BRAIN TUMOR CLASSIFICATION USING CNNS: PROJECT PROGRESS REPORT

Marco Mastrangelo
Student# 1009133518
marco.mastrangelo@mail.utoronto.ca

Hussein Ismail
Student# 1008747910
h.ismail@mail.utoronto.ca

Devraj Solanki
Student# 1009065707
devraj.solanki@mail.utoronto.ca

Damilola Aina
Student# 1008932292
dami.aina@mail.utoronto.ca

Total Pages: 9

1 PROJECT DESCRIPTION

Brain tumors are a potentially life-threatening medical condition. The 5-year relative survival rates for certain malignant brain tumors can be as low as 6% (Ostrom et al., 2019). Therefore, early detection and classification of brain tumors is crucial for improving patient outcomes. Currently, medical professionals visually inspect medical resonance imaging (MRI) scans to identify brain tumors. However, MRI scans produce many images that represent layers in the brain, making it difficult to manually identify the existence of brain tumors (Ullah et al., 2023). Applying deep learning models, specifically CNNs (which exhibit high proficiency in object detection and classification), can result in faster processing of MRI images while eliminating the risk of human error. This project aims to produce a CNN that can accurately detect and classify 3 different types of brain tumors: glioma, meningioma, and pituitary, as well as classify inputs with no tumor present, given an image of an MRI scan of a patient's brain.

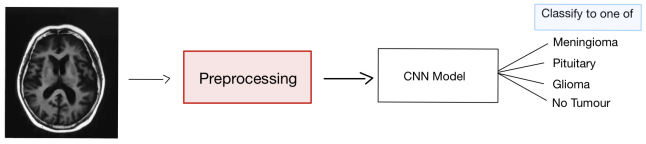

Figure 1: Diagram of Project Model. Image: Hussein Ismail

2 INDIVIDUAL CONTRIBUTIONS AND RESPONSIBILITIES

The following section outlines how the group has been operating until now, and how the group will continue to operate. This includes information about how the group has managed work distribution and deadlines. A table containing the notable contributions of each group member is also present in this section. Finally, future plans and strategies are discussed to ensure redundancies are in place in the case of unforeseen circumstances.

2.1 PROJECT MANAGEMENT

The team is currently using a variety of project management software to coordinate project work. Instant messaging software such as Instagram and Discord, is used among team members to communicate with one another. Additionally, GitHub is being used to share data files and code with team

members and to track updates and changes to code during the project. Lastly, a team Google Calendar is being used to track internal deadlines and record the teams' progress. For major deliverables such as this progress report, the team starts work at least one week in advance, and meets on Google Meet for approximately 15-30 minutes each day, up until the due date. This ensures that everyone is always keeping up with work, and teammates are aware of the progress of each teammate and any situation they may be in. The team has also adopted an internal deadline for report completion – 24 hours before the actual due date. This allows for 24 hours of minor editing and adjustments and avoids stressful situations in the final hours.

2.2 Current Contributions and Responsibilities

The responsibilities of each team member were laid out in the teams' project schedule, created during the project proposal. The project schedule broke down the tasks and the respective deadlines assigned to each team member.

For the project progress report deadline, the team planned to perform data preprocessing, build a baseline model, split and balance the dataset, and build the primary model. Table 1 below describes what each team member has accomplished until this deadline.

Table 1: Individual responsibilities and contributions of each team member

Team Member	**Responsibility**	**Contributions (to date)**
Devraj Solanki	- Perform data preprocessing - Split and balance dataset	*Data preprocessing*: Merged multiple datasets into 1 data set. Resized and normalized images. *Data Splitting*: Split data into validation and training dataset.
Hussein Ismail	- Perform data preprocessing - Split and balance dataset	*Data preprocessing*: Added noise, rotation and color intensity augmentations to training images. *Data Splitting*: Created data loaders for validation and training data.
Damilola Aina	- Select and build baseline model	*Building baseline model*: modified a preexisting CNN model, created train and test functions, chose optimizer and loss function, visualized model results.
Marco Mastrangelo	- Select and build primary model	*Building primary model*: Created an augmented *ResNet-18* model, chose optimizer and loss function, tuned hyperparameter settings, created test and train functions, visualized model results.

