# Brain Tumor Detection with Multi-Class CNN's and MLP

Hung Do, Amira Bendjama, Kenneth Pan, Ramona Rubalcava

***Abstract*—Brain tumors, whether cancerous or not, pose serious health risks, potentially causing severe brain damage or loss of life. Timely and accurate detection is vital for effective treatment and better patient outcomes. Magnetic Resonance Imaging (MRI) is valuable for visualizing these tumors, but manual MRI analysis is time-consuming and requires expertise. In this study, we propose and compare two distinct approaches: Multilayer Perceptron (MLP) and Convolutional Neural Network (CNN) models, to automate brain tumor detection and classification. Our models categorize tumors into glioma, meningioma, pituitary, or the absence of a tumor, offering a promising solution for expediting diagnosis and treatment. Remarkably, we implemented these models without using machine learning libraries, incurring significant computation costs. This unique approach contributed to MLP's superior performance over CNN, with our experimental results indicating that our models achieved up to 97.89% and 57% classification accuracy, respectively, for the used dataset.**

***Index Terms*—Machine Learning, Computer Vision, Deep Learning, MLP, CNN, Brain tumors**

## I. Introduction

Today, deep learning networks are leveraged to tackle a wide range of problem sets, commonly computer vision tasks, present in the medical field. One important task that is reliant on computer vision, specifically object detection, is the task of early detection of brain tumors found in MRI scans. Brain tumors are abnormal cell growth that can develop in the brain and be classified as malignant (cancerous) or benign (non-cancerous)[5]. While there are many different types of brain tumors and each type has the possibility of being either benign or malignant[5], this project only considers glioma, meningioma, and pituitary tumors. Early recognition and diagnosis of these tumors are critical for improving the quality of life and survival rate of patients. Nevertheless, conventional methods of detecting brain tumors, such as magnetic resonance imaging (MRI) and computed tomography (CT), are time-consuming, pricey, and require human expertise. As a result, there is a growing need to develop an automated and precise method of detecting brain tumors using deep learning techniques.

In this project, we develop and test a convolutional neural network (CNN) and multi-layer perception (MLP), without the use of machine learning libraries, to detect brain tumors using images of MRI scans. The images of MRI brain scans used in this project contain the brain tumor types glioma, meningioma, and pituitary tumors discovered in various patients' brains.

Our objective is to explore and compare the effectiveness of the CNN and MLP deep learning architectures in detecting the different brain tumor types. Specifically, we compare the training time, overall loss, and final accuracy of both architectures. These different architectures offer distinct approaches to leveraging deep learning methods for computer vision, specifically object detection.

## II. Related work

Saikat et al.[7] proposed an advanced approach for precise brain tumor detection. Their study leverages deep CNNs to enhance the accuracy of brain tumor identification, addressing the challenges associated with manual MRI image analysis. The research introduces two distinct deep-learning models designed to identify both binary (normal and abnormal) and multi-class (meningioma, glioma, and pituitary) brain tumors. Two publicly available datasets, comprising 3064 and 152 MRI images, are utilized to develop and evaluate these models.

The initial model employs a 23-layer convolutional neural network (CNN) to analyze the first dataset, benefitting from its extensive training data. However, the second dataset presents a challenge due to its limited volume, leading to overfitting issues with the original "23-layer CNN" architecture. To mitigate this problem, transfer learning techniques are employed. Specifically, the authors combine the VGG16 architecture with a refined version of their "23 layers CNN" architecture. This combination resulted in an accuracy of 97.8%.

