Final Project

CID 01252821

Introduction

This project focuses on the topic of Brain MRI segmentation with the help of Deep Learning techniques. More precisely, it explores the application of a U-Net (a convolutional neural network developed for biomedical image segmentation) on Brain MRIs. The aim is to train a model with the help of manually created fluid attenuated inversion recovery (FLAIR) abnormality segmentation masks,capable of detecting a lower-grade glioma (LGG - a type of brain tumor) in a 2D Brain MR image.

Medical Terminology

- **Glioma** is a type of tumor that starts in the glial cells (non-neuronal cells that surround, protect, and support the brain and spinal cord neuorns) of the brain or the spine. Gliomas are the most common type of brain tumor.
- **Low-grade gliomas (LGG)** are gliomas which grow slowly, often over many years, and can be followed without treatment unless they grow and cause symptoms. They correspond to Grade 1 and Grade 2 gliomas (based on WHO classification from Grade 1 to Grade 4, where less invasive and slow-growing tumors have a lower grade and are associated with more prolonged survival). Commonly, they are non-cancerous (benign) slow-growing tumors which, on rare occasions, can transform into aggressive (malign) tumors.
- **Magnetic resonance imaging (MRI)** is a medical imaging technique used in radiology to form pictures of the anatomy and the physiological processes of the body.
- An **MRI sequence** refers to a specific combination of radiofrequency pulses and gradients that result in a set of images with a particular appearance. Each sequence is designed to highlight certain types of tissues or abnormalities, based on how different tissues respond to magnetic fields and radio waves. A **multiparametric MRI** is a combination of two or more sequences, in our case **pre-contrast**, **FLAIR**, **post-contrast**:
  - **Pre-contrast** refers to sequences taken before the administration of a contrast agent. This initial image provides baseline images of the anatomy without any enhancement from contrast agents. It's valuable for assessing the natural state of tissues and identifying any abnormalities that might be inherently high or low in signal.
  - **Fluid Attenuated Inversion Recovery Magnetic Resonance Imaging (FLAIR)** is a specialized medical imaging technique that enhances the detection of brain and spinal cord abnormalities. It's designed to suppress the signal from cerebrospinal fluid (CSF), making lesions in the brain and spinal cord, such as multiple sclerosis plaques, more discernible. The result is an image where fluid appears dark, while pathological alterations in nearby tissue stand out with increased contrast. This unique enhancement ensures that abnormalities, often obscured by the bright fluid signal in other sequences, become more conspicuous and readily identifiable. Refer to Figure 1.
  - **Post-contrast** refers to the initial images being retaken after the administration of a contrast agent, typically a gadolinium-based compound. The contrast agent is injected into the patient's bloodstream and helps to enhance certain tissues and blood vessels, making it easier to identify areas of inflammation, tumor, infection, or other pathologies. The comparison between pre-contrast and post-contrast images can provide critical information about the nature of any abnormalities,

such as whether a tumor is present and how aggressively it's behaving (based on how much it 'lights up' with contrast).