2.3 Future Project Plan

Moving forward, future project tasks and deadlines will be distributed according to the following Gantt chart, seen below in Figure 2. The project timeline will be split into 1-week segments to create checkpoints that ensure the team is on track to complete the project on time. Names will be color-coded to signify which team members are assigned to each task.

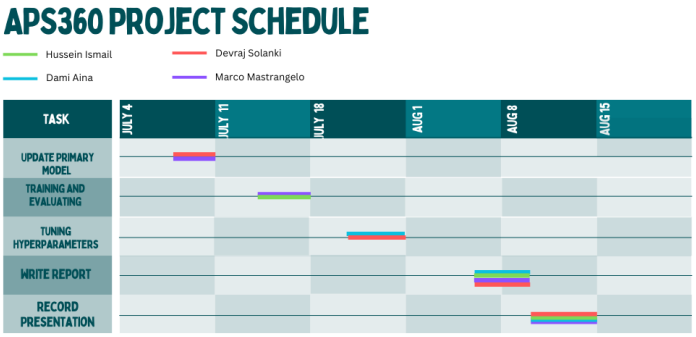

Figure 2: Future Project Schedule. Image: Devraj Solanki

For important tasks, such as updating the primary model; redundancies will be put in place to address potential setbacks and delays to the task completion process.

Weekly team meetings will be held on Google Meet, to update the team on the progress of each team member and discuss any questions or difficulties team members may have with their tasks. Therefore, in the event that a team member is unsure of how to complete a certain task, the rest of the team will be informed enough to intervene and help complete the task.

3 Notable Progress and Contribution

The following section describes the progress made by the group up to this point. As of now, all data has been collected and processed, a baseline model has been implemented, fully trained, and tested, and the group has started to experiment with deeper CNN models such as *ResNet-18*.

3.1 Data Processing

The team collected two datasets with identical labels (no tumor, glioma tumor, meningioma tumor, and pituitary tumor) from Kaggle (Nickparvar, 2021), (Sartaj et al., 2020). To begin, the group concatenated the two datasets, this was done locally and allowed the team to significantly increase the volume of data available to train and test the models. Together, the combined dataset had 2547 glioma tumor images, 2582 meningioma tumor images, 2396 images with no tumor, and 2658 images with a pituitary tumor. This is a balanced dataset which leaves the team with a good base to be able to classify each of these categories accurately without significant bias. The team then loaded the datasets onto a shared Google Colab file to get an initial idea of the size of the images collected.

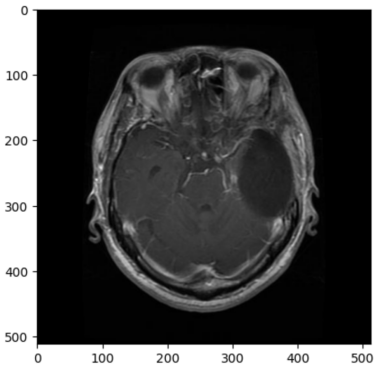

Figure 3: Image before any augmentations were made. Image: Hussein Ismail

```python
import torch

#convert to tensor
x = torch.from_numpy(img)
print(x.shape)

# move the channel dimension to the beginning
x = x.permute(2,0,1)
print(x.shape)
```

```
torch.Size([512, 512, 3])
torch.Size([3, 512, 512])
```

Figure 4: Code to get Image Dimensions with output. Image: Hussein Ismail

As seen in Figures 3 and 4 above, the visualization of one of these images shows that the input dimensions are $3 \times 512 \times 512$, indicating that the images have 3 color channels with a length and width of 512 pixels. With this knowledge, the group proceeded to clean the data. Since the images are black and white (MRI scans), having 3 channels was unnecessary and would only increase processing time and add noise. This was changed to 1 grayscale channel where a value of 255 indicated a white pixel and 0 indicated black. These values were then normalized to take on values between 0 and 1. Furthermore, the team decided to decrease the size of the images to 224×224 to decrease the number of input parameters, which in turn will decrease training time (Thambawita et al., 2021).

