Brain Tumor Detection

Russel Tjahjadi Ngoc Khoa Le

Introduction

In this project we want to develop a machine learning model that will help classify images brain MRI images into two categories. Brain tumors are a significant health concern due to their effects on cognitive function, motor skills, and overall quality of life.

Brain tumors could eventually lead to brain cancer. According to the Cancer Council, it is estimated that more than 1,900 people were diagnosed with brain cancer in 2023 and that it affects more on the elderly as they said that the average age at diagnosis is at 59 years old (Cancer Council, 2023). Below is shown the decrease of brain cancer as technology advances (*Brain Tumors*, 2019).

Detecting brain tumors can be challenging for several reasons. The first reason is that it is because we might not know where exactly it is located in the brain. As the brain is a highly complex organ,tumors can develop in various locations in the brain. Some of these tumors are not obvious and sometimes because of human error, doctors can wrongly prescribe a patient who has tumors and the other as well. In addition to that, these tumors can be similar to normal tissue where it is quite an obstacle to identify between healthy and cancerous tissue.

There are also a lot of tumor types. There are many different types of tumors each with their own unique way of us identifying it under the MRI. With this in mind, the symptoms that these tumors give out could also overlap with the other conditions.

The project addresses a critical real-world problem in healthcare and we hope that this project would contribute towards the early detection of brain tumors from medical imaging. With the insidious nature of brain tumors and with their potential to elude early diagnosis, underscores the urgency for innovative solutions. This initiative aims to address this challenge by creating an advanced algorithm capable of early detection, thereby facilitating prompt and effective treatment strategies. By employing machine learning techniques, we aspire to significantly enhance the capabilities of medical professionals in identifying and managing brain tumors. Characteristics such as pattern recognition with which the machine learning algorithm can be trained to recognize patterns and anomalies in the data.This capability allows them to identify subtle differences between normal and abnormal brain tissues that may be indicative of a tumor.

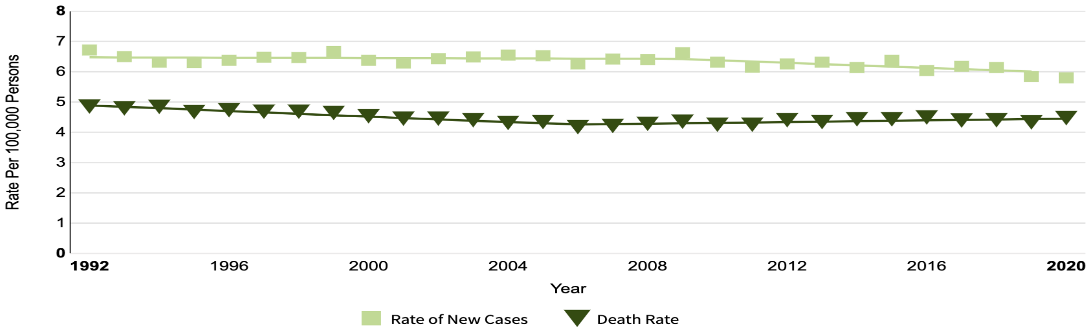


Figure 1. Rate of Brain Cancer

Method

The method that we are currently using is the Convolutional Neural Network. Our approach is to load the trained images with labels (supervised learning) to the model and then testing them on the test data which is new and unseen. We have decided to proceed with this method because CNN is specialized for applications in image processing.