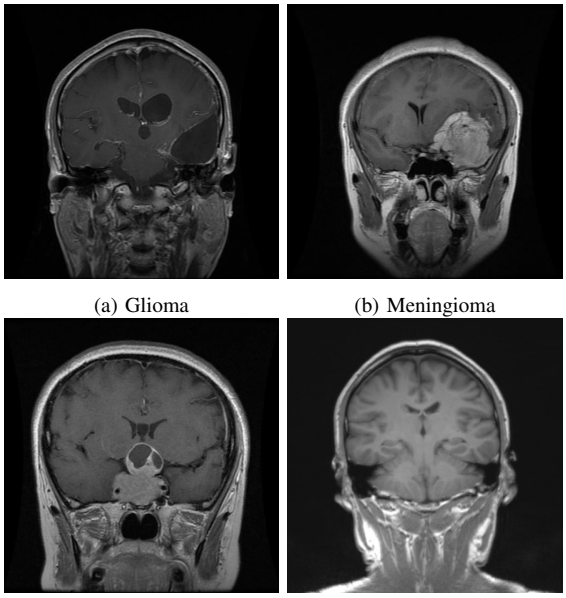
Another related work we referenced when building our CNN was the study, MRI-based brain tumor image detection using CNN-based deep learning method[3]. The data used in this article was the 2020 BraTS dataset, which contains images of brain tumor MRI scans[1]. The model that achieved the highest accuracy was trained on a data split of 2473 training images and 273 testing images. This study aimed at utilizing CNNs to detect the two classes, MRI images with brain tumors and MRI images without.

Their architecture consisted of a 9-layer CNN, which had an initial convolutional layer with 32 filters of size 2 by 2, followed by batch normalization and a max pooling layer of size 2 by 2. Following the max pooling layer, they used another convolution layer with 64 filters of size 2 by 2, followed by another batch normalization and max pooling layer. The final max pooling output was then sent to a dropout layer to avoid overfitting. After the dropout layer, the data passed through a dense layer of 512 nodes, then to the final fully connected layer with a SoftMax activation. The activation function used in the convolutional layers was the ReLU (Rectified Linear Unit) activation function and the objective function used was cross-entropy. While it is not stated how long it took this model to train, this architecture achieved an accuracy of 99.74

We adopted methodologies used by both studies when implementing our CNN, which served as the basis for our model's architecture, and we fine-tuned the parameters and filters accordingly to suit our specific dataset and requirements.

## III. Data and Pre-Processing

The data used in this project is the Brain Tumor MRI Dataset[8] obtained from Kaggle. The dataset is a combination of the three datasets; brain tumor dataset[4], Brain Tumor Classification (MRI)[2], and Br35H dataset[6]. This combination contains 7023 images of MRI scans that either contain a brain tumor or no tumor. These images were further separated into four classes; glioma, meningioma, pituitary, and no tumor (see Figure 1). Due to the different sources for the images, each class had different image sizes (e.g., 512 by 512 or 236 by 236 pixels). To standardize the image sizes and reduce downstream computation time, pre-processing was done to resize the image sizes to 128 by 128 pixels and convert the images to a grey scale. The decision to pre-process the data in this manner was due to the related works referenced by this project using similar specifications. Following the conversion of the images, the data was flattened, separated into training and validation sets, and saved to two CSV files. The training set consists of 5712 images, and the remaining 1311 images are used for validation. This results in a training set comprised of around 80% of the data and a validation set comprised of 20%. The 4 classes are mostly balanced, with each class taking up close to 20% to 30% of the data. Due to this, the data will have little to no bias towards any specific class. Before feeding the data to the model, the data was normalized by dividing by 255.0.



(a) Glioma (b) Meningioma

(c) Pituitary (d) No Tumor

Fig. 1: Four classes of brain tumor

## IV. Multi-Layer Perception

We constructed a Multi-Layer Perceptron (MLP) from scratch to classify MRI scans into one of the four categories: pituitary tumor, meningioma, glioma, or no tumor.

The model is a feed-forward neural network architecture. The architecture consists of:

- Input Layer
- Fully Connected Layer
- ReLU Activation Layer (To introduce non-linearity)
- Dropout Layer (Prevent overfitting)
- Fully Connected Layer
- ReLU Activation Layer (To introduce non-linearity)
- Dropout Layer (Prevent overfitting)
- Fully Connected Layer
- Softmax Output Layer (to produce probabilities for each tumor category)

In the forward propagation phase, MRI data is processed starting from the input layer and successively passes through each layer. Neurons compute a weighted sum of their inputs:

$$Z = XW + b$$

and undergo an activation function, specifically ReLU. The ReLU function is defined as:

$$H = \max(0, Z)$$

This essentially zeroes out any negative values, allowing only positive activations to pass through. By the time the data reaches the softmax layer, a probability distribution across our tumor classes is produced.

During the backward propagation phase, the model computes the error between its predictions and the actual labels. This error is then used to adjust the model's weights and biases to minimize future errors.

