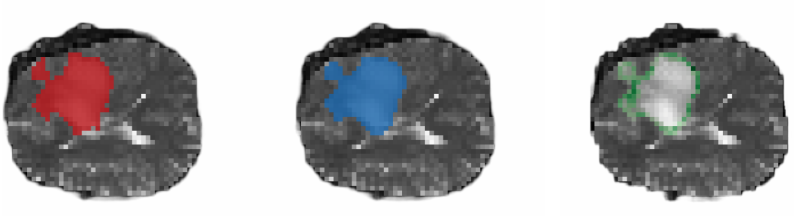
ResU-Net: Image segmentation of brain cancer tumors using a U-Net architecture with residual connections.

Deep Learning - methods and applications (5TF078)

Author
Theodor Jonsson thjo0148@student.umu.se



Department of Applied Physics and Electronics
Umeå University - Sweden
June 8, 2023

Abstract
The use of deep learning to automatically segment MRI scans for cancerous tumors is a useful tool in assisting radiologists in making an accurate diagnosis. This paper focuses on improving the segmentation of 2D MRI slices while providing an estimation of the uncertainty in the deep learning model's prediction. The proposed ResU-net is a deep U-Net-based architecture with internal residual components. This model achieved a Dice-score of 88.9% on the BraTS 2019 dataset.

Keywords
Deep Learning, Segmentation, Uncertainty estimation, Monte Carlo Dropout

[1] *Department of Applied Physics and Electronics, Umeå University, Sweden*
*__Corresponding author__: thjo0148@student.umu.se
__Supervisor__: tomas.nordstrom@umu.se[1]

Contents

1. Introduction

The task of segmenting images has been an important task within computer vision for several years. During this time it has been used in a variety of applications including; video editing, robotics, autonomous driving, and many more. This paper focuses on applying image segmentation to segment cancerous tumors in the brain from MRI scans. This technology has been used to assist radiologists in many hospitals across the world. Specifically, this paper discusses the use of a U-Net [1] with a ResNet-like structure [2]. The data was the BraTs 2019 dataset and were supplied via the course Convolutional Neural Networks with Applications in Medical Image Analysis and can be found in the Github repository containing the code of this project.

Using deep learning in the medical domain means that we must be able to explain the results of the model. If we can't measure any form of uncertainty in the model's predictions then we can't say if the prediction is a result of some statistical anomaly. Therefore this paper also discusses the uncertainty in the model using Monte Carlo dropout for uncertainty estimation to further improve the robustness of the model's predictions suitable for medical applications.

2. Theory

2.1 Evaluation - Dice score

To construct and train a model we must first determine a way to evaluate a model on a segmentation task. One common way of evaluation is to use the Sørensen–Dice coefficient or more commonly referred to as the *Dice score*. This measures the amount of overlap per class and divides it by the total number of occurrences per class or mathematically:

$$D = \frac{2\left|\hat{Y} \cap Y\right|}{\left|\hat{Y}\right| + |Y|}. \tag{1}$$

Where $\hat{Y}$ is the model's prediction per pixel and Y is the ground truth label. This metric has been used to determine the model's capability to segment the binary masks correctly.[3] To train the model we can then construct the Dice loss.

2.2 Loss function - Focal dice loss

The dice loss is used to optimize the dice score via

$$\ell_D = 1 - D \tag{2}$$

where D is the dice score as described in Eq. 1. This loss would optimize for the dice score but due to imbalances in the class distributions, this would probably result in only predictions of the background. Instead, we utilize a combination between Focal loss and dice loss. Focal loss is a way to penalize the prediction of the more common background class by adding a regularizing to the loss function in the form of

$$\ell_F = -\alpha\left((1-Y)\cdot(1-\hat{Y})^{\gamma}\log\left(1-\hat{Y}\right)+Y\cdot\hat{Y}^{\gamma}\log\hat{Y}\right). \quad (3)$$

Notice that if $\gamma = 0$ and $\alpha = 1$ then this would be the usual binary cross entropy loss.[4] The loss function during training was then:

$$\ell = w_F\cdot\ell_F + w_D\cdot\ell_D \quad (4)$$

Where w_F and w_D are relative importance hyper-parameters.

2.3 Uncertainty estimation

To estimate the uncertainty in the model and find where the model is uncertain in its predictions we implemented Monte Carlo Dropout. Monte Carlo Dropout is a technique to estimate the epistemic uncertainty, i.e. the uncertainty in the model. This is done by making predictions on the data over and over with dropout activated. Dropout is typically deactivated during inference to make the model utilize all activation maps but the Monte Carlo method uses the dropout layers in the model to induce a form of masking. Running inference several times on the same input data can be used to estimate the uncertainty of the model by examining the output. In this paper, the uncertainty is determined by the standard deviation of the segmentation map over the many iterations performed.