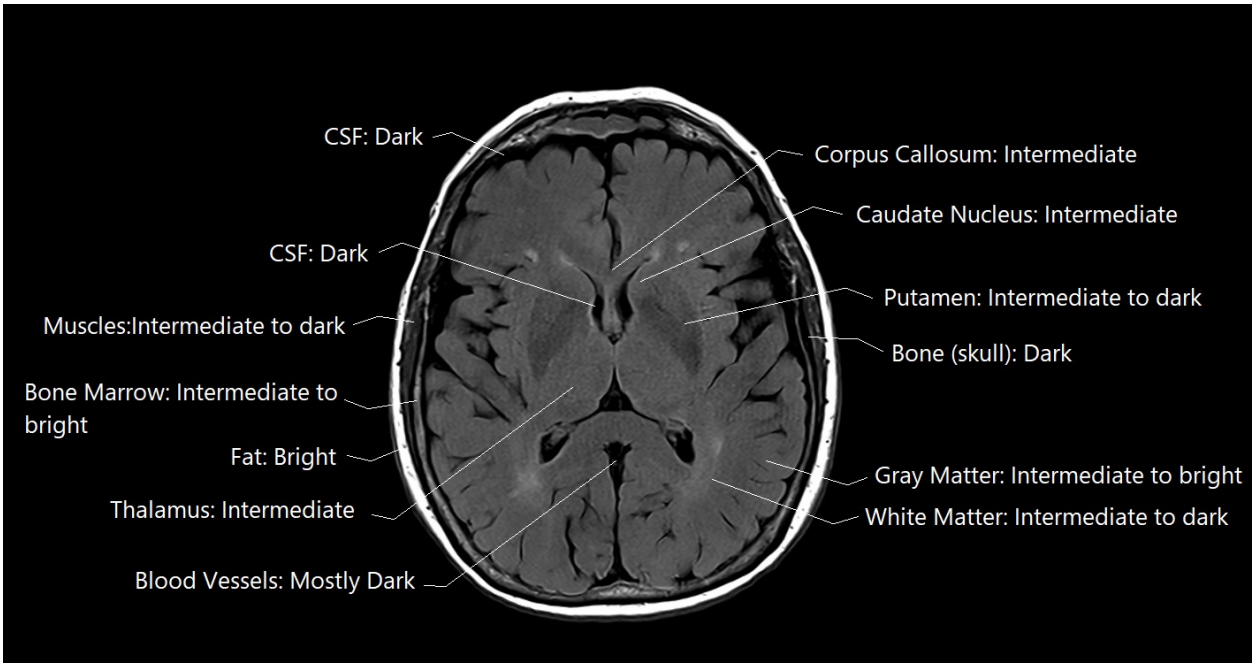

Figure 1: FLAIR MRI of a Brain (source)

Data

The image data for this project is taken from Kaggle. The LGG Segmentation Dataset is used in "Mateusz Buda, AshirbaniSaha, Maciej A. Mazurowski"Association of genomic subtypes of lower-grade gliomas with shape features automatically extracted by a deep learning algorithm." Computers in Biology and Medicine, 2019." and "Maciej A. Mazurowski, Kal Clark, Nicholas M. Czarnek, Parisa Shamsesfandabadi, Katherine B. Peters, Ashirbani Saha"Radiogenomics of lower-grade glioma: algorithmically-assessed tumor shape is associated with tumor genomic subtypes and patient outcomes in a multi-institutional study with The Cancer Genome Atlas data." Journal of Neuro-Oncology, 2017.".

The dataset contains preoperative brain MRIs together with manual FLAIR abnormality segmentation masks showing the presence (or absence) of LGG. The images were obtained from The Cancer Imaging Archive (TCIA). They correspond to 110 patients, with number of slices varying among patients from 20 to 88, included in The Cancer Genome Atlas (TCGA) lower-grade glioma collection with at least FLAIR sequence and genomic cluster data available. The

All images are provided in `.tif` format with 3 channels per image corresponding to the following 3 MRI sequences in the given order: pre-contrast, FLAIR, post-contrast. For 101 cases, the 3 sequences are available, for 9 cases, post-contrast sequence is missing and for 6 cases, pre-contrast sequence is missing. The missing sequences are replaced with FLAIR sequence to make all images 3-channel. Masks are binary, 1-channel images, and they segment FLAIR abnormality present in the FLAIR sequence, and are available for all cases.

The dataset has 110 folders named after case ID that contains information about source institution. Each folder contains MRI images with the naming convention `TCGA_<institution>_<patient-id>_<slice-number>.tif`. Corresponding masks have a `_mask` suffix.

Two main categories are present: LGG present, where the FLAIR abnormality corresponding to the presence of LGG is in white in the mask (see Figure 2), and LGG not present, where these is no LGG present and

hence the mask is all black (see Figure 3).

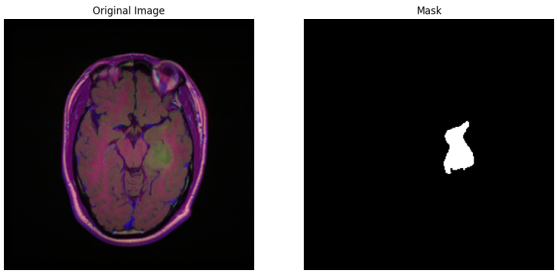

Figure 2: Brain MRI (left) with LGG present, shown in the mask in white (right)

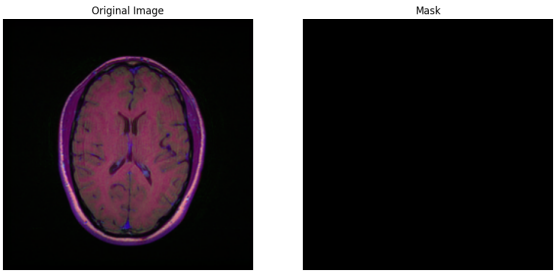

Figure 3: Brain MRI (left) with LGG not present, mask is black (right)

Problem Statement

In the past years there has been a new research direction in cancer called "radiogenomics," which investigates the relationship between tumor genomic characteristics and medical imaging. Correlations between tumor shape features from MRI images and genomic subtypes of tumors have been found, however, the initial step of extracting these features involves manually segmenting MRIs, which is costly, time-consuming, and prone to high variability between different annotators. With the help of Deep Learning models can be developed such that they yield high quality segmentation of LGG in brain MRI and would potentially allow for automatization of the process of tumor genomic subtype identification through imaging that is fast, inexpensive, and free of inter-reader variability. Thus, the analysis in this project attempts to explore the capabilities of Deep Learning for performing tumor segmentation visible on 2D brain MRI slices, more precisely detecting and uncovering the shape of a present LGG.