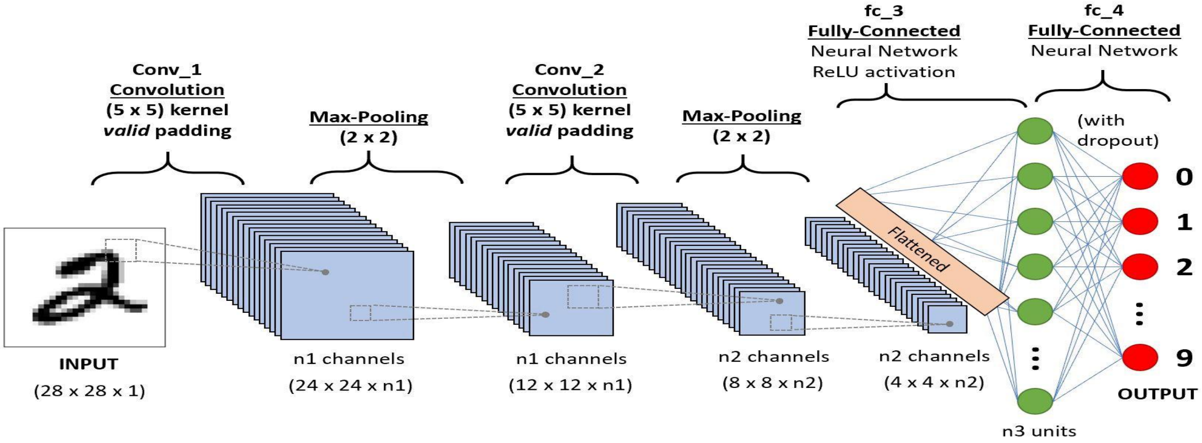


Figure 2. CNN Architecture (Ratan, 2020)

The first layer is the convolutional layer which is a typical neural network where the inputs are being fed to the layer and these input layers are then connected to the next hidden layer. This layer is responsible for learning local patterns and features in the input image. With this layer, activation functions like ReLU are applied to introduce non-linearity and allow the network to learn more complex representations.

The second layer is the pooling layer where it is used to reduce the dimensionality of the feature map. Pooling helps make the learned features more robust to translations and variations in the input and it reduces the computation complexity of the network.

The third layer is the fully connected layer. This layer is responsible for the combination of features learned in the previous layers to make future predictions. The output of the fully connected layer is usually passed through the softmax activation function, which is then used to obtain a probability distribution over the classes.

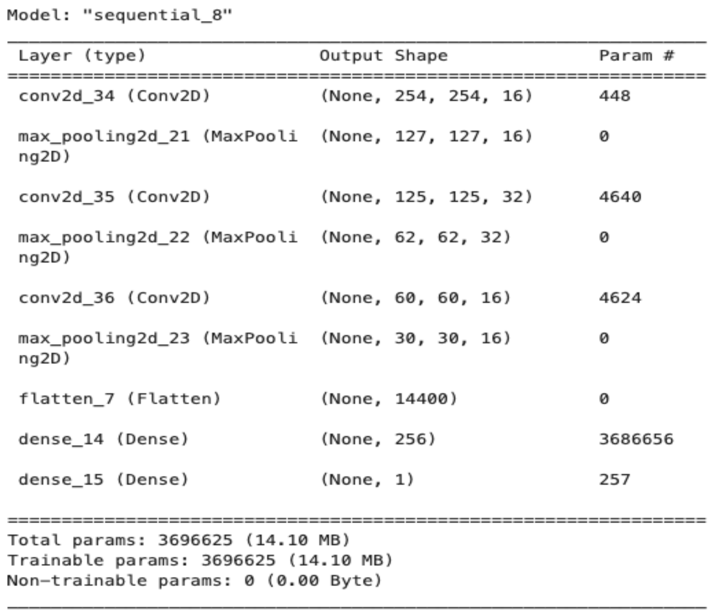
The first step is to make sure we have our editors set up. Downloading Anaconda which will help us run a Jupyter notebook is a crucial step for us to do because without it, we cannot continue with the model training and data analysis. Then, we have installed the necessary libraries locally which will then be run on the launching of anaconda. Then, we trained two models. The first model is a binary classification and the second mode is a multi classification model.