3. Method

The main parts of the model's development pipeline were to

1. Find an appropriate dataset.
2. Process the data such that it can be interpreted by the model.
3. Determine the hypothesis space and the loss function which together construct the model.
4. Train the model.
5. Evaluate the model.

The model proposed was developed using Tensorflow and Keras.

3.1 BraTS

The BraTS dataset is a dataset containing **Bra**in **T**umor **S**egmentations hence the abbreviation **BraTS**. The dataset is part of an annual challenge designed to improve the segmentation models used in radiology today.

The specific version used during this project is available through the project's Github in the form of NumPy slices of the 3D scans. The scans consist of 4 different scanning formats.

- T1-weighted visualizes the various tissues in the scan. A higher value implies higher fat content.
- T2-weighted visualizes the differences in water content per tissue. Fluids appear brighter than solids such as bone or such.
- FLAIR or Fluid-Attenuated Inversion Recovery suppresses the liquid material making it easier to see abnormalities in the scans.
- T1CE are T1-weighted images with increased contrast (Contrast-enhanced).

In total, there are 8000 scans for training 960 for validation and 960 for testing.[5][6][7][8][9]

3.1.1 Pre-processing

In order to make the inputs more model friendly we utilized pre-processing in the form of normalizing and zooming, i.e. centering the inputs and removing the redundant background. We also utilize all possible information about the image by stacking the different contrasts as a 4-dimensional image, one per scan format. In summary, the initial (256×256) contrasts that are available per format are; stacked, normalized, and cropped to create a $(64\times64\times4)$ input.

3.2 Model Architecture

The model architecture can be seen in the Appendix figure A3, or in the repository, and is a modification of the typical U-Net, see figure A1 for reference of the U-Net structure. These modifications are:

- Internal residual connections similar to a ResNet architecture.
- A deeper and larger network with larger minimum latent space representation.
- Spatial dropout regularization which drops activation maps at random.

3.2.1 Residual Blocks

The "residual_block" is one of the main additions to the base U-Net model. It is a typical ResNet block and its structure can be seen in figure 1.

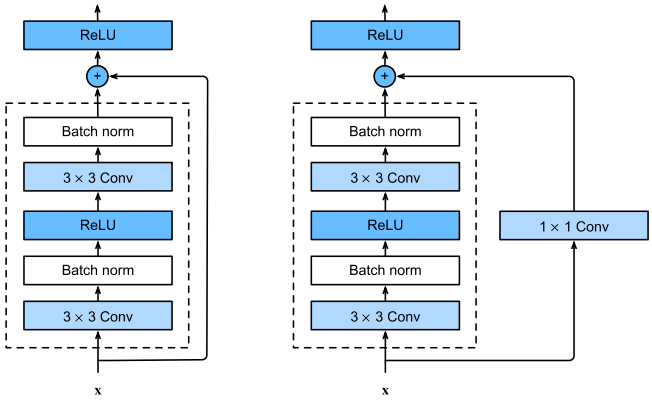


Figure 1. Visualization of the "residual_block". To the right is the structure of the residual block if it is the first in its depth cycle. To the left is all consecutive blocks after the initial residual block per depth.

The image above demonstrates how the internal skip-connections are constructed. After every down- or up-sampling layer there is a depth of residual blocks. The image demonstrates the two possibilities of how the blocks can be constructed, one if it is the initial block at the current depth (left) and the other (right) if it is one of the later blocks per depth. The right is denoted as "ResidualConvBlock" and the left "ResidualLinearBlock" in the repository and visualization of the image found in the Appendix figure A3 and here.

3.2.2 Encoder

The structure of the encoder was 5 layers with an increase of residual block per layer. To reduce the dimensionality of the activation maps produced between the sections of residual blocks, there is a max pooling layer to reduce the height and width per activation map by half in between each encoder layer. There are also spatial dropout layers in between each residual block.

Table 1. Encoder architecture. Each layer produces one output which is used as context in the decoder. The internal depth represents the number of residual blocks per layer.

