Brain MRI Segmentation with U-Net

Chieh-Ju Wu
School of ITM
wu5@kth.se

Daniel Grönås
School of ITM
dgronas@kth.se

Fredrik Mazur
School of ITM
fomazur@kth.se

Niclas Määttä
School of ITM
nmaatta@kth.se

Abstract

With the progress of deep learning, there are huge potentials within the medical sector for medical imaging and diagnostic. In 2015 a paper with a new architecture for biomedical image segmentation, U-Net, was published. This paper covers the architecture of an U-Net neural network and its implementation of segmenting brain tumors from MRI scans. Two experiments were done. In the first test, the best optimizer between SGD and Adam was chosen. Both tests were conducted with binary cross-entropy loss, and Adam received a higher accuracy. The second test was to test three different loss functions, binary cross-entropy loss, dice coefficient loss, and a combination of the two. These tests were performed on Adam, since it was the best performing optimizer from experiment 1. The best test accuracy achieved was a dice score of 76.48 $\%$ with Adam optimizer and binary cross-entropy loss.

1 Introduction

Up until the beginning of the 20th century, the interior of human bodies had remained a mystery in medical diagnostics. With the introduction of X-ray technology and magnetic resonance imaging (MRI), inspecting the internal of a human body was made possible. MRI is now widely used worldwide for diagnostics; however, a clear image of the internal does not guarantee an accurate diagnosis from the doctors.

With the development of deep learning, medical imaging has become an interesting topic within image segmentation, since well functioning models could save lives. The development of artificial neural networks have a huge potential within the medical imaging and diagnostics sectors [1].

The purpose of this project was to investigate the possibilities with the U-net architecture. This was done by performing brain tumor segmentation from MRI images. The best components of the model was decided through testing of different optimizers and loss functions.

2 Related Work

This project is based on the architecture introduced in the original publication: *U-Net: Convolutional Networks for Biomedical Image Segmentation*. In the original paper, the U-Net model was able to achieve an average IOU (intersection over union) score of 92% [2]. Ever since the publication, numerous contributions of U-Net implementations have been made on image segmentation tasks. For instance, a paper about skin lesion segmentation, which implemented different types of U-Net models, got a dice score of 85.33% [3]. Furthermore, in 2017, there was an online challenge called the "Carvana Image Masking Challenge". The goal of the challenge was to get the highest segmentation accuracy on a dataset of cars with pre-determined masks. In this challenge, some of the best contributions were implementations of U-Nets, with accuracy over 99% [4].

3 Dataset

To perform MRI segmentation, a dataset from Kaggle was used. The dataset includes MRI of 110 patients from the *The Cancer Genome Atlas* (TCGA) collection. The dataset in total adds up to 3928 images of MRI scans, along with the same amount of corresponding masks taken from healthy patients and of those suffering from lower-grade glioma (LGG) [5]. The images are of size 256x256 pixels and of type $.tif$.

3.1 Data-preprocessing

There are two types of MRI - true positive and true negative. Since the neural network model will not learn from true negative images (MRI without brain tumors), these images were removed. The dataset was further separated into two parts for training (75%) and testing (25%). This resulted in a dataset of 1043 images for training and 330 images for testing.

3.2 Data Augmentation

Due to the limitations of available images, data augmentation was used. By rotating, flipping horizontally and/or vertically, the original images were transformed into multiple new images. This provides more training samples which can potentially results in better training models. In this project, the augumentation was performed with the computer vision tool, *Albumentations* [6].

4 Methods

4.1 U-Net Architecture

There are two main changes of U-Net the architecture in this project comparing to the original, see section 2. The first change was the implementation of zero-padding at each convolutional layer, where the original paper did not use any padding. This avoids the need of cropping images, and thus, losing information when concatenating the skip connections in the contracting path of the network. The second change was the introduction of batch normalization in between each convolution and the activation layer for the purpose of increasing the training speed and test accuracy of the network model, this change was based on a study about batch normalization in convolutional neural networks found in [7].