Method

The typical use of CNNs is on classification tasks, where the output to an image is a single class label. However, for example in the case of brain tumor segmentation where the output is supposed to be a shape of a tumor, a class label is supposed to be assigned to each pixel. Thus, in the analysis U-Net (paper), a type of convolutional neural network developed for biomedical image segmentation, is being explored. U-Net is known for its efficiency in learning from a limited amount of images and for its ability to accurately segment different parts of the images, making it highly popular in medical image analysis.

Network architecture

The U-Net architecture consists of a contracting path (left side) and an expansive path (right side). The contracting path has a typical CNN architecure consisting of the repeated application of two 3x3 unpadded convolutions, each followed by a ReLU and a 2x2 max pooling with stride 2 for downsampling. At each downsampling step the number of feature channels are doubled.

Every step in the expansive path consists of an upsampling of the feature map followed by a 2x2 convolution that halves the number of feature channels, a concatenation with the correspondingly cropped feature map from the contracting path, and two 3x3 convolutions, each followed by a ReLU. The cropping is necessary due to the loss of border pixels in every convolution.

At the final layer a 1x1 convolution is used to map each 64-component feature vector to the desired number of classes, in our case 1. In total the network has 23 convolutional layers.

The architecture adjusted for the Brain MRI data used in the project is illustrated in Figure 4.

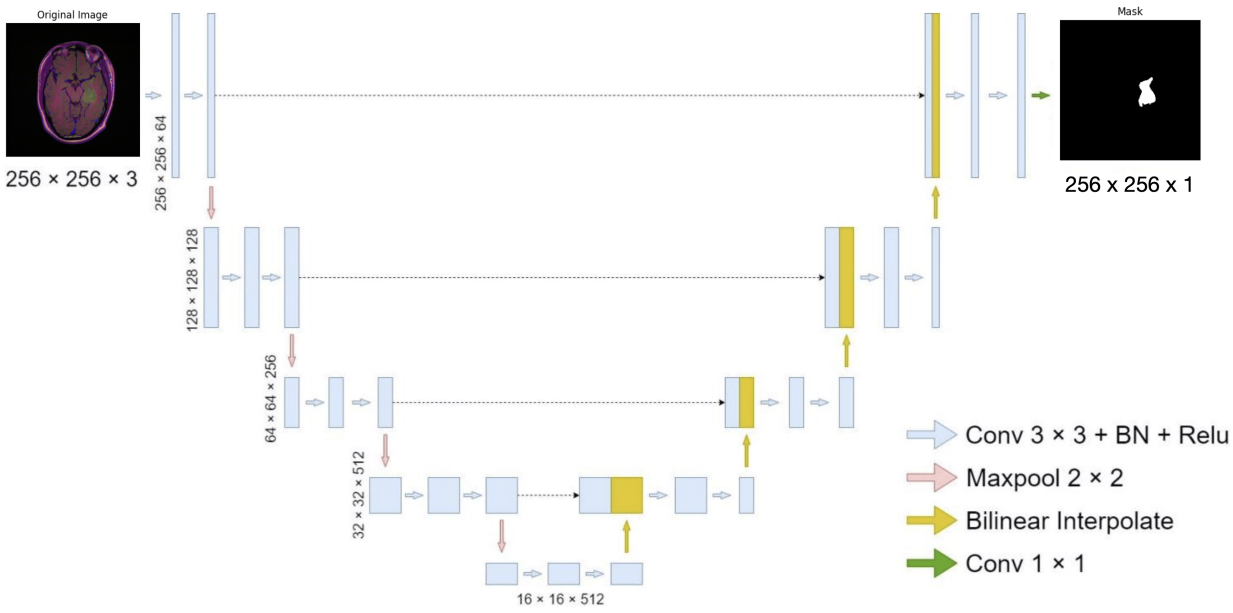

Figure 4: U-Net like architecture, BN in the figure represents batch normalization. (source)

Implementation

The implementation used is taken from the following GitHub repository. In the current project repository it is located in the file unet.py.

Training, Validation, and Testing of the Model (on Full Kaggle Data)

The aim of the project is to train a U-Net model (as illustrated in Figure 4) such that it takes as an input a 2D 3-channel Brain MRI and produces an output in the form of a 1-channel mask, which contains the shape of the tumor predicted by the model:

$$256 \times 256 \times 3 \text{ Brain MRI} \xrightarrow{\text{U-Net}} 256 \times 256 \times 1 \text{ mask}$$

The model is trained and validated on test and validation datasets for 94 epochs, during which training loss, training IoU (Intesrection over Union), validation loss, and validation IoU are computed and saved for each epoch. After that the epoch with highest validation IoU is chosen and then the model is tested on test dataset with model parameters from the chosen epoch. For the purpose, the data has been initially divided in three datasets: train, validation, and test. The train, validation, and test datasets are created so that the images of each unique patient are all in one of the 3 datsets, so that data leakage is not introduced. This is done in the file `train_model.py`. When executed in a Linux environment with the set seeds, we obtain the following characteristics of the datsets:

- Train dataset with 2769 image pairs of the type (Brain MRI, mask) corresponding to 79 unique patients.
- Validation dataset with 756 image pairs corresponding to 20 unique patients.
- Test dataset with 404 image pairs corresponding to 11 unique patients.