Layer	Output Shape	Internal depth
Input	$(64 \times 64 \times 4)$	–
1	$(32 \times 32 \times 16)$	1
2	$(16 \times 16 \times 32)$	2
3	$(8 \times 8 \times 64)$	3
4	$(4 \times 4 \times 128)$	4
5	$(2 \times 2 \times 128)$	4

3.2.3 Decoder

The decoder has the task to reconstruct an image from the feature maps produced by the encoder. In a U-Net architecture, the decoder uses the intermediate feature maps through concatenation with the features which has been passed through the entire network, see figure A1 for reference. The decoder uses the results of the encoder to produce a segmentation map. The structure of the decoder can be found in table 2.

Table 2. Decoder architecture. Each layer produces one output which is used as context in the decoder. The internal depth represents the number of residual blocks.

Layer	Output Shape	Internal depth
Input	$(2 \times 2 \times 256)$	1
1	$(4 \times 4 \times 256)$	1
2	$(8 \times 8 \times 128)$	1
3	$(16 \times 16 \times 64)$	1
4	$(32 \times 32 \times 32)$	1
5	$(64 \times 64 \times 16)$	1
Output block	$(64 \times 64 \times 1)$	1

3.2.4 ResU-Net

The ResU-Net is a proposed model architecture that combines the encoder and decoder to the U-Net-like structure with components resembling those of a ResNet. The model has a total of $6,854,609$ trainable along $9,824$ static parameters. All model's layers are best seen in the repository found here but can also be seen in figure A3 in the appendix.

3.3 Training Process

The training process consisted of three parts; data augmentation, model optimization, and hold-off validation. The model was trained on an Nvidia 3060 Ti with a batch size of 64 set to train 40 epochs

3.3.1 Data Augmentation

To reduce overfitting and make the model generalize better we added data augmentation as a step in the pre-processing. The augmentation used and the probability of using them:

- Flipping the image in both the x and y direction. $p = 25\%$
- Masking parts of the input image. $p = 40\%$ with a maximum of 6 patches each covering at most 20% of the image.
- Adding noise to the input image. $p = 25\%$. The noise, ε, sampled as $\varepsilon \sim \mathcal{N}(\mu = 0.05, \sigma^2 = 0.1)$.

Note that each batch was given the same augmentation for parallel efficiency purposes. To see the results of the augmentation you can see some input samples in the Appendix, figure A4.

3.3.2 Model Optimization

The model was trained using Keras implementation of the Adam optimization algorithm [11] combined with Keras' early stopping, with stopping patience set to 5 epochs, and learning rate scheduler, reduction factor set to 0.5 and a patience of 2 epochs. Model check-pointing was also used during training to save the best model based on the loss of the validation set. The following hyperparameters were initially set when starting the model's training:

Table 3. Hyper-parameters used during training. These were what the parameters were initialized to but due to the learning rate scheduler, the learning rate decreases as the training process stagnates. As a result of the early stopping the number of epochs may not be reached.

Hyper-parameter	Value
Learning rate	0.01
Batch size	64
Number of epochs	40
Weight decay	0.0
Dropout rate (encoder)	0.1
Dropout rate (decoder)	0.1
α in Focal loss	0.25
γ in Focal loss	1.0
Relative importance of Focal loss (w_F)	0.8
Relative importance of Dice loss (w_D)	0.2

3.4 Model Evaluation

The model was evaluated both using the hold-off validation and a test set. The primary metric used was the dice score described in section 2.1.

4. Results

4.1 Model results

Following the described training process the model. The training was stopped after 19 epochs due to stagnation in the learning with a final training loss of $3.6 \cdot 10^{-2}$. The final learning rate was $6.25 \cdot 10^{-4}$. The total training time was roughly 16 minutes and was performed using full precision. The dice score throughout the entire training process can be seen in figure 2.

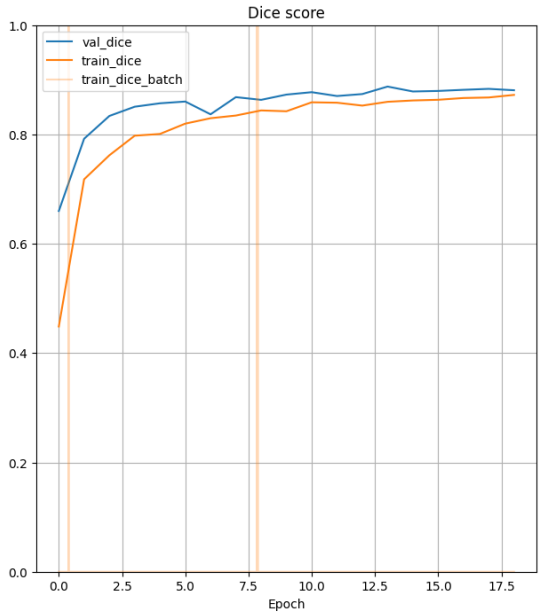


Figure 2. The dice score on the training and validation data during training.