The U-Net architecture of this project consists of an contracting path, which captures the context of the input pictures, and an expanding path enabling precise localization. These parts combined together give the network an U-shaped form, see figure 1. The contracting path follows a traditional architecture of a convolutional network. Each layer consists of a twice repeated 3x3 padded convolution, each convolution is followed by a batch normalization and a rectifier linear units (ReLU) activation. The output feature map is stored as a skip connection which will later be concatenated together with the expanding path. Moving downwards through the network, a down sampling is performed by a 2x2 max pooling operation with a stride of 2, which doubles the number of feature channels.

The expanding path has four repeated 2x2 transposed convolution (up-convolution) with a stride of 2 which reduced the number of feature channels by 2 at each step. The output is then concatenated with the corresponding skipped connection stored previously. This result is then sent through a double 3x3 padded convolution together with batch normalization and ReLU activation. At the output layer, a padded 1x1 convolution is done which outputs an one channel segmentation map.

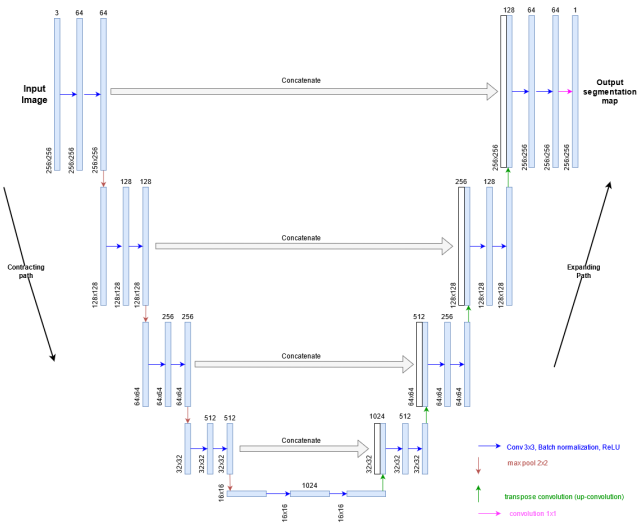


Figure 1: U-Net architecture implemented in the paper.

4.2 Accuracy

One way to compute the accuracy of the prediction is to compare the mask with the prediction at each pixel in the image. However, this method is not reliable when the classes of pixels are unbalanced. For example, in this project, the tumor only takes up a small part of the image, while the rest are mostly black pixels. So even if the prediction accuracy of the tumor was low, the overall prediction accuracy could still be high. This very high accuracy shows a misleading result instead of the real performance of the model [8].

In order to get a better accuracy measurement, the Sørensen-Dice coefficient was used. This is a method widely used in checking segmentation accuracy. It measures the total amount of pixels, in both the prediction and the target mask, as well as, the intersection of the overlapping areas. The equation for the dice coefficient is shown in equation 1, where, X is the prediction and Y is the ground truth, TP is True Positive, FP is False Positive and FN is False Negative.

$$DSC = \frac{2|X \cap Y|}{|X| + |Y|} = \frac{2TP}{2TP + FP + FN} \tag{1}$$

4.3 Optimizers

The Stochastic Gradient Descent (SGD) is a commonly used optimizer, which allows convergence without increased computing time per iteration for bigger training sets. The original paper of U-Net, mentioned in Section 2, uses SGD as its optimizer, and therefore it seemed like a reasonable test for this project. Moreover, it also gives the possibility to compare the training with other optimizers, such as Adam, an optimizer combining RMSProp and momentum (with some differences). Since Adam is robust in regards to hyperparameters [9], it was seen as the most appropriate optimizer to compare, due to the time limitation of this project.

To be able to compare the performance of SGD and Adaptive Estimation Momentum (Adam), SGD was implemented with a suitable learning rate, while the hyperparameters in the network utilizing the Adam optimizer was set as the recommended default.

4.4 Loss Functions

In order to find a good result, binary cross-entropy loss, dice loss and the combination of the two were used for comparison.