For our first attempt in trying to train the binary classification, we obtain the dataset from this Kaggle website (*Brain Tumor Classification [CNN|977%]*, n.d.). : https://www.kaggle.com/code/zahraaalaatageldein/brain-tumor-classification-cnn-977/input

Once we have downloaded the zip, we then extracted the zip and decided to only utilize 2 classes: [no tumor] and [meningioma]. With our output, we decide to set the label as 1 for [no tumor] and 0 for [meningioma]. For reading in the files, the keras library was utilized as the keras library offers a wide variety of options for us to control.

The model uses the 2D convolutional layer, Conv2D, and with this function/class, we are able to add the activation function of ReLU (Rectified Linear Unit). With that in mind, the model also uses the ReLU because this activation function introduces non-linearity to the model as the neural networks need to understand that the data is not linearly separable. In our case for brain image pattern recognition, it exhibits real world data which shows complex structures and patterns that cannot be captured by a simple linear decision boundary. In addition to that, we have an adam optimizer with a learning rate of 0.001. The Adam optimizer was used because it is generally better than every other optimization algorithm and usually requires fewer parameters for tuning. There is also an addition of the sigmoid activation function because of this model using and training for a binary classification. This function maps the network's raw output to the value of 0 and 1 which can be interpreted as a probability. For example, if the probability is found to be below 0.5, then it is classified as that class, else, the other.