After the training finished, the model was reset to the best checkpoint based on the validation loss. This is then the final model which was evaluated further. The performance of this model can be found in table 4.

Table 4. The model's performance using the best checkpoint based on the validation data. Note that data augmentation was not applied during the evaluation of the training set.

Dataset	Dice-score
Train	88.9%
Validation	88.9%
Test	78.5%

4.1.1 Samples

Note that none of these predictions were cherry-picked. Visualizing the prediction along with the uncertainty gives a lot of insight into how the model behaves.

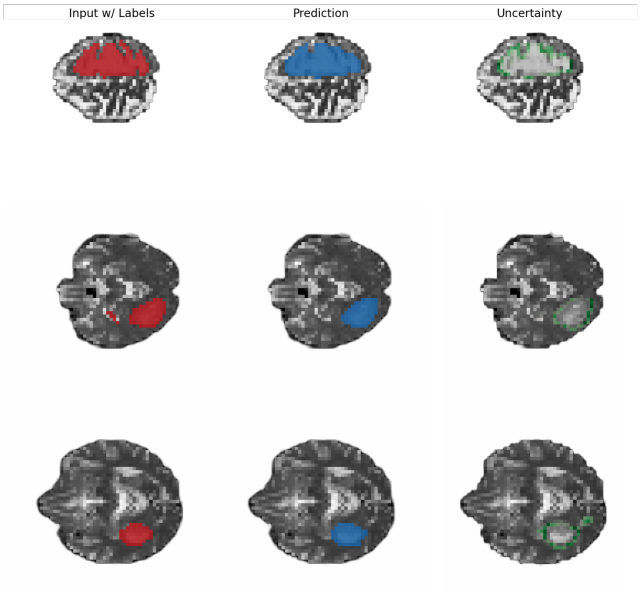


Figure 3. To the far left is the input from the test set with the ground truth highlighted in red. In the center are the model's predictions and to the right is the uncertainty. The uncertainty is normalized for better visualization, the greener the more uncertain.

Prediction samples of the validation set can be seen in appendix figure A2.

4.2 Data analysis

Because of the lacking test set performance, we examined the data present in the test set versus the data found in the train and validation sets since they perform similarly. We calculated the mean mask as well as the pixel-wise standard deviation in the segmentation mask throughout the data sets. These results can be found in figure 4.

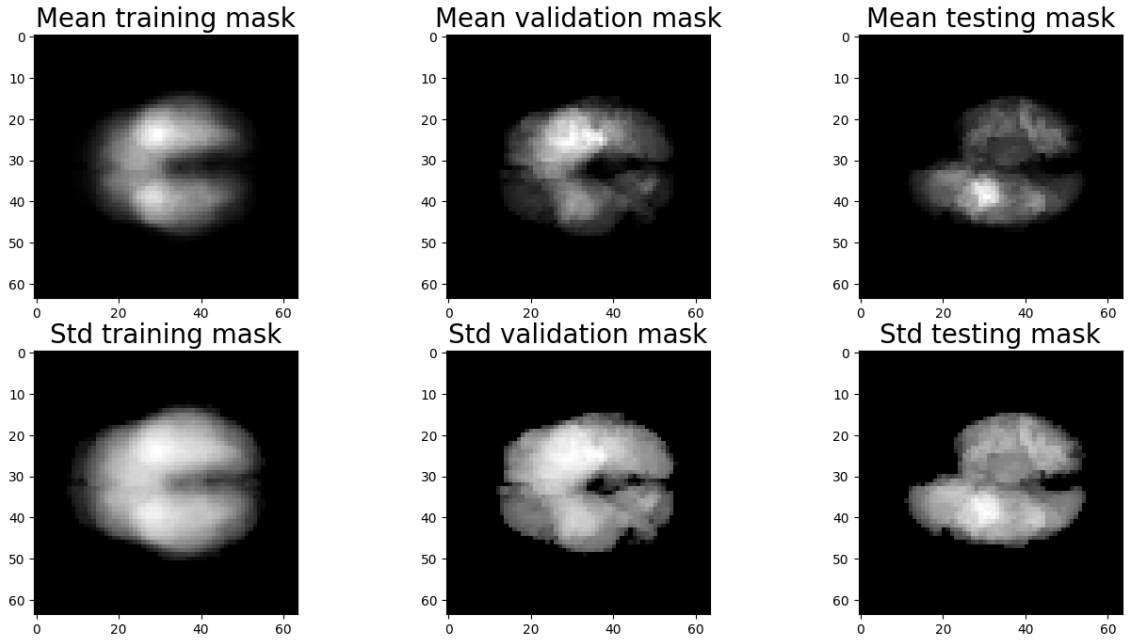


Figure 4. Comparision of the three dataset distributions in the form of their pixel-wise mean and standard deviation. Note that the pixels have been normalized.