Weight initialization plays a pivotal role in ensuring efficient training. We adopt the He weight initialization method, which is defined as:

$$\sqrt{\frac{2}{n_{\text{prev}}}}$$

where $n_{\text{prev}}$ is the number of units in the preceding layer. He initialization offers rapid convergence, especially in networks utilizing ReLU activations.

The cross-entropy loss, characterizing our training objective, is:

$$J(Y, \hat{Y}) = -\frac{1}{m} \sum_{i=1}^{m} \sum_{j=1}^{C} Y_{ij} \log(\hat{Y}_{ij})$$

To optimize the weights and biases during backward propagation, they are updated as:

$$W = W - \eta \frac{dJ}{dW},$$
$$b = b - \eta \frac{dJ}{db},$$

where $\eta$ represents the learning rate. An early stopping mechanism is also employed to prevent overfitting, and halting training if there's no improvement in the validation loss over a specific number of epochs.

## V. Convolutional Nueral Network

CNNs are particularly adept at image tasks due to their innate ability to discern patterns and features in spatial data automatically. Our CNN's design includes:

- Input Layer
- Convolutional Layer with ReLU Activation
- Dropout Layer
- MaxPooling Layer
- Flattened Layer
- Fully Connected Layer
- Softmax Output Layer

When the MRI data goes through the CNN, it first meets the convolutional layer. Here, the model employs cross-correlation using a filter $K$ on the input $X$ to extract features. Specifically, for an $M \times M$ section of the image at position $a, b$, the result is:

$$F_{a,b} = \sum_{i=1}^{M} \sum_{j=1}^{M} X\left(a - \frac{M}{2} + i, b - \frac{M}{2} + j\right) K_{i,j}$$

Once this is done throughout the image, we get a feature map.

After this, the ReLU function ensures only the positive values remain, highlighting the features.

Next, the MaxPooling layer comes into play. It takes small areas of the feature map (like $2 \times 2$) and keeps only the largest value from that area. The concept of "stride" in MaxPooling refers to how much the pooling window moves after each operation. For example, with a stride of 2, after examining the first $2 \times 2$ section, it would move 2 units over for the next operation. This ensures efficient data compression and retains the vital attributes, aiding faster computation and minimizing overfitting chances.

Following the convolution and pooling processes, the data is sent through a dropout layer and then flattened. The Flattened Layer in a CNN serves a straightforward yet crucial purpose. After processing through convolutional and pooling layers, the data is still in a 2D or 3D shape, representing the height, width, and depth (or channels) of the feature maps. But the subsequent layers, like the Fully Connected Layer, expect data in a 1D format. Then, the network goes through the regular neural network layers until it reaches the softmax layer, which predicts the tumor-type probabilities.

Training this model means adjusting it for better predictions using the cross-entropy loss, and we use the He method to initialize the weights efficiently.

## VI. Results

Hyperparameters for MLP are as follows: $\eta = 0.01, 4000$ epochs, dropout probability = 0.1, 64 neurons for 1st Fully Connected Layer, 32 for 2nd Fully Connected Layer, and then finally 4 (classes) for final Fully Connected Layer. The hyperparameters for CNN are as follows: $\eta = 0.001, 200$ epochs, dropout probability = 0.5, kernel size = $3 \times 3$, kernel stride = 1, maxpool size = $2 \times 2$, pool stride = 2, and number of filters = 1. Different parameters and architectures were tested but below is the best achieved result.

We chose these learning rates because higher learning results in increasing loss, so it doesn't converge at all, and a slower learning rate results in it converging too slowly. The dropout probability was chosen based on how much the validation loss curve follows the training loss curve. The rest of the hyperparameters were chosen to save computation costs and time, even if they are not the most optimal hyperparameters. The confusion matrix shows that the MLP model performs well in all classes, whereas the CNN model does well in all classes except meningioma.