We train the CNN model by first separating the dataset into 3 classes: training, testing and validation. We then moved some of the images from the testing folder available in the Kaggle website to the validation folder. As we utilize only the Meningioma and No Tumor as 2 classes, we realized that the algorithm might be able to detect the tumor because there seems to be a white spot in the MRI image while there is supposedly none in a healthy brain.



```
Model: "sequential_8"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_34 (Conv2D)          (None, 254, 254, 16)      448

 max_pooling2d_21 (MaxPooli  (None, 127, 127, 16)      0
 ng2D)

 conv2d_35 (Conv2D)          (None, 125, 125, 32)      4640

 max_pooling2d_22 (MaxPooli  (None, 62, 62, 32)        0
 ng2D)

 conv2d_36 (Conv2D)          (None, 60, 60, 16)        4624

 max_pooling2d_23 (MaxPooli  (None, 30, 30, 16)        0
 ng2D)

 flatten_7 (Flatten)         (None, 14400)             0

 dense_14 (Dense)            (None, 256)               3686656

 dense_15 (Dense)            (None, 1)                 257

=================================================================
Total params: 3696625 (14.10 MB)
Trainable params: 3696625 (14.10 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

Figure 3. First CNN Model Summary

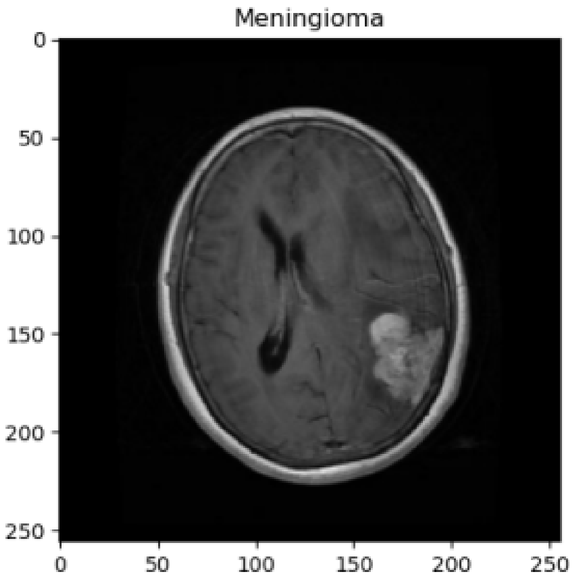


Figure 4.1 Meningioma

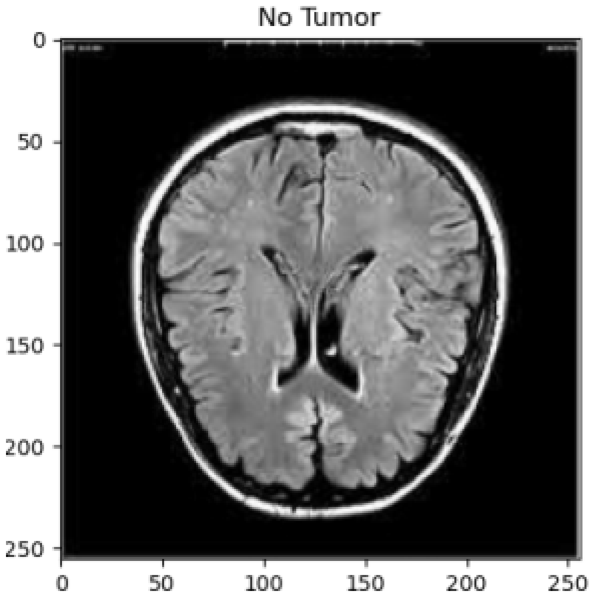


Figure 4.2 No tumor

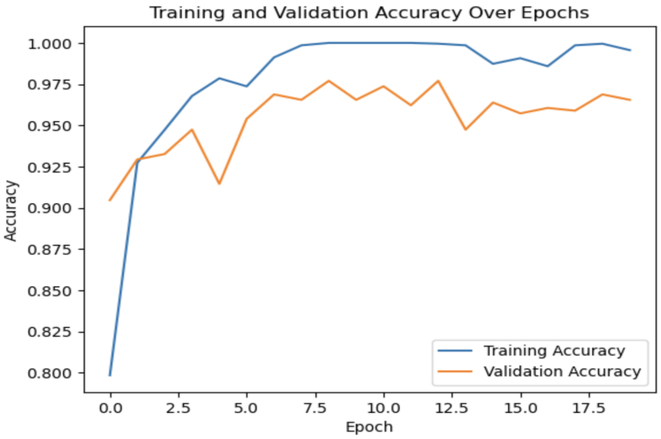


Figure 5.1 Accuracy

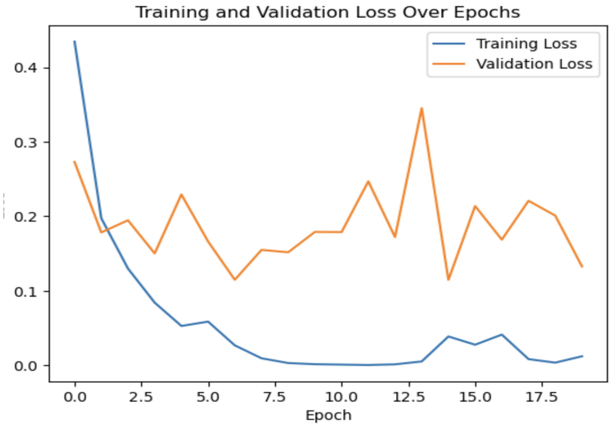


Figure 5.2 Loss

Based on the figures above, we observed that our first attempt was a good try because the loss started to decrease and that the accuracy started to increase as the number of epochs was run.