5. Discussion & Conclusion

5.1 Performance discrepancies

ResU-Net seems to perform very well on the validation set and train set but lacks behind on the test set. This might be due to overfitting of the validation set. This is usually the case when performing hyper-parameter optimization (HPO) but since there hasn't been any HPO conducted it had to be investigated further. Another possibility is that the data distributions differ.

The distribution of the masks seemed to differ in their distribution based on figure 4. You can see that the validation mask distribution aligns very well with that of the training set, but the test distribution seems to be less prominent in the 2nd quadrant of the image. This might indicate some sort of corruption in the data or a simple bias in the dataset. Finally, to handle this we should collect some new data to check how the model performs or investigate further into how the test data is distributed to validate its similarities or dissimilarities to the training/validation sets.

5.2 Uncertainty

The model's uncertainty seems to lie at the edges of the tumors, through visual inspection of the uncertainty plots, see figure 3 and A2 in the appendix. This is very reasonable since there is usually no clear line between where the tumor ends and where it begins.

The uncertainty can further be used to give insight into how the model reasons about the image. For example; if there is a very large uncertainty spot but no clear detection of any tumor it might be worth investigating further. It will also make the predictions more descriptive as we can assign an uncertainty per pixel.

5.3 Performance comparison

Further investigation into how other papers have evaluated their models shows that most papers only present the result of the validation set evaluation. In comparison with the top 3 models presented according to papers with code, the ResU-Net seems to perform on par. The "Attention-Guided" was added as a reference for 2D-based models as the others are 3D-based.

Table 5. Comparison of current SOTA models on the validation data of the BraTS dataset using the whole tumor (WT). (*) used BraTS 2018.

Model	Dice Score	Params (m)
Extension of nnU-Net (3D) [12]	89.4%	41.2
Bag of tricks (3D) [12]	**90.4**%	-
Segtran (i3d) (3D) [12]	89.5%	166.7
Attention-Guided* (2D) [13]	89.5%	-
ResU-Net (2D)	88.9 %	6.86

It shall be noted that the performance in table 5 was on the validation set. The other models and the current SOTA are 3D-based models which utilize voxel representations of the MRI scans. This give the model more spatial information as it can utilize the information from many layers to make its prediction. Conclusively the proposed ResU-Net performs very well on the task of segmenting the cancerous tumors and even performs on par with the current SOTA when evaluated on the BraTS 2019 validation set.

References

[1] Olaf Ronneberger, Philipp Fischer, Thomas Brox. ”U-Net: Convolutional Networks for Biomedical Image Segmentation”. 2015. arXiv. https://arxiv.org/abs/1505.04597 (Retrieved 2023-06-3)

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun. ”Deep Residual Learning for Image Recognition”. 2015. arXiv. https://arxiv.org/abs/1512.03385 (Retrieved 2023-06-3)

[3] Wikipedia. ”Sørensen-Dice coefficient”. Available online: https://en.wikipedia.org/wiki/S%C3%B8rensen%E2%80%93Dice_coefficient (Retrieved 2023-06-04).

[4] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, Piotr Dollar, Facebook AI Research (FAIR). ”Focal Loss for Dense Object Detection”. 2017. Available online: https://arxiv.org/pdf/1708.02002.pdf (Retrieved 2023-06-04).

[5] B. H. Menze, A. Jakab, S. Bauer, J. Kalpathy-Cramer, K. Farahani, J. Kirby, et al. ”The Multimodal Brain Tumor Image Segmentation Benchmark (BRATS)”, IEEE Transactions on Medical Imaging 34(10), 1993-2024 (2015) DOI: 10.1109/TMI.2014.2377694 Available online: https://ieeexplore.ieee.org/document/6975210. (Retrieved 2023-06-05)

[6] S. Bakas, H. Akbari, A. Sotiras, M. Bilello, M. Rozycki, J.S. Kirby, et al., ”Advancing The Cancer Genome Atlas glioma MRI collections with expert segmentation labels and radiomic features”, Nature Scientific Data, 4:170117 (2017) DOI: 10.1038/sdata.2017.117. Available online: https://www.nature.com/articles/sdata2017117. (Retrieved 2023-06-05)