The model is trained, validated and tested in Google Colaboratory with GPU runtime type: `NVIDIA A100-SXM4-40GB` with `CUDA Version: 12.2`. Hence, seeds are also set for `cuda`, and `torch.backends.cudnn.deterministic` is set to `True`.

The training of the model with the full Kaggle dataset with approximately 1 GB of data, and for 94 epochs takes approximately 40 minutes: notebook.

IoU (Intesrection over Union)

As mentioned above, the metric **Intersection over Union (IoU)** is computed for evaluation of the model. It is used in the fields of computer vision and image processing, particularly in object detection and segmentation tasks. It is a measure of the overlap between two areas, the area of overlap between a predicted object and its ground truth. The formula for IoU is

$$\text{IoU} = \frac{\text{Area of Intersection}}{\text{Area of Union}},$$

where **intersection** is the area that is common between the predicted tumor and the ground truth (the area where both predictions and reality agree that the object is located), and **union** is the total area covered by both the predicted boundary and the ground truth, including the intersection area.

The value of IoU ranges from 0 to 1, where:

- 0 indicates no overlap between the predicted boundary and the ground truth.
- 1 indicates perfect overlap, meaning the predicted boundary perfectly aligns with the ground truth.

In practice, a higher IoU score indicates better accuracy of the segmentation model, as it shows a greater degree of overlap between what the model predicts and what is actually present in the image.

Interpretation and Reflection on Output

The results presented here refer to the network fine-tuned for 94 epochs and the full 1 GB dataset. In Figure 5 the training and validation loss can be observed with respect to the epoch, and in Figure 6 the training and validation IoU metric can be observed with respect to the epoch.

Figure 5 illustrates a common and ideal scenario where both the training and validation loss decrease over time, indicating that the model is learning. The training loss has a smooth decline, suggesting stable learning, while the validation loss exhibits some fluctuations but overall trends downward. However, the general downward trend is a positive sign that the model is improving its generalization over time.

In Figure 6, we see that the training IoU is consistently higher than the validation IoU, which suggests that the model is fitting well to the training data. However, the gap between training and validation IoU indicates that the model might be overfitting since it is performing significantly better on the training data compared to the validation data. It's also worth noting that the training IoU continues to rise, which means the model is continuing to learn from the training data. In contrast, the validation IoU appears to plateau, which might suggest that the model has reached its generalization limit given the current architecture and dataset.

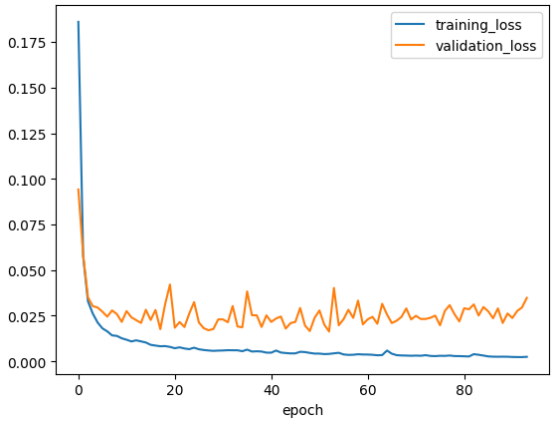

Figure 5: Training and validation loss vs time

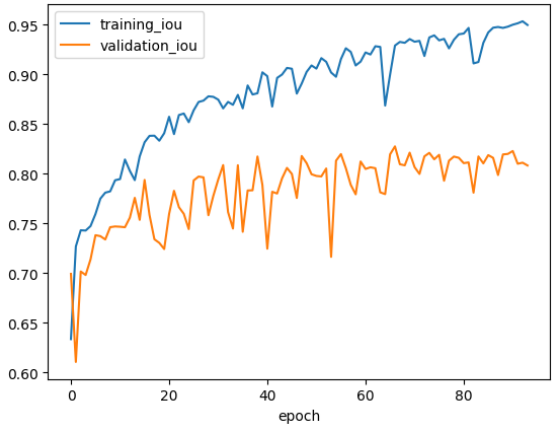

Figure 6: Training and validation IoU vs time

When the model is evaluated on the test set with parameters from the epoch with highest validation IoU (epoch 66), the IoU is approximately 0.85 which is quite high, especially given the complex nature of the dataset, and indicates that the model's predictions overlap with the ground truth by a substantial amount.