- Cross-Entropy Loss

Cross-entropy loss measures the performance of the classification model with an output of probability between 0 and 1. It is defined in Equation 2, where (t_i, p_i) = *(truth label, softmax probability for the i^{th} class)*.

$$L_{CE} = -\sum_{i=1}^{n} t_i \log(p_i) \tag{2}$$

- Dice Loss

  Dice loss measures the performance of the classification model with an output of probability between 0 and 1. It is defined in Equation 3, where X is the prediction and Y is the ground truth. As discussed previously, dice loss can reflect the actual model performance at class imbalanced problems.

$$L_{Dice} = 1 - DSC = 1 - \frac{2|X \cap Y|}{|X| + |Y|} \tag{3}$$

- Cross-Entropy Loss with Dice Loss

  By combining these two loss functions, as defined in Equation 4, the new combined loss function can capture the advantages of Cross-Entropy Loss for its smoother gradients and Dice Loss for solving class imbalanced problems together at the same time.

$$L_{Combined} = L_{CE} + L_{Dice} \tag{4}$$

5 Experiments

Two experiments were conducted in this project. Experiment 1 dived into finding the best optimizer. In the tests, SGD and Adam were evaluated with binary cross-entropy loss. The second experiment was to find the best loss function by implementing BCE, dice loss, and BCE+dice loss. An overview the experiments are presented in Table 1.

Table 1: Experiments Overview

Experiment	**Test Objects**	**Description**
1	ADAM and SGD	Performance comparison with BCE loss
2	Best optimizer from experiment 1	BCE, Dice, BCE+Dice

5.1 Experiment 1 - Optimizer comparison (SGD and Adam)

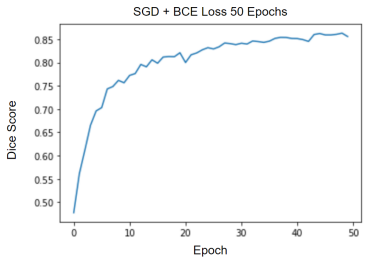
In experiment 1, the fixed hyperparameters and configurations of the models are shown in Table 2. The training with SGD optimizer is shown in Figure 2, and Figure 3 shows the training with Adam. Adam performed better as an optimizer and had a more stable training , compared to SGD. The final accuracies for the test set is shown in Table 3. The resulting prediction pictures when using Adam and SGD can be seen in appendix A.1 and A.2, respectively.

Table 2: Fixed Hyperparameters and Configurations of the Network

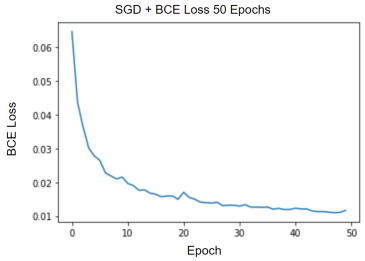
Hyperparameter	**Adam**	**SGD**
Batch size	8	2
Learning Rate	1e-4	1e-2
Momentum	None	0.9
Loss Function	Binary Cross Entropy	Binary Cross Entropy
Num of Epochs	50	50

Table 3: Test Accuracy

Optimizer	Epochs	Dice Coefficient
Adam	50	0.7648
SGD	50	0.68134

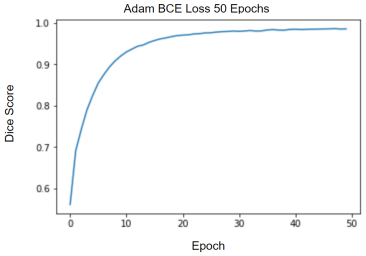


(a) Meane Accuracy per epoch

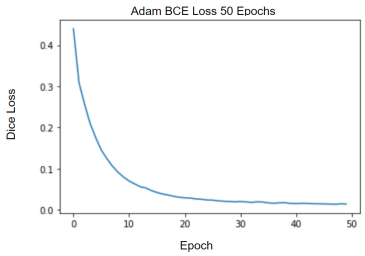


(b) Meane Loss per epoch

Figure 2: Dice score and loss plot for training with SGD



(a) Mean accuracy per epoch



(b) Mean Loss per epoch

Figure 3: Dice score and loss plot for training with Adam

From the results, it can be seen that Adam performed better than SGD with more stable training and higher test accuracy. Consequently, it was decided to use Adam for further investigation of loss functions. One important note is that due to time limitations, SGD was only tested with a batch size of 2, learning rate of 0.01 and momentum of 0.9. It might have performed better if a search for an optimal learning rate had been performed. Moreover, Adam was also only tested with one learning rate. Since Adam is not as sensitive to hyperparameter settings, it was concluded that is was valid to only use one learning rate.