Finally, augmentations were made to the images to increase the robustness of the model. This is because images that will be inputted into the model may not be under ideal conditions. for example, scans might be taken from different angles and can be blurry, faded, or oriented differently. To ensure our model can deal with these circumstances, augmentations were made to the original images to simulate imperfections. Random rotations of up to 45 degrees were applied to each of the images so that the model has experience with MRI scans that are oriented imperfectly. Furthermore, a discoloring effect was applied to images with a probability of 0.5. This was added to the images to simulate a faded scan. Finally, Gaussian Blur was applied to simulate a scan where the patient moved slightly. Seen below in Figure 5 are multiple MRI images with augmentations applied.

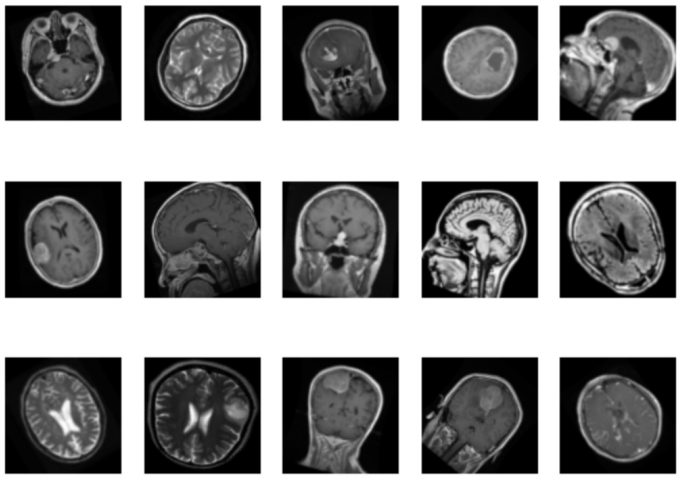

Figure 5: Examples of the Random Augmentations. Image: Hussein Ismail

Moving forward, to get reliable test accuracy, data that has never been seen by the model needs to be collected. Typically, this can be done by collecting primary data. However, with the nature of MRI scans being difficult to access and costly to generate, the best course of action to collect data is to find another publicly accessible dataset that will be used entirely for testing.

3.2 Baseline Model

For the baseline model, the team augmnented a simple convolutional neural network, originally designed for pneumonia detection from x-ray imagery (Foster, 2019). This simplistic CNN serves as an ideal benchmark for assessing our neural network's performance, as it also deals with a similar input of medical CT scans.

3.2.1 Model Outline and Implementation

The baseline model consists of 2 convolutional layers, 2 max-pooling layers, 1 flattened layer, 2 fully connected layers, and 1 dropout layer. The original output layer consists of one neuron that provides the images with a binary classification based on whether they contain pneumonia or not.

By modifying certain aspects of the model, the team retrofitted the inputs and the outputs while keeping the general inner workings consistent with its original purpose. Firstly, the number of input channels was reduced from 3 to 1 due to the black-and-white nature of our scans. Only one color channel is necessary. Next, the input image resolution to be fed into the model was increased from 150×150 pixels to 224×224 pixels to match the size of the images after they were processed. Finally, the output layer of the baseline model was augmented to contain 4 neurons so that it could predict between the 4 classes used in the project.

With these changes in place, the team implemented the model in PyTorch. Within the model, the two convolutional layers contained 32 and 64 output channels, respectively. A dropout layer with a dropout rate of 50% was added to the fully connected layers. To train, this model uses the Cross-Entropy loss function to calculate error, and the Adam optimizer to adjust the weights and biases during backward passes. Also, the hyperparameters for this model were set at a batch size of 32, a learning rate of 0.001, and an epoch size of 10. These hyperparameter choices allowed for efficient train times that allowed the group to obtain baseline results quickly. See model structure below, in Figure 6.