[7] S. Bakas, M. Reyes, A. Jakab, S. Bauer, M. Rempfler, A. Crimi, et al., ”Identifying the Best Machine Learning Algorithms for Brain Tumor Segmentation, Progression Assessment, and Overall Survival Prediction in the BRATS Challenge”, arXiv preprint arXiv:1811.02629 (2018) (Retrieved 2023-06-05)

[8] S. Bakas, H. Akbari, A. Sotiras, M. Bilello, M. Rozycki, J. Kirby, et al., ”Segmentation Labels and Radiomic Features for the Pre-operative Scans of the TCGA-GBM collection”, The Cancer Imaging Archive, 2017. DOI: 10.7937/K9/TCIA.2017.KLXWJJ1Q. (Retrieved 2023-06-05)

[9] S. Bakas, H. Akbari, A. Sotiras, M. Bilello, M. Rozycki, J. Kirby, et al.,(opens in a new window) ”Segmentation Labels and Radiomic Features for the Pre-operative Scans of the TCGA-LGG collection”, The Cancer Imaging Archive, 2017. DOI: 10.7937/K9/TCIA.2017.GJQ7R0EF. Available online: https://wiki.cancerimagingarchive.net/pages/viewpage.action?pageId=24282668. (Retrieved 2023-06-05)

[10] Case Western Reserve. ”Magnetic Resonance Imaging (MRI) of the Brain and Spine: Basics”. Available online: https://case.edu/med/neurology/NR/MRI%20Basics.htm (Retrieved 2023-06-04).

[11] Diederik P. Kingma, Jimmy Ba. ”Adam: A Method for Stochastic Optimization”. 2014. Available online: https://arxiv.org/pdf/1412.6980.pdf (Retrieved 2023-06-04).

[12] Shaohua Li, Xiuchao Sui, Xiangde Luo, Xinxing Xu, Yong Liu, Rick Goh ”Medical Image Segmentation Using Squeeze-and-Expansion Transformers”. 2021. arXiv. (Retrieved 2023-06-05). Available online: https://case.edu/med/neurology/NR/MRI%20Basics.htm

[13] MM. Noori, A. Bahri and K. Mohammadi, ”Attention-Guided Version of 2D UNet for Automatic Brain Tumor Segmentation”, 2019 9th International Conference on Computer and Knowledge Engineering (ICCKE), Mashhad, Iran, 2019, pp. 269-275, doi: 10.1109/ICCKE48569.2019.8964956. Available online: https://ieeexplore.ieee.org/document/8964956(Retrieved 2023-06-05)

A. Appendix

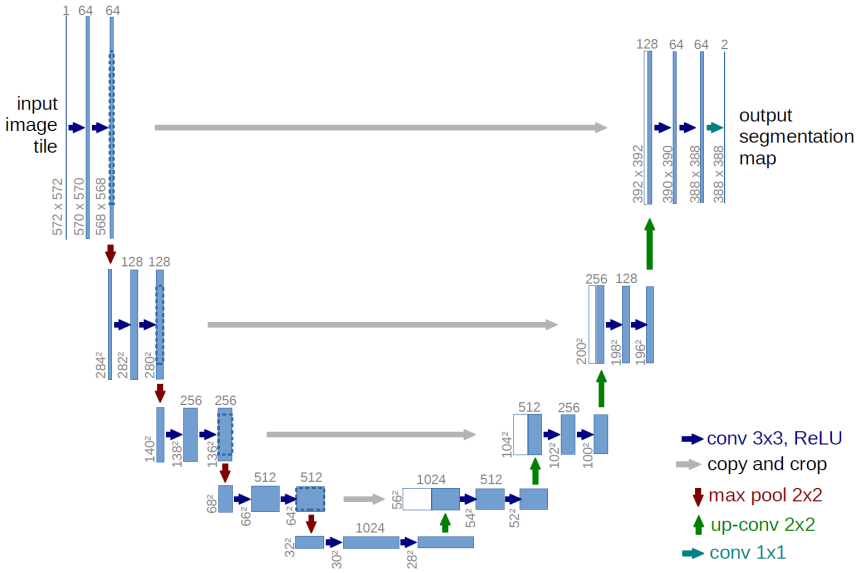


Figure A1. The original U-Net architecture. It is clearly visualized how the skip connections are used as context in the decoder to produce a better segmentation map.[1]