An interesting thing was that the training loss plot was very low, especially for SGD. One explanation for this could be that BCE generates a very low loss for this type of segmentation tasks. Similar to what was discussed under accuracy, see 4.2, the dataset is unbalanced since the tumors only take up smaller fractions of the images. Therefore, this type of loss function will quickly show a low value in segmentation tasks, since all the black surroundings will be viewed as correct.

5.2 Experiment 2 - Loss function comparison for best optimizer

In experiment 2, Adam was used since it was the best performing optimizer in experiment 1, see 5.1. The hyper-parameters were set to be the same as shown for Adam in Table 2, excluding the loss function which was investigated. Three loss functions were tested in order to decide which one resulted in the highest accuracy. The loss functions were Binary Cross Entropy Loss, dice loss and the combination of the two. These can bee seen in Figure 3, 4, and 5, respectively.

From the results, the best performing loss function was BCE with an 0.7648 dice coefficient accuracy, see Table 4. The plots for the training with the different loss functions is shown in Figures 3-5. The resulting prediction pictures for BCE, Dice and BCE + Dice loss can bee seen in appendix A.1, A.3 and A.4, respectively.

Table 4: Test Accuracy

Loss Function	Optimizer	Epochs	Dice Coefficient
BCE	Adam	50	0.7648
Dice Loss	Adam	50	0.7413
BCE + Dice Loss	Adam	50	0.7381

(a) Mean Accuracy per epoch (b) Mean Loss per epoch

Figure 4: Dice Loss for Adam

(a) Mean Accuracy per epoch (b) Mean Loss per epoch

Figure 5: BCE and Dice Loss for Adam

Which can be seen in Figure 3-5, the curve is smoother for BCE loss. Furthermore, since the task uses dice coefficient as a measure of success, it was thought that dice coefficient loss or the combination of the two would be better measures. Another observation was that the training process was slower with the combination of both loss functions than using them separately. However, further testing might be needed before any final conclusions, since the test accuracies only differed from 0.7381 to 0.7648.

6 Conclusion

The usage of U-Net gave a pretty good performance of the image segmentation. However, it did not perform as well as previous work in similar fields. One reason for this could be the image resolution. In the original paper, the image resolution was 512x512, which is four times the resolution compared to the dataset used in this project. Resolution is an important part in image segmentation, and therefore a lower dice coefficient was expected. For future testing, another dataset could be used in order to compare the final results to previous work in a more exact manner.

Nonetheless, the brain MRI segmentation task was seen as a success. It was able to predict the main areas of where the tumor was in a functioning manner, even if the model sometimes has blurry edges around the tumor edges. With only 1043 training images, the U-Net architecture is thought to be a functioning method for this type of task.

References

[1] Alexander Selvikvpg Lundervold; Arvid Lundervold. "An overview of deep learning in medical imaging focusing on MRI". In: (2018). DOI: https://doi.org/10.1016/j.zemedi.2018.11.002.

[2] Olaf Ronneberger; Philipp Fischer; Thomas Brox. "U-Net: Convolutional Networks for Biomedical Image Segmentation". In: (2015).

[3] Peng; et.al. Tang. "Efficient skin lesion segmentation using separable-Unet with stochastic weight averaging". In: 178 (2019). DOI: https://doi.org/10.1016/j.cmpb.2019.07.005.

[4] Carvana. *Carvana Image Masking Challenge*. URL: https://www.kaggle.com/c/carvana-image-masking-challenge/overview. (Accessed: 27.04.2021).

[5] Mateusz Buda. *Brain MRI segmentation*. URL: https://www.kaggle.com/mateuszbuda/lgg-mri-segmentation. (Accessed: 26.05.2021).

[6] Alexander Buslaev et al. "Albumentations: Fast and Flexible Image Augmentations". In: *Information* 11.2 (2020). ISSN: 2078-2489. DOI: 10.3390/info11020125. URL: https://www.mdpi.com/2078-2489/11/2/125.