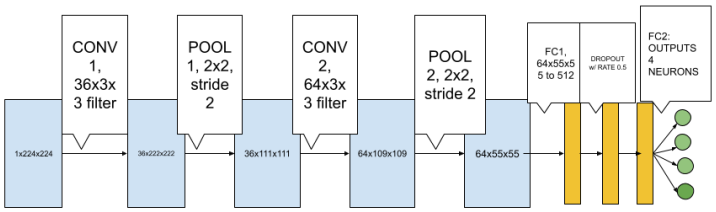

Figure 6: Baseline model visualization. Image: Marco Mastrangelo

3.2.2 Quantitative and Qualitative Findings

With these parameters in place, the baseline model achieved a training accuracy of 88.1% and a validation accuracy of 86.2% while achieving a loss below 0.35 for both the validation and training. With the test set, 87.3% accuracy was achieved. See Figure 7 below for a training/validation curve of for the baseline model.

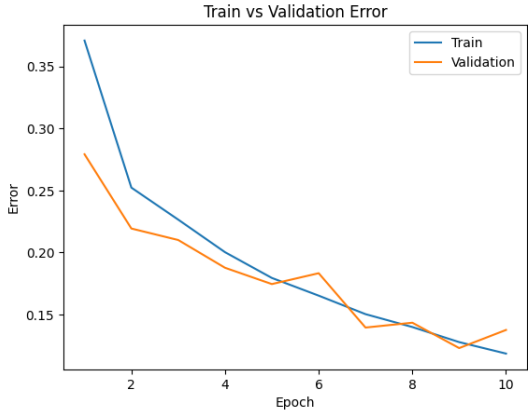

Figure 7: Train and validation error over 10 epochs. Image: Damilola Aina

These numbers prove the feasibility of the model to serve as a baseline, but to refer to a major ethical consideration within our proposal, the presence of false negatives and positives could drastically affect the outcome. Seen below is the confusion matrix of the baseline model on the test set.

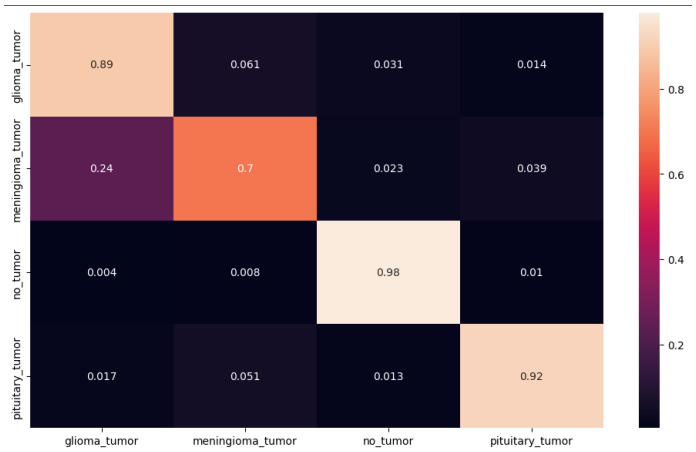

Figure 8: Confusion Matrix obtained from testing baseline model. Image: Damilola Aina

The model is almost perfect in detecting whether a scan has no tumor, achieving a 98% accuracy. The pituitary tumor detection also shows good results, with an accuracy of 92%. However, glioma and meningioma classification has lots of room for improvement. The model seems to misclassify the glioma and meningioma tumors at a high rate, with about 24% of the meningioma tumors classified as glioma tumors and 6.1% of the glioma tumors classified as meningioma tumors. This indicates that the baseline model may be too simple to differentiate between meningioma and glioma tumors.

With the smaller size of this model, it took 10 minutes to train with a rate of about an epoch per minute. With this model in place, we planned to improve on it's accuracy with the creation of the first version of the primary model.

3.3 Primary Model

For the primary model, the group wanted to experiment with CNN architectures that are well known for being used for medical image recognition and classification. Kumar-Mall et al. (2023) reviews past and current image recognition studies using CNNs and describes various model structures. One model structure outlined in this paper that caught the group's attention was *ResNet-18*, a variation of a CNN model introduced in He et al. (2015). Using transfer learning, the group fine-tuned this model to adapt it to the task of classifying each brain MRI image as having a meningioma tumor, glioma tumor, pituitary tumor, or no tumor present.