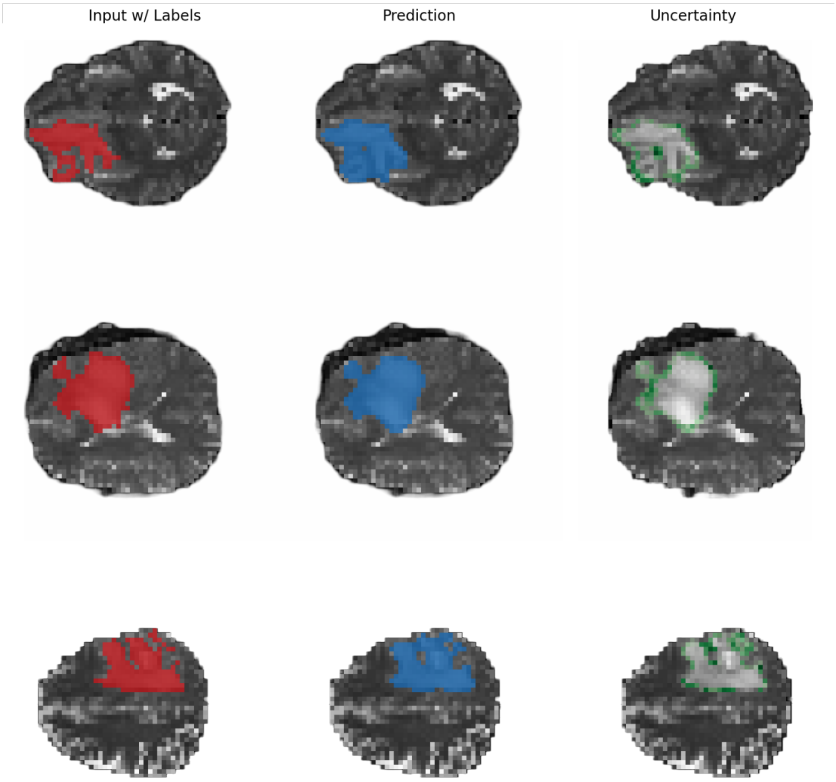


Figure A2. Model prediction on the validation set. The uncertainty is normalized for better visualization.

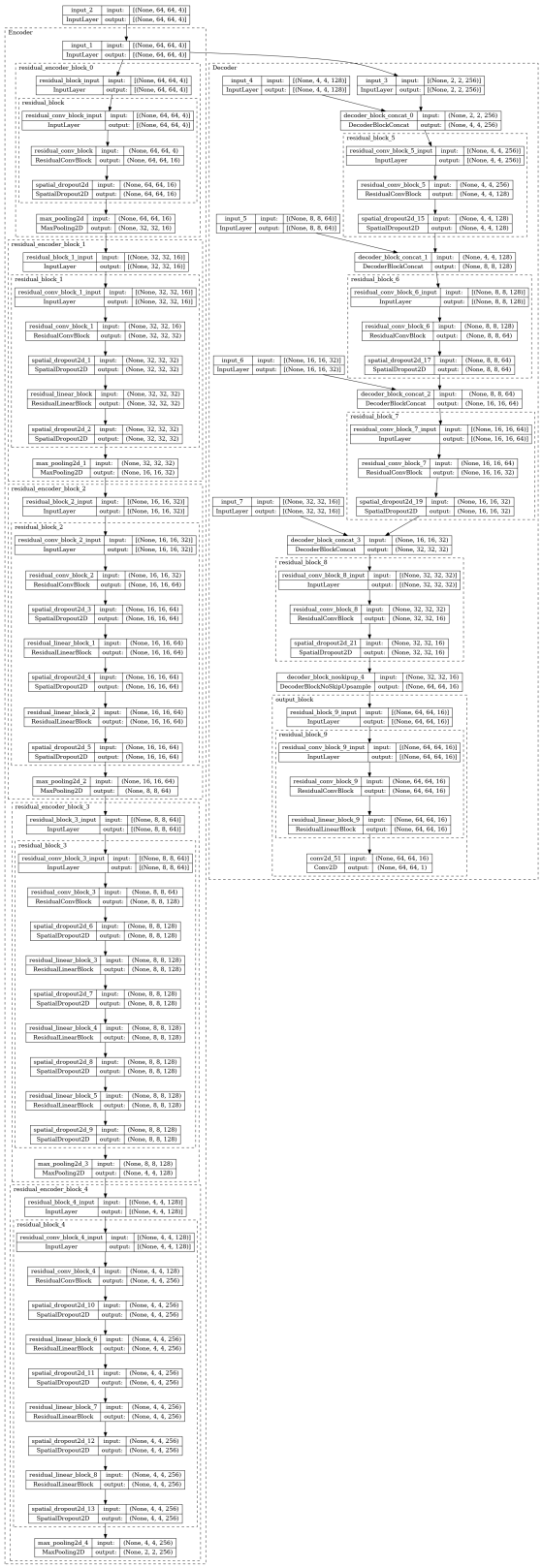


Figure A3. The U-Net model architecture with residual connection blocks. These blocks are standard ResNet blocks consisting of convolutions, activations, batch normalizations, and a highway skip connection. The activation function used was the standard ReLU function. This model had a total of $6,854,609$ trainable parameters. A better view of the model can be found in the repository here.