Moreover, comparing these test metrics with the training and validation metrics suggests the test IoU is in line with the IoU trends seen during training and validation, implying that the model's performance is consistent across different datasets. The high test IoU in comparison to the validation IoU suggests that the model generalizes well to new data. While there's a gap between the training and validation IoU, the fact that the test IoU is close to the validation IoU suggests that any potential overfitting is not severely impacting the model's performance on new data. In conclusion, the test results support the effectiveness of the model.

Some key insights on the predictions from the test dataset:

- The model performs generally well on tumors with smooth boundaries

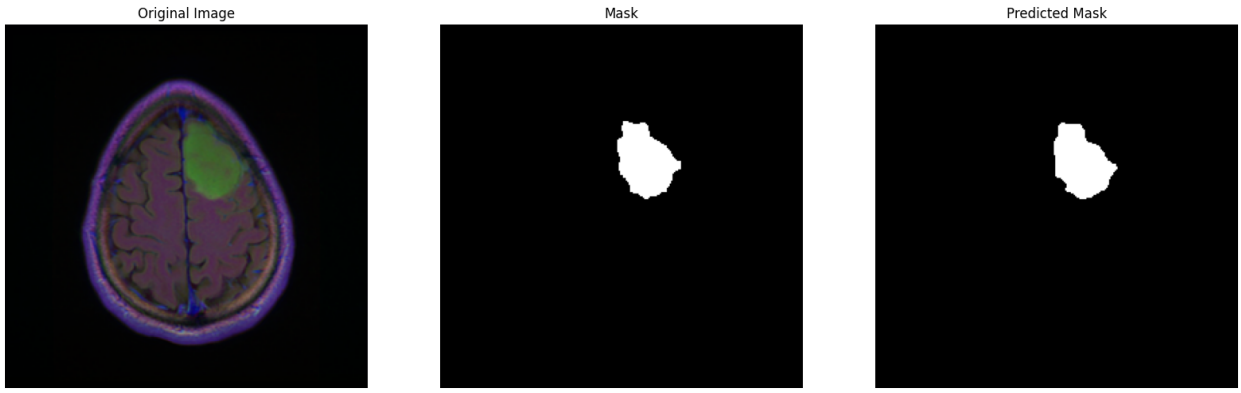

Figure 7: Good prediction of a tumor with smooth boundaries

- But it seems to fail at detecting smaller tumors

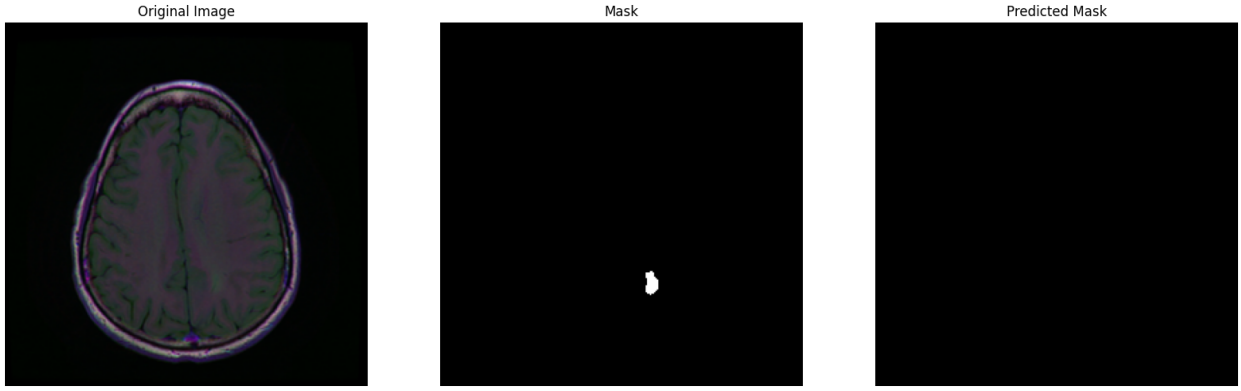

Figure 8: Missed detection of a small tumor

- Also, the model tends to miss one part of the tumor if it consists of two parts

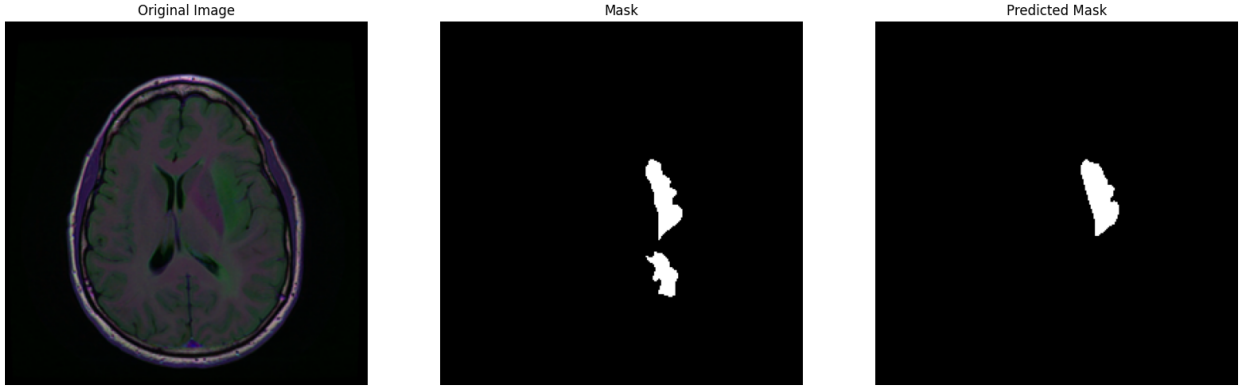

Figure 9: Missed part of a two-part tumor

- For some bigger tumors with more complex structure of the boundary the model has diffuclties in capturing the size of the tumor as well as the complexities of the boundaries