3.3.1 Model Outline

ResNet-18 is a deep CNN model (18 layers) that works to reduce the effects of the vanishing gradient problem by implementing skip connections. ResNet-18 is the smallest version of the ResNet family, with other versions containing up to 152 layers (He et al., 2015). The group chose to use the small version to reduce train time, as it is a relatively shallow model compared to its other versions. This model contains 18 layers – 17 convolutional layers, and a fully-connected output layer. These convolutional layers are split into four main stages, with each stage containing two residual blocks (two convolutional layers per residual block). Each of these residual blocks contains a skip connection to the next residual block in the stage. At each convolutional layer, batch normalization is applied, followed by a ReLU activation function. Downsampling is carried out using a stride value of 2, rather than applying pooling layers. After all convolutional layers, global average pooling is applied to reduce a $512 \times 7 \times 7$ feature map to a $512 \times 1 \times 1$ vector, which is then fed into one fully connected layer that outputs probabilities of 1000 classes using a softmax function. Specific information on filter size and number of filters applied at each layer can be seen in Figure 9 below.

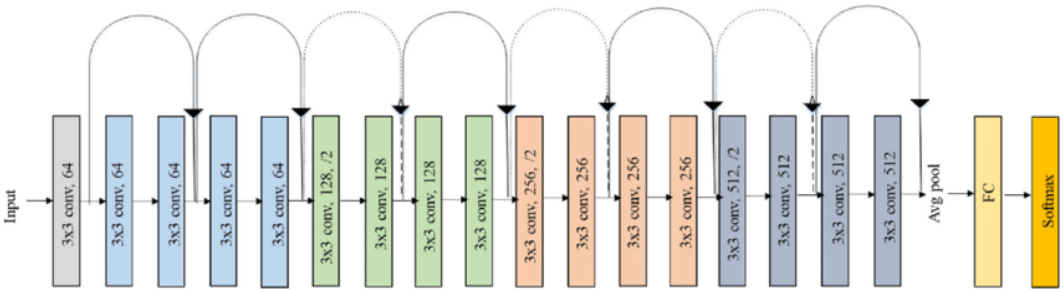

Figure 9: Original ResNet-18 Architecture. Image: Farheen Ramzan

3.3.2 Implementation

Modifications had to be made to the original model to adapt to our task/project. To start, the pre-trained model was loaded onto Colab using the `torchvision.models` package. Furthermore, ResNet-18 takes in RGB images of size $3 \times 224 \times 224$, and the MRI images in the group's dataset are grayscale, of size $1 \times 224 \times 224$. This was modified accordingly. This model was originally made for the ILSVRC competition, where images belong to 1 of 1000 classes. The MRI images each belong to 1 of 4 classes, so the final fully connected layer was adjusted accordingly for this as well. The last modification to the architecture involved freezing the weights first two stages (first 8 convolutional layers) of the model. This was done to speed up training time, which allowed the group more time to tune hyperparameters. See Figure 10 below for an image of the group's modified architecture.

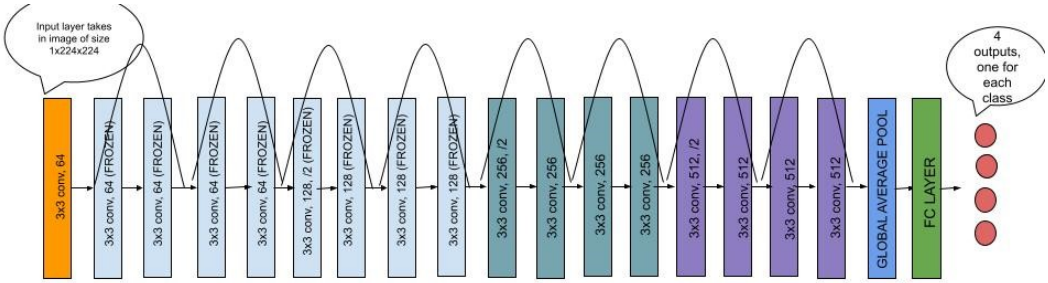

Figure 10: Group 20's modified ResNet-18 Architecture. Image: Marco Mastrangelo

Several values were tested for each hyperparameter, as well as different optimizers. The final model that is to be documented in Section3.3.3 was trained using a batch size of 128 and a learning rate of 0.01 for 25 epochs. The model also uses the Adam optimizer and a cross-entropy loss function, as it is a multiclass classification problem. These choices allow for maximizing validation accuracy while minimizing train time.