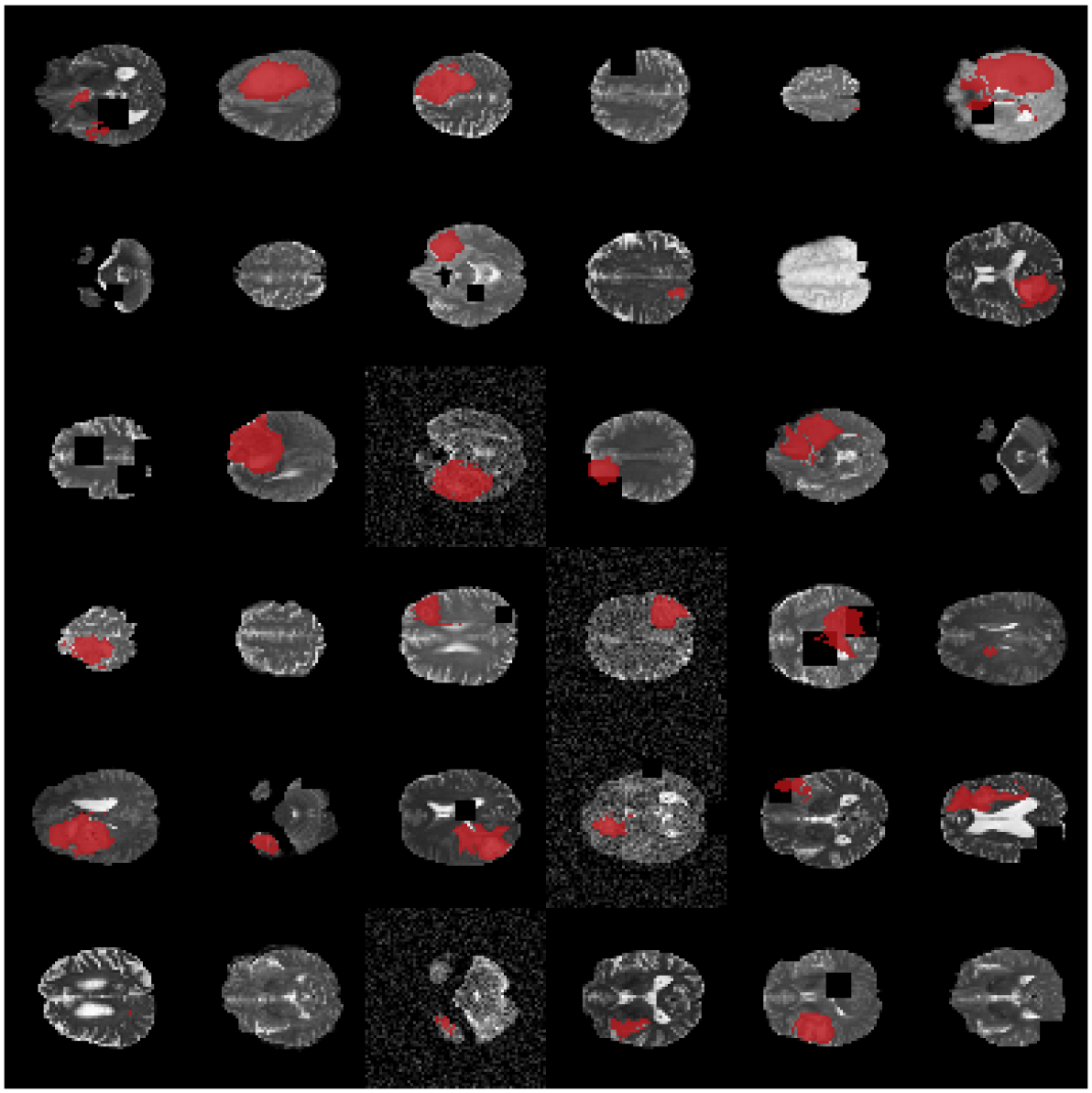


Figure A4. Input samples to the model after pre-processing with augmentations. The shown channel is the one containing the FLAIR contrast. The ground truth is highlighted in red but not a part of the input.