Figure 10: Complex boundary structure not fully captured

- But at the same time for some (moderately) complex cases the model manages to detect the boundary more closely to the original

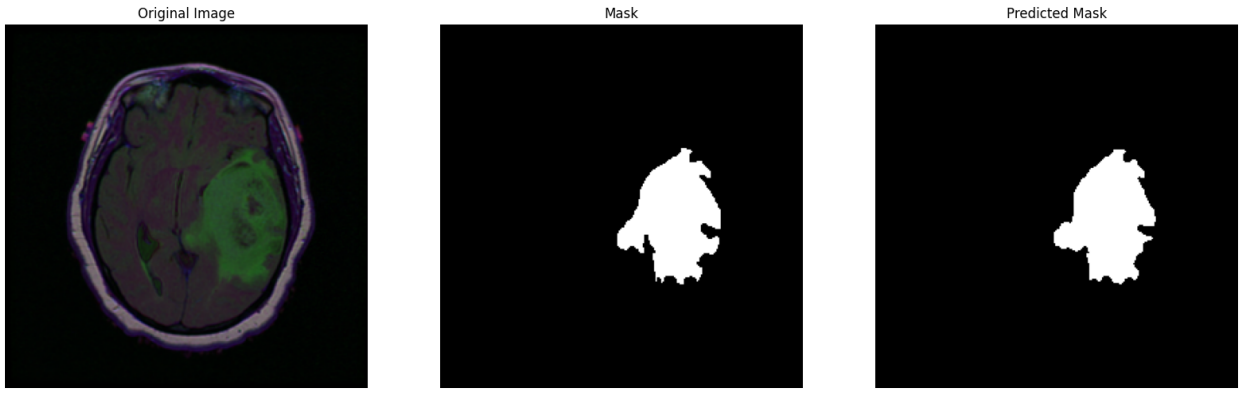

Figure 11: Complex boundary structure moderately captured

- For cases with no tumors the model always performs as expected and does not introduce false postive outputs

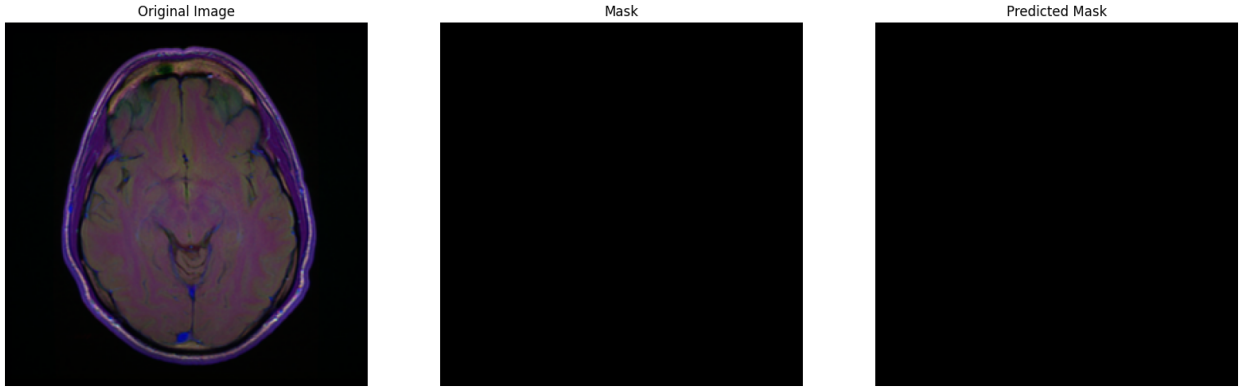

Figure 12: No tumor detected

Small Subset Results

The training of the model with the subset dataset from the GitHub project repository with approximately 100 MB of data, and for 94 epochs takes approximately 6 minuts: notebook.

If we take a look at the training and validation loss (Figure 13) and training and validation IoU (Figure 14) charts, it can be seen that the training and validation loss converge closely after an initial drop, which is a good sign indicating that the model is not overfitting significantly, and the loss values stabilize relatively early during training, suggesting the model quickly exhausts the learning potential from the smaller dataset. The training and validation IoU values are closer together than in the previous full dataset results, which could indicate less overfitting, both the training and validation IoU exhibit more noise, likely due to the smaller dataset size, which can lead to higher variance in the IoU from epoch to epoch, and finally the validation IoU, while volatile, does not show a clear increasing or decreasing trend, which suggests the model may have difficulties in learning further from the reduced amount of data.

