that there will be companies offering: "With us you get MRI diagnostics by AI – much faster and cheaper than at the doctor's." If that is not prohibited by law, there could be radiological services without radiologists. Then you would only need staff to position the patients in the machines; the rest would be done by AI. Similar to how it is today in laboratory medicine.

Lüscher: How does that work in laboratory medicine? I'm not familiar with that.

Diepers: As a laboratory physician, you primarily organize that the incoming samples are correctly processed. The actual analyses are performed by machines: concentrations, enzyme activities, cell counts – all that is measured automatically and printed in tables.

The laboratory physicians mainly monitor that their devices function correctly. A similar automation, I think, is conceivable in radiology as well.

Lüscher: Yes, that was pretty much all the questions I had. Thank you very much for the detailed answers. It was very interesting and thank you for taking the time for me and my work.

Diepers: My pleasure. If you have time and interest, I'd be curious how your work turns out. AI in radiology is a very current topic. And a commonly voiced thesis is: "One day all radiologists will be replaced by AI." That, of course, will not happen. But whether the demand for radiologists might actually grow with increasing AI activity in the future – that is indeed something to consider.

Alte Kantonsschule Aarau, Bahnhofstrasse 91, CH-5001 Aarau
Telefon +41 62 834 67 00, info@altekanti.ch, www.altekanti.ch

ALTE KANTI

AUTORINNEN UND AUTOREN VON MATURAARBEITEN

Kenntnisnahmen (Plagiatserkennung und Urheberrecht)

Kenntnisnahme Plagiatserkennung:

„Ich nehme zur Kenntnis, dass meine Arbeit zur Überprüfung der korrekten und vollständigen Angabe der Quellen mit Hilfe einer Software (Plagiatserkennungstool) geprüft wird. Zu meinem eigenen Schutz wird die Software auch dazu verwendet, später eingereichte Arbeiten mit meiner Arbeit elektronisch zu vergleichen und damit Abschriften und eine Verletzung meines Urheberrechts zu verhindern. Falls Verdacht besteht, dass mein Urheberrecht verletzt wurde, erkläre ich mich damit einverstanden, dass die Schulleitung meine Arbeit zu Prüfzwecken herausgibt."

Kenntnisnahme Urheberrecht:

„Ich nehme zur Kenntnis, dass für Maturaarbeiten ein öffentliches Interesse an der Aufbewahrung für spätere Forschungszwecke und ein dafür notwendiges Zugänglichmachen besteht. Ich bin damit einverstanden, dass die Alte Kanti meine Maturaarbeit schulintern für alle Schulangehörigen einsehbar ablegt (Sharepoint) und dass sie allenfalls auch als Print-Version in der Auslage des Medienzentrums öffentlich zugänglich gemacht wird (z.B. bei einer Prämierung). Bei einer Veröffentlichung der Maturaarbeit durch den Autor / die Autorin muss zudem die Zustimmung der Betreuungsperson sowie über die Schulleitung die Zustimmung der Schule eingeholt werden."

Name: Lüscher Vorname: Mikkel Abt.: G22K

Aarau, 16.10 (Datum) Unterschrift: 

Bitte zweite Seite beachten.

KANTON AARGAU

Anhang:

www.copy-stop.ch
Regelung über die Abgabe der Maturaarbeit an Mittelschulen

Ausführliche Beschreibung des Verfahrens und der Konsequenzen:

Unsere Schule hat festgelegt, dass schriftliche Arbeiten, insbesondere die Abschlussarbeiten, zusätzlich zur Printversion auch in elektronischer Form abgegeben werden müssen.

Sie bereiten die Datei folgendermassen vor:

1. Alle Texte Ihrer Arbeit müssen in einer einzigen Datei im Format Word oder PDF abgespeichert werden.
2. Die Bezeichnung der Datei soll internettauglich sein, also keine Umlaute wie ä, ö und ü und auch keine Sonderzeichen wie é und keine Leerschläge enthalten. Nennen Sie aus Gründen des Datenschutzes nicht Ihren Namen, sondern nur das Jahr der Abgabe und ein bis zwei Stichworte des Titels und verbinden Sie alles mit Bindestrichen. Beispiel: 13-Ueberschwemmung-Auenwaelder
3. Löschen Sie alle Bilder. Die Datei wird damit in der Regel kleiner als 0.5 MB.
4. Aus Gründen des Datenschutzes müssen Ihr Name, der Name von Drittpersonen und der Name der Lehrperson, bei der die Arbeit eingereicht wurde, in der Datei überall gelöscht werden (Titelseite, Fuss- oder Kopfzeile oder wo sie sonst noch vorkommen).

Die Lehrperson überprüft Ihre elektronisch eingereichte Arbeit auf Textstellen, die Sie nicht selbst verfasst haben und die nicht ordnungsgemäss mit den Quellenangaben versehen sind, das heisst so genannte Plagiate. Sie übergibt Ihre Arbeit mittels eines Webinterfaces dem professionellen Plagiatserkennungstool „copy-stop“, das von den Universitäten Hannover und Braunschweig entwickelt wurde. Die Arbeiten werden gleichzeitig mit dem Prüfen auch in eine geschlossene Datenbank eingelagert.

Der „copy-stop“-Prüfcomputer in Braunschweig lädt sich eine Kopie der Arbeit kurzzeitig in den Arbeitsspeicher, prüft die Arbeit und löscht sie anschliessend. Während des Prüfvorgangs ist die Arbeit für niemanden einsehbar. Zusätzlich zur Überprüfung wird der Text der Arbeit indexiert, das heisst mit den wichtigsten Stichworten erfasst, wobei die Indexierung keine Rückschlüsse auf den ganzen Text erlaubt. Ihre Urheberrechte werden mit diesem Verfahren garantiert.

Die Indexierung ermöglicht es der Software „copy-stop“, die Texte der Arbeit nicht nur mit den im Internet publizierten Texten zu vergleichen, sondern auch mit allen bisher schon geprüften und in der Datenbank abgespeicherten Arbeiten einen Textvergleich vorzunehmen.

Falls Ihre Arbeit grösstenteils von einer fremden Arbeit kopiert wurde, ersucht Ihre Schulleitung das Mittelschul- und Berufsbildungsamt des Kantons Zürich um die Herausgabe der Originalarbeit. Wenn der Vergleich eine hohe Übereinstimmung ergibt, weist die Schule Ihre Arbeit zurück.
Falls eine Arbeit Texte aus dem Internet enthält und die Quellenangaben dazu fehlen, dann werden alle diese Textstellen gestrichen und als nichtexistierend betrachtet. Die Bewertung des übrig bleibenden Textgerippes führt in weniger gravierenden Fällen zu einer schlechteren Note. In gravierenden Fällen wird die Arbeit zurückgewiesen oder sie hat gar die Nichtzulassung zur Maturitätsprüfung zur Folge. Disziplinarische Massnahmen bleiben vorbehalten.