In addition to that, the model was also tested on another different dataset.

Classification	Correct Prediction	Total Images	Accuracy
Meningioma	289	306	94.44%
No Tumor	399	405	98.52%

Table 1. Testing result of model 1

From the testing result, we can see that our first model is very successful with high accuracy prediction upon unseen data.

After the first model, we make changes to it so it can be trained to classify multiple classes. In this case we want the model to not only detect tumors but also be able to differentiate between different types of tumor including [glioma], [meningioma], [pituitary] and [no tumor].

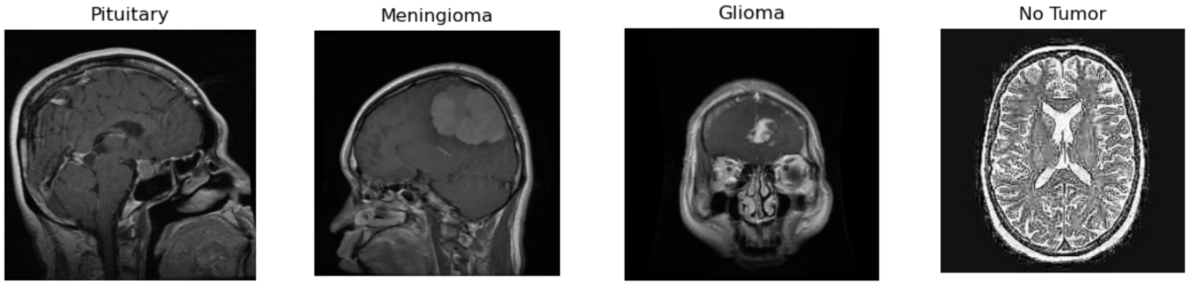
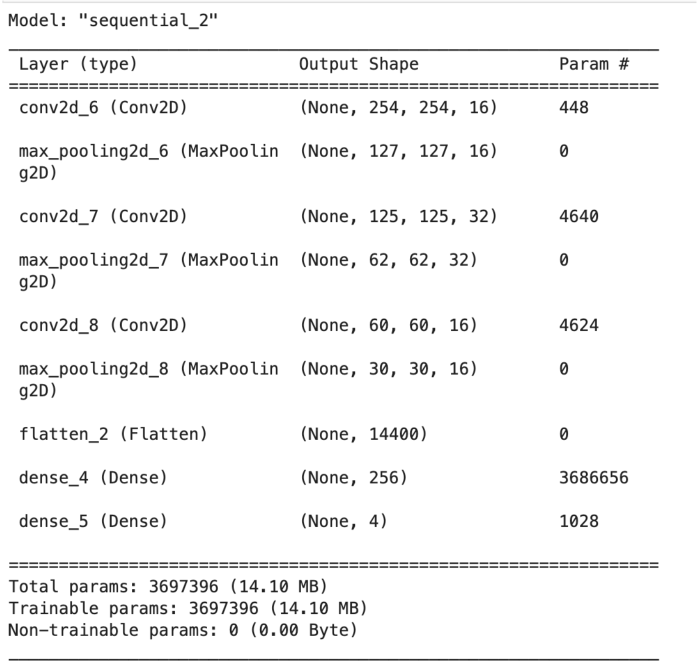


Figure 6. Four classes of data

The model does not change in terms of layers and general design. The sigmoid activation function, tailored for binary classification, was replaced with the softmax activation function to facilitate multi-class categorization. This enhancement enables the model to assign probabilities across multiple classes, providing a more nuanced and probabilistic perspective in the decision-making process. We still used Adam Optimizer with a learning rate of 0.001 for training, aiming for an optimal trade-off between rapid adaptation and precision. A substantial shift in the choice of loss function is the replacement of Binary Cross Entropy with Categorical Cross Entropy. Categorical Crossentropy is a loss function commonly used in multi-class classification problems.

Here is our summary for our CNN model for the multi-classification problem:



```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_6 (Conv2D)           (None, 254, 254, 16)      448

 max_pooling2d_6 (MaxPoolin  (None, 127, 127, 16)      0
 g2D)

 conv2d_7 (Conv2D)           (None, 125, 125, 32)      4640

 max_pooling2d_7 (MaxPoolin  (None, 62, 62, 32)        0
 g2D)

 conv2d_8 (Conv2D)           (None, 60, 60, 16)        4624

 max_pooling2d_8 (MaxPoolin  (None, 30, 30, 16)        0
 g2D)

 flatten_2 (Flatten)         (None, 14400)             0

 dense_4 (Dense)             (None, 256)               3686656

 dense_5 (Dense)             (None, 4)                 1028

=================================================================
Total params: 3697396 (14.10 MB)
Trainable params: 3697396 (14.10 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

Figure 7. Second CNN Model Summary

As such, we ran the training algorithm through the new training data with 20 epochs. Ultimately, we got following results of loss and accuracy from training and validation:

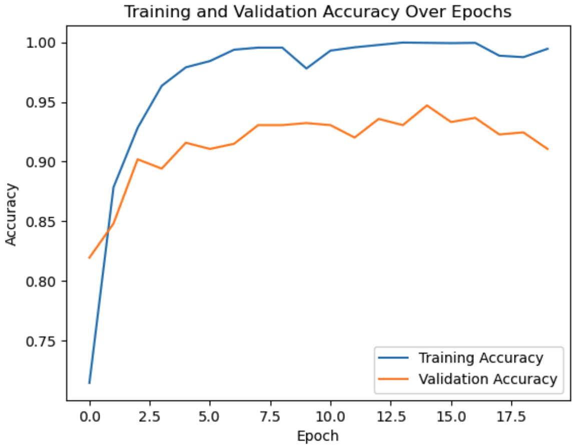


Figure 8.1 Accuracy

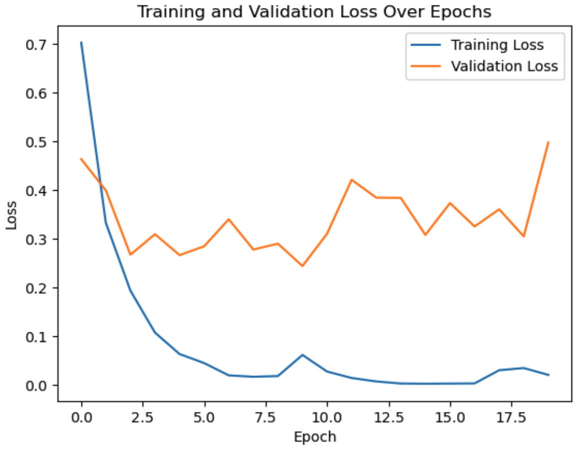


Figure 8.2 Loss

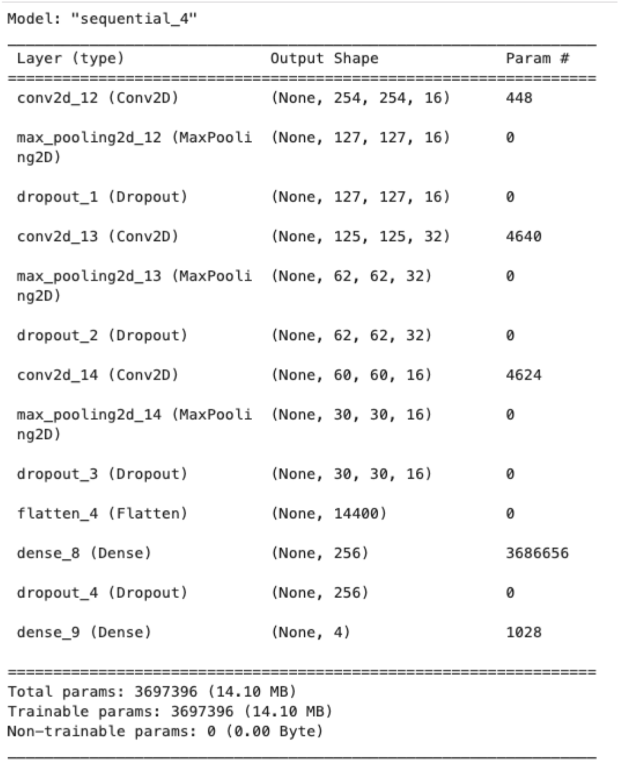
After that, we ran the trained model through an unseen test data set with following result:

Classification	Correct Prediction	Total Images	Accuracy
Meningioma	223	306	72.88%
No Tumor	402	405	99.26%
Glioma	280	300	93.33%
Pituitary	300	300	100.00%

Table2. Testing result of model 2

From the testing result, we can see good testing accuracy from Meningioma, Glioma and Pituitary with accuracy > 90.00%. However, we see a low accuracy when testing on the Meningioma data set. This might be due to overfitting as model 2 does not have a way to avoid overfitting.