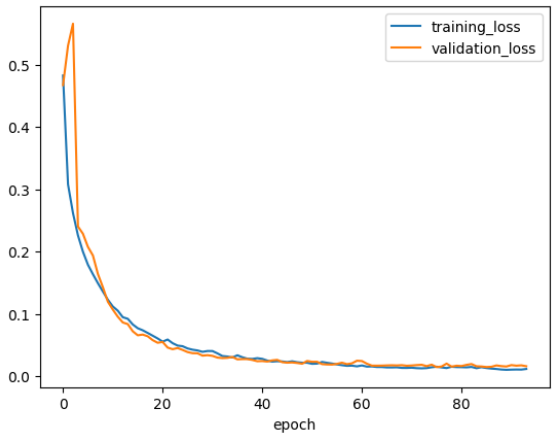

Figure 13: Training and validation loss vs time

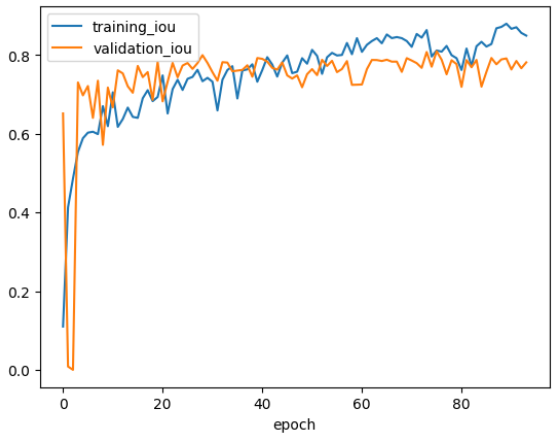

Figure 14: Training and validation IoU vs time

The test IoU is approximately 0.62 which is significantly lower than the IoU obtained from the full dataset, indicating that the model's predictions are less accurate when trained on the smaller dataset. This is expected because the model is trained on less data, and hence has less information to learn the underlying patterns. Moreover, the dataset is imbalanced and has more no tumor data than data with tumors, as well as the fact that the subsets now are

- Train dataset with 178 image pairs of the type (Brain MRI, mask) corresponding to 8 unique patients.
- Validation dataset with 45 image pairs corresponding to 2 unique patients.
- Test dataset with 113 image pairs corresponding to 2 unique patients.

showing there are very few patients in each dataset, implying it would be hard to train a model which can generalize from less than 10 unique patients to nearly 100.

Overall, the results from the reduced dataset indicate that while the model is not overfitting, its ability to generalize has been compromised. This suggests that the size of the dataset is an important factor in this model's ability to learn and generalize. The training seems to stabilize quickly, which could mean that either the model is not complex enough to capture more intricate patterns in the data or that additional data is required to further improve the model's performance.

Some key insights on the predictions from the test dataset:

- The model fails to capture the majority of the tumors no matter their size

Figure 15: Big tumor not detected

- A few false positive cases are observed, although the predicted tumor is fairly small

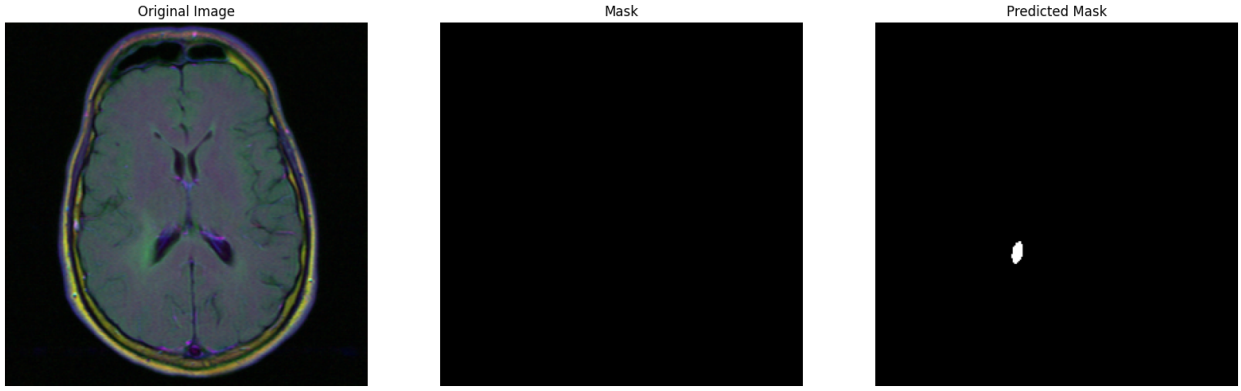

Figure 16: No tumor detected

- There are a few cases where the model manages to partially capture the presence of a tumor