3.3.3 Quantitative and Qualitative Findings

The model took roughly 2 hours to train, which was a surprise to the group, as members were under the impression that a pre-trained model would not have to make many drastic changes to the weights to adapt to the new task. Minimizing train time will be a topic to be explored further by the group going forward, discussed in Section 4. After 25 epochs, the model achieves a 96.77% train accuracy (error rate 3.23%) and a 93.56% validation accuracy (error rate 6.44%). After using a separate test set on the model, we achieve a test accuracy rate of 95.78% (as seen in Figure 12). Figure 11 shows a train and validation error rate curve over 25 epochs.

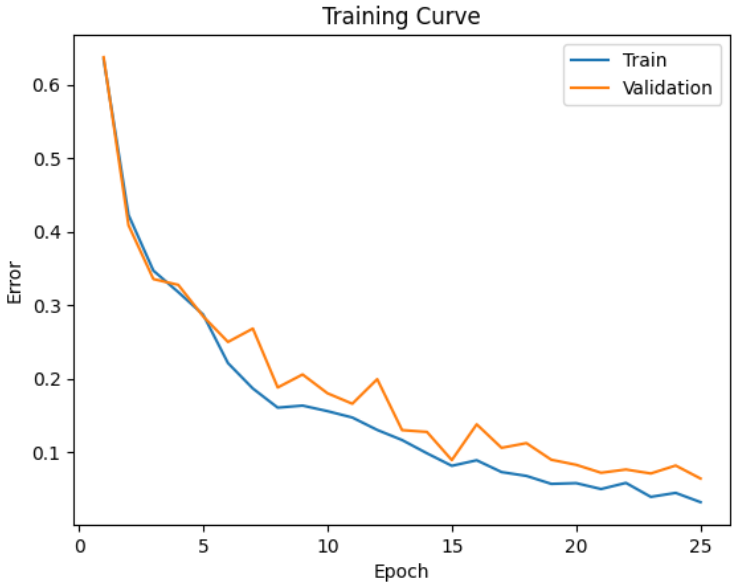

Figure 11: Train and validation error rates over 25 epochs. Image: Marco Mastrangelo

As seen by the curve above, a plateau has not been definitively reached at 25 epochs. Therefore, the model may be slightly underfitting. The model could most definitely be run for 5-10 more epochs in the future in order to fully define when overfitting may start to occur. To start uncovering qualitative findings, the group can use a confusion matrix to address and uncover trends in the model's prediction patterns, as seen in Figure 12.

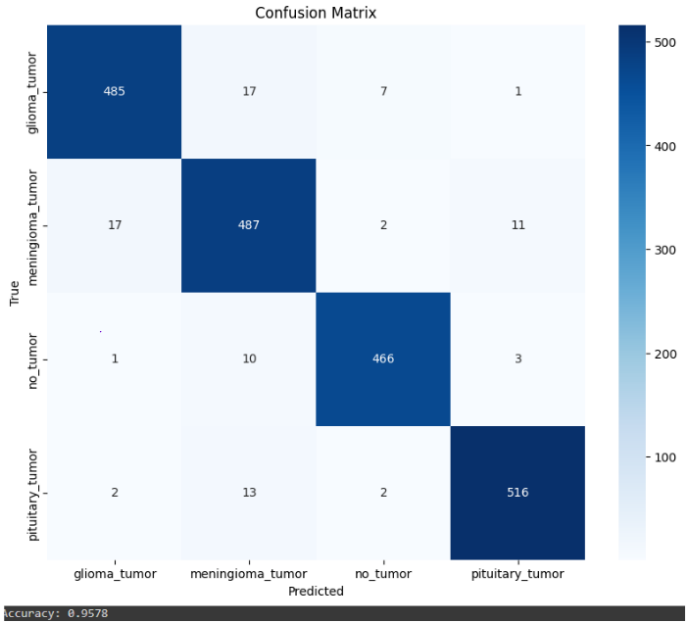

Figure 12: Confusion matrix on the test set for the primary model. Image: Marco Mastrangelo