To prevent overfitting and improve accuracy of this model. We add dropout to our model:



```
Model: "sequential_4"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_12 (Conv2D)          (None, 254, 254, 16)      448

 max_pooling2d_12 (MaxPooli  (None, 127, 127, 16)      0
 ng2D)

 dropout_1 (Dropout)         (None, 127, 127, 16)      0

 conv2d_13 (Conv2D)          (None, 125, 125, 32)      4640

 max_pooling2d_13 (MaxPooli  (None, 62, 62, 32)        0
 ng2D)

 dropout_2 (Dropout)         (None, 62, 62, 32)        0

 conv2d_14 (Conv2D)          (None, 60, 60, 16)        4624

 max_pooling2d_14 (MaxPooli  (None, 30, 30, 16)        0
 ng2D)

 dropout_3 (Dropout)         (None, 30, 30, 16)        0

 flatten_4 (Flatten)         (None, 14400)             0

 dense_8 (Dense)             (None, 256)               3686656

 dropout_4 (Dropout)         (None, 256)               0

 dense_9 (Dense)             (None, 4)                 1028

=================================================================
Total params: 3697396 (14.10 MB)
Trainable params: 3697396 (14.10 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

Figure 9. Third CNN Model Summary

By adding Dropout(0.25) at the , we got a new testing accuracy result:

Classification	Correct Prediction	Total Images	Accuracy
Meningioma	258	306	84.31%
No Tumor	399	405	98.52%
Glioma	282	300	94.00%
Pituitary	296	300	98.67%

Table 3. Testing result of model 3

From the testing result, we can see that despite not being very good compared to other classes, Meningioma has a much better accuracy compared to model 2 with the increase from 72.88% to 84.31%. This means that model 3 resolved some overfitting issues that present in model 2.

Results and Discussion

Table 1 shows the accuracy of Model 1 on a data set with two classes. Model 1 is very successful without the need for considering overfitting with accuracy around 95% and above. However, when we increased the data set to have two more classes, the same model struggled to keep an average accuracy of above 80% for Meningioma. We observe that by adding Dropout, our model suffers less from overfitting because dropout is a form of regularization that introduces noise during training, preventing the network from becoming overly reliant on specific features of the training data. This encourages the model to learn more general and robust representations, ultimately improving its performance on unseen data and reducing overfitting.

Conclusion

In this project, we have observed that even when our first attempt was rather optimal, we still needed to keep caution on the next models that we were going to build. We also realized that the number of epochs does not mean the more accurate it is, it could even lead to overfitting. It was finding the right optimizer with the right tuned parameters. We tried using different optimizers but we settled on Adam because we decided that it's best for the algorithm.
Overall, the project made us more confident in knowing how machine learning works in general and how it can potentially save people's lives.

References:

1. Cancer Council. (2023). *Brain cancer | Causes, Symptoms & Treatments*. Www.cancer.org.au. https://www.cancer.org.au/cancer-information/types-of-cancer/brain-cancer
2. *Brain Tumors*. (2019). National Cancer Institute; Cancer.gov. https://www.cancer.gov/types/brain
3. *Brain Tumor Classification [CNN|977%]*. (n.d.). Kaggle.com. Retrieved December 6, 2023, from https://www.kaggle.com/code/zahraaalaatageldein/brain-tumor-classification-cnn-977/input
4. Ratan, P. (2020, October 28). *Convolutional Neural Network Made Easy for Data Scientists*. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2020/10/what-is-the-convolutional-neural-network-architecture/