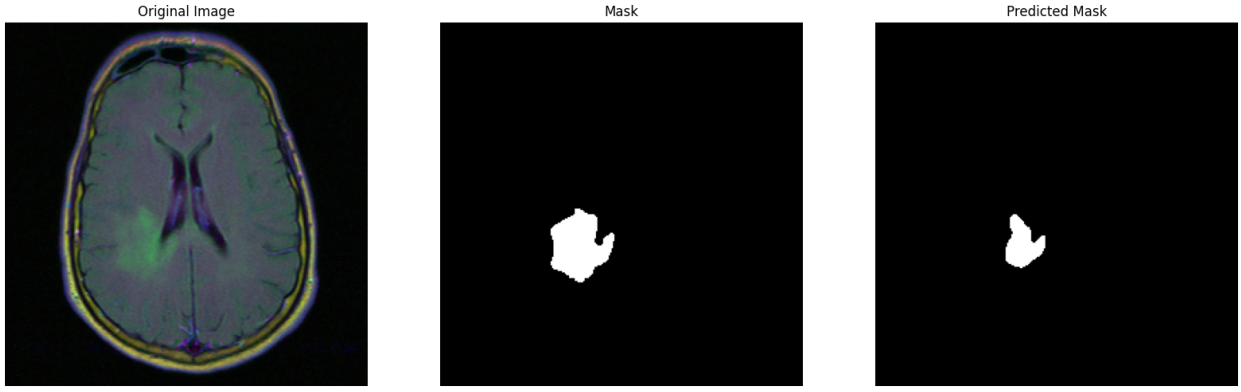

Figure 17: No tumor detected

Conclusion

In this project, we trained and evaluated a machine learning model for object detection, U-Net, employing both a full dataset and a significantly reduced subset of that data, which constituted around 10% of the original size. The analysis of training and validation metrics, along with the final test results, has provided us with insightful findings on the model's performance and generalization capabilities.

When trained on the full dataset, the model achieved commendable performance, with a high Intersection over Union (IoU) and low loss metrics on the test data. This affirmed the model's proficiency in accurately detecting and localizing objects, as indicated by a test IoU of approximately 0.85 and a test loss near 0.007. The consistent improvement in training IoU and a simultaneous decrease in loss over epochs underscored the model's capability to learn effectively and generalize well to unseen data.

Conversely, the reduced dataset presented a different scenario. Despite the training and validation metrics converging and displaying less discrepancy, the overall model performance on the test data was notably diminished. With a test IoU of approximately 0.62 and a test loss of approximately 0.043, it became evident that the decrease in dataset size had a substantial impact on the model's ability to generalize. The increased noise in the IoU and loss graphs over the epochs further suggested that the model faced challenges in extracting and learning robust features from the limited data.

In conclusion, this study underscores the balance between model complexity, dataset size, and generalization. Suggestions for future improvements of the model are scuh that images can be cropped excluding the high

amount of useless background pixels, as well as working on segmenting and removing the skull and fat parts of the brain MRIs. Furthermore, the model can bu tuned to penalize false negatives more than it penalizes the false positives.

Repository and Realted Technical Details

GitHub Repository

The code of the project is stored in the following public GitHub repository.

The folders:

- `lgg-mri-segmentation/kaggle_3m` contains a subset of the data of size 100 MB corresponding to 12 patients in total.
- `models` is kept empty and is for saving checkpoints of the model during the training procedure.

The files:

- `unet.py` contains the implementation of the U-Net model
- `constants.py` contains the constants shared between the files: model parameters and image transformations
- `utils.py` stores multiple functions used in multiple files
- `train_model.py` performs the creation of train, validation, and test datasets, the training and validation of the model while saving checkpoints for each epoch in the `models` folder
- `process_models.py` takes the checkpoints saved in the `models` folder, processses them and produces a `results.csv` file containing information about each epoch such as training loss, training iou, validation loss, validation iou
- `test_model.py` selects the model parameters corresponding to the epoch with highest validation IoU and evaluates the model with them on the test dataset

The notebooks:

- `train_test_100mb.ipynb` clones the project GitHub repository, executes `train_model.py`, `process_models.py`, `test_model.py`, and visualises the results and predictions for the 100 MB subset of the data. No credentials are needed for executing the notebook.
- `train_test_1gb.ipynb` downloads the full dataset from Kaggle, clones the project GitHub repository, executes `train_model.py`, `process_models.py`, `test_model.py`, and visualises the results and predictions for the whole 1 GB dataset

Environment

A list of the needed libraries with their versions:

```
Package                         Version
------------------------------- --------------------
matplotlib                      3.7.1
numpy                           1.23.5
opencv-python                   4.8.0.76
pandas                          1.5.3
Pillow                          9.4.0
scikit-learn                    1.2.2
torch                           2.1.0+cu121
torchaudio                      2.1.0+cu121
tqdm                            4.66.1
```