[7] Vignesh Thakkar, Suman Tewary, and Chandan Chakraborty. "Batch Normalization in Convolutional Neural Networks — A comparative study with CIFAR-10 data". In: *2018 Fifth International Conference on Emerging Applications of Information Technology (EAIT)*. 2018, pp. 1–5. DOI: 10.1109/EAIT.2018.8470438.

[8] Ekin, Tiu. *Metrics to Evaluate your Semantic Segmentation Model*. URL: https://towardsdatascience.com/metrics-to-evaluate-your-semantic-segmentation-model-6bcb99639aa2. (Accessed: 22.05.2021).

[9] Ian Goodfellow; Yoshua Bengio; Aaron Courville. *Deep Learning*. Massachusetts Institute of Technology. ISBN: 9780262035613.

A Appendix

A.1 Predictions Adam and BCE loss

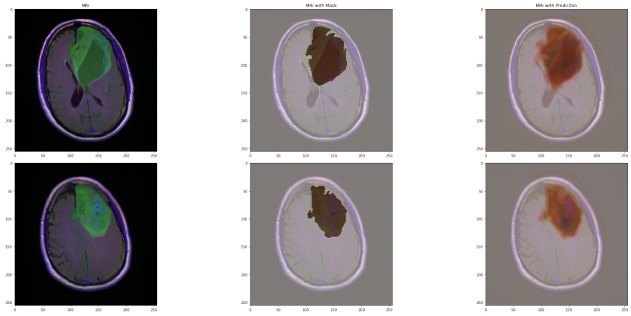


Figure 6: Predictions using Adam and BCE loss.

A.2 Predictions SGD and BCE loss

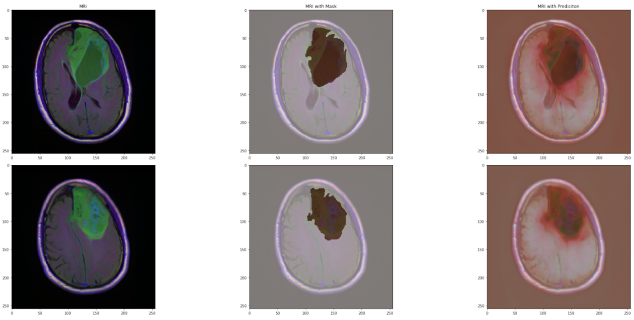


Figure 7: Predictions using SGD and BCE loss.

A.3 Predictions Adam and Dice loss

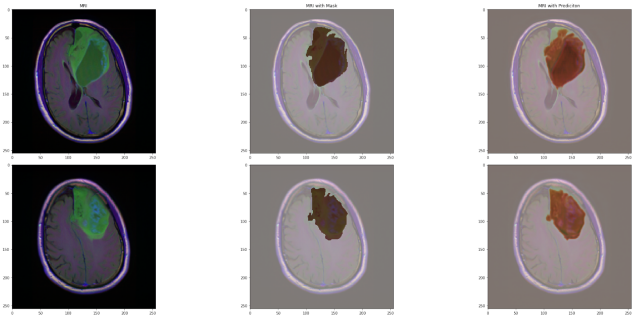


Figure 8: Predictions using Adam and Dice loss.

A.4 Predictions Adam and BCE + Dice loss

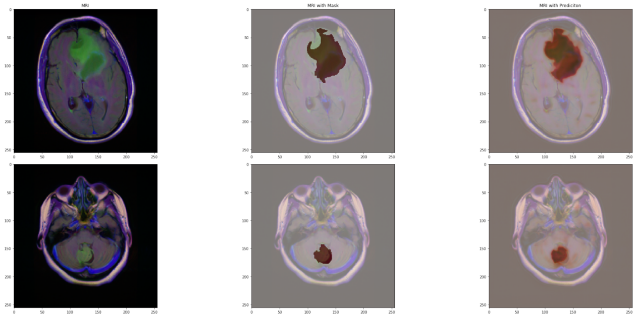


Figure 9: Predictions using Adam and BCE + Dice loss.