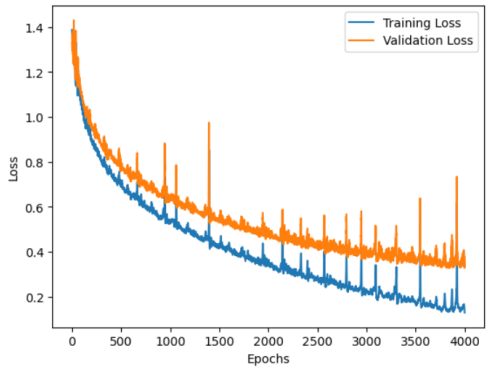



Fig. 2: MLP Loss Graph

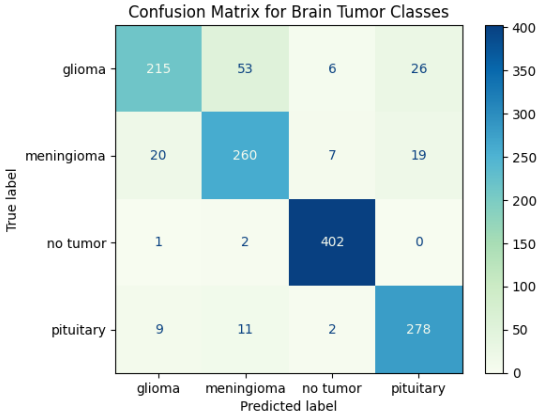



Fig. 3: Confusion Matrix for MLP

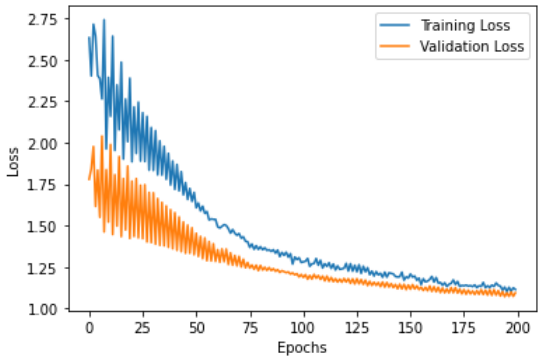



Fig. 4: CNN Loss Graph

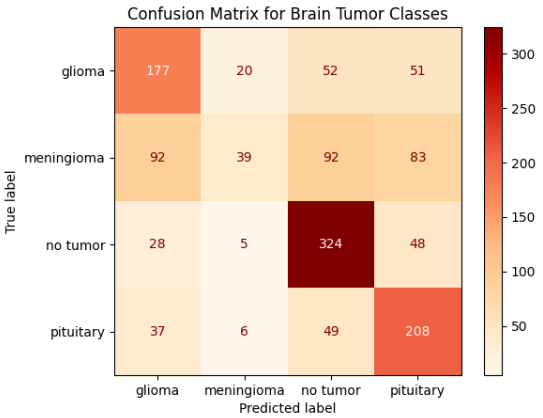



Fig. 5: Confusion Matrix for CNN

## VII. Limitations

The limitations that exist for this project as a result of the data used for training are that the MRI images contain fully formed brain tumors and do not show the progression of tumor formation. Another limitation as a result of the data is that the model developed is meant for the classification of tumor type (e.g., Glioma, Meningioma, Pituitary, no tumor), not the diagnosis of the MRI scan (e.g., benign or malignant tumor). Due to these limitations resulting from the data, the model should not be used for detecting the progression of tumor formation, or any other tumor type besides the types used in training or diagnosing benign or malignant tumors.

Limitations outside of the data include computation costs and optimization. For example, writing MLPs and CNNs proves to be difficult as writing them from scratch using only NumPy results in significant challenges both in terms of performance and efficiency. Using existing frameworks/libraries such as TensorFlow and PyTorch would offer much better model performance as they are already optimized for computational efficiency and can leverage the full potential of hardware accelerations, speeding up computations tremendously, especially for large-scale neural network architectures such as CNNs. An additional general limitation of this model is that the model should not be used without review by a medical professional.

## VIII. Conclusion

In this project, we have demonstrated an MLP model that can classify brain tumor types from MRI images and a CNN model that can be developed further to make better predictions using the same MRI images. Both models were built from scratch, and achieved different results on account of their architecture.

The MLP architecture shows the capability of detecting and classifying brain tumors after achieving a training accuracy of 96.78% and validation accuracy of 88.10% (see Appendix A.2). However, while these results show better performance compared to the CNN model, which achieved a training accuracy of only 54.71% and validation of 54.06% (see Appendix B.2), the training time needed for 4000 epochs for the MLP model was 8 hours. For the CNN model, the training time for 200 epochs was 5 hours.