As seen above, the model seems to have the most trouble identifying subtle differences between glioma and meningioma scans. The model incorrectly predicted that 17 meningioma tumors were glioma tumors, and that 17 glioma tumors were meningioma tumors. These are the highest false-

positive counts across all incorrect predicted vs. true combinations. This can possibly be attributed to the fact that glioma tumors can form in multiple locations of the brain, whereas meningioma tumors are usually located in a part of the brain called the dura mater, and pituitary tumors are always located in the pituitary gland (Miami Neuroscience Center). A subtype of a glioma called an oligodendroglioma forms almost exactly where meningiomas form (near the dura mater), which is why the model may misclassify these at times (Miami Neuroscience Center).

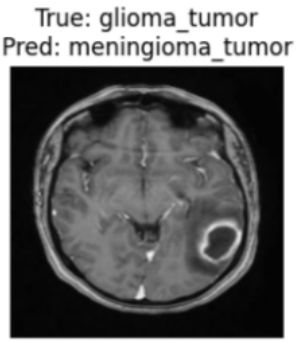

Figure 13: Misclassified glioma tumor. Image: Marco Mastrangelo

As seen in Figure 13, the model predicts a meningioma tumor, when the true tumor is a glioma (this is most likely an oligodendroglioma). Many more examples of the misclassification of gliomas and meningiomas look similar to this. Errors in classification seen between other classes can be due to image quality and subtle differences in brain structure from person to person, that may cause the model to believe that a specific tumor is present.

4 Future Steps

As the group continues to progress with the project, focus will be on two major aspects:

- Training Time
- Output precision

With a 2-hour training time for the primary model, the team plans to reduce this by optimizing code and improving GPU power. This model was pre-trained and is relatively shallow compared to other pre-trained CNNs that are available, so reducing train time must be addressed. The group also has plans to implement the *U-Net* architecture to allow the model to not only detect the presence of a specific tumor, but also indicate its location within the scan Ronneberger et al. (2015).

In conclusion, the team has designed the first version of the primary model, serving as a starting point for future improvements. The major next step is to either reduce or at least maintain training time while increasing the functionality of the model.

REFERENCES

Lyron Foster. Building a medical image classifier with deep learning and python. *Medium*, 2019.

Kaiming. He, Xiangyu. Zhang, Shaoqing. Ren, and Jian Sun. Deep residual learning for image recognition. *Computer Vision and Pattern Recognition*, 2015.

Pawan. Kumar-Mall, Pradeep. Singh, Swapnita. Srivastav, Vipul. Narayan, Marcin. Paprzycki, Tatiana. Jaworska, and Maria Gahnza. A comprehensive review of deep neural networks for medical image processing: Recent developments and future opportunities. *Healthcare Analytics*, 2023.

Miami Neuroscience Center. Brain tumor types.

Msoud Nickparvar. Brain tumor mri dataset, 2021. URL `https://www.kaggle.com/dsv/2645886`.

Quinn. Ostrom, Giono. Cioffi, Haley. Gittleman, Nirav. Patil, Kristin. Waite, Carol. Kruchko, and Jill. Barnholtz-Sloan. Cbtrus statistical report: Primary brain and other central nervous system tumors diagnosed in the united states in 2012–2016. *Neuro-Oncology*, 1 -100, 2019.

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

Bhuvaji. Sartaj, Kadam. Ankita, Bhumkar. Prajakta, Dedge. Sameer, and Kanchan Swati. Brain tumor classification (mri), 2020. URL `https://www.kaggle.com/dsv/1183165`.

Vanjira. Thambawita, Imga. Strumke, Steven. Hicks, Pal. Halvorsen, Sravanthi. Parasa, and Michael. Riegler. Impact of image resolution on deep learning performance in endoscopy image classification: An experimental study using a large dataset of endoscopic images. *Diagnostics (Basel)*, 2021.

Naeem. Ullah, Ali. Javed, Ali. Alhazmi, Syed. Hasnain, Ali. Tahir, and Rehan. Ashraf. A unified deep learning model for brain tumor detection and classification. *PLOS ONE.*, 2023.