We attempted various architectures and initialization parameters aimed at achieving better results. The architecture experiments include adding a convolution layer (see CNN-2.py), larger kernel and max pooling sizes, and an increased number of filters. However, these experiments never resulted in a training and validation accuracy higher than 30%. We assess that the performance of the CNN is a result of the extensive computation resources and time needed to adequately train a CNN model with high accuracy.

We further assessed our models using a variety of metrics, such as precision, recall, and F1-score (see Appendix). These calculations were done for the overall model performance and per class. We noted that in both models the classes Pituitary and no tumor and the highest precision, recall, and F1-scores (see Appendix). This suggests that the features of Glioma and Meningioma are harder for either model to discern, and likely require extra training time or more data.

Conclusively, we believed that while the MLP performs better for detection and classification, our CNN model has the potential to increase performance if trained for longer and with more bias towards the harder-to-predict classes, Glioma and Meningioma.

## IX. Future Work

Future work for this project would include the incorporation of mini-batches and or batch normalization to reduce training time and overfitting, as well as refining the use of the ADAM optimizer in training to produce consistent results. Another future work consideration would be to leverage more robust computing resources to train CNN models with more convolutional layers and for a longer duration.

## References

[1] Awsaf. *Brain Tumor Segmentation(BraTS2020)*. 2020. URL: https://www.kaggle.com/datasets/awsaf49/brats2020-training-data.

[2] Sartaj Bhuvaji et al. *Brain Tumor Classification (MRI)*. 2020. DOI: 10.34740/KAGGLE/DSV/1183165. URL: https://www.kaggle.com/dsv/1183165.

[3] Arkapravo Chattopadhyay and Mausumi Maitra. "MRI-based brain tumour image detection using CNN based deep learning method". In: *Neuroscience Informatics* 2.4 (2022), p. 100060. ISSN: 2772-5286. DOI: https://doi.org/10.1016/j.neuri.2022.100060.

[4] Jun Cheng. "brain tumor dataset". In: (Apr. 2017). DOI: 10.6084/m9.figshare.1512427.v5. URL: https://figshare.com/articles/dataset/brain_tumor_dataset/1512427.

[5] Mayo Clinic. *Brain Tumor*. 2023. URL: https://www.mayoclinic.org/diseases-conditions/brain-tumor/symptoms-causes/syc-20350084.

[6] Ahmed Hamada. *Br35H::Brain Tumor Detection 2020*. 2020. URL: https://www.kaggle.com/datasets/ahmedhamada0/brain-tumor-detection?select=no.

[7] Md. Saikat Islam Khan et al. “Accurate Brain Tumor Detection Using Deep Convolutional Neural Network”. In: (2023).

[8] Msoud Nickparvar. *Brain Tumor MRI Dataset*. 2021. DOI: 10.34740/KAGGLE/DSV/2645886. URL: https://www.kaggle.com/dsv/2645886.

## APPENDIX

### A.1 MLP Classification Report

| Class | Precision | Recall | F1-Score |
|---|---|---|---|
| glioma | 0.88 | 0.72 | 0.79 |
| meningioma | 0.80 | 0.85 | 0.82 |
| no tumor | 0.96 | 0.99 | 0.98 |
| pituitary | 0.86 | 0.93 | 0.89 |

### A.2 MLP Accuracy

| | |
|---|---|
| Training Accuracy | $96.78\%$ |
| Validation Accuracy | $88.10\%$ |

### A.3 MLP Overall Metrics

| Precision | Recall | F1-Score |
|---|---|---|
| 0.87 | 0.87 | 0.87 |

### B.1 CNN Classification Report

| Class | Precision | Recall | F1-Score |
|---|---|---|---|
| glioma | 0.53 | 0.59 | 0.56 |
| meningioma | 0.56 | 0.13 | 0.21 |
| no tumor | 0.63 | 0.80 | 0.70 |
| pituitary | 0.53 | 0.69 | 0.60 |

### B.2 CNN Accuracy

| | |
|---|---|
| Training Accuracy | $54.71\%$ |
| Validation Accuracy | $57.06\%$ |

### B.3 CNN Overall Metrics

| Precision | Recall | F1-Score |
|---|---|---|
| 0.56 | 0.55 | 0.53 |