Convolutional Neural Network for Brain Tumor Detection and Diagnosis

Zifan Luo
Department of Cognitive Science
University of California, San Diego
z9luo@ucsd.edu

Abstract

Brain tumor detection is an important task in medical imaging as early diagnosis significantly improves treatment outcomes. This project explores the application of a **Convolutional Neural Network (CNN)** implemented in **PyTorch** for classifying brain tumors from MRI images. The model is trained on a labeled dataset, utilizing multiple convolutional layers to extract spatial features, followed by fully connected layers for classification. Various preprocessing techniques, including data augmentation and normalization, are employed to enhance model generalization. The performance of the model is evaluated using accuracy, precision, recall, and F1-score metrics. The results demonstrate the efficacy of CNNs in automating brain tumor detection, with promising accuracy levels that could contribute to medical decision-making.

1 Introduction

Brain tumors are one of the most serious neurological conditions, requiring early and accurate diagnosis for effective treatment. Traditional methods such as MRI scans and manual radiological analysis are often time-consuming and prone to human error. **Deep Learning** has emerged as a powerful tool in the field of medical imaging and has shown great potential in aiding the health community in the detection and diagnosis of brain tumors. By leveraging deep learning algorithms, we can analyze medical images, such as MRI or CT scans, with unprecedented accuracy and efficiency. Also, it can assist in the classification of brain tumors into different subtypes such as gliomas, meningiomas, or metastatic tumors. Overall, deep learning has the potential to revolutionize brain tumor detection and diagnosis. By leveraging the power of neural networks, we can enhance the accuracy, efficiency, and understanding of brain tumor imaging, ultimately leading to improved patient care and outcomes in the field of neuro-oncology.

2 Data Preprocessing

Dataset used in this project was sourced from Kaggle's Brain Tumor Dataset[1]. It consists of MRI scan images categorized into two classes as tumorous(2426 images) and non-tumorous(2080 images).

2.1 Data Spliting

Before feeding the images into the model, we split the dataset into training and validation sets to ensure proper evaluation and avoid data leakage. The original dataset is split into 80% training data and 20% validation data. This ensures that the model is trained on one subset of images while being evaluated on a separate subset.

2.2 Data Transformation

Effective preprocessing and augmentation of image data are important for improving model performance and ensuring generalizability. To ensure consistency in input dimensions, all images are resized to a fixed resolution of 256x256 pixels so that they could better integrated into deep learning architectures that require uniform input sizes. Beyond resizing, the dataset underwent a series of augmentation techniques designed to improve the model's robustness to variations in tumor presentation. Horizontal and vertical flipping are applied randomly with a probability of 50%, introducing variability in image orientation and preventing the model from overfitting to specific spatial configurations. Furthermore, random rotations within a range of ± 30 degrees are incorporated to simulate natural variations in tumor positioning and imaging conditions. After augmentation, images are converted into a numerical format suitable for computational processing. To optimize the training process, pixel values are normalized using mean values of 0.485, 0.456, and 0.406, along with standard deviation values of 0.229, 0.224, and 0.225 for the red, green, and blue (RGB) channels, respectively. These values are derived from the ImageNet dataset[2], which serves as the benchmark for many pretrained models.Once the transformation pipeline is defined, it is consistently applied to both the training and validation datasets.

Figure 1 below gives a peak of what do tumor-affected and non-tumor brains look like from the training dataset.The tumor brain images shows regions of abnormal intensity and asymmetry, while non-tumor brain images shows relatively uniform and symmetrical structures with no abnormal mass.

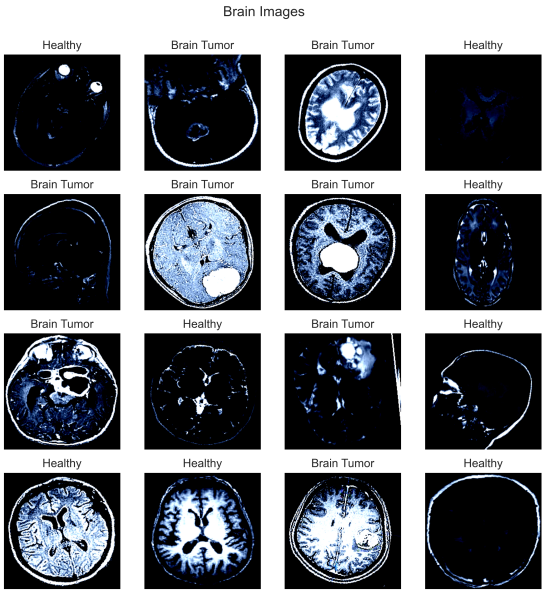

Figure 1: Examples of an augmented brain tumor images from training dataset.

3 Method

3.1 Model Architecture

After data pre-processing, we were able to input the data into our CNNs. The default CNN model(CNN_TUMOR) used in this project consists of four sequential convolutional layers, each followed by a max-pooling operation to progressively reduce spatial dimensions while preserving essential feature representations. The model is initialized with an input shape of $(3, 256, 256)$, where the three channels correspond to RGB components. Each convolutional layer utilizes a **3×3 kernel size**, progressively doubling the number of filters at each stage. The first convolutional layer begins with **8 filters**, and subsequent layers expand this to **16, 32, and 64 filters**, respectively. A non-linear activation function, **ReLU (Rectified Linear Unit)**, is applied after each convolution operation to introduce non-linearity and improve feature learning. Max-pooling with a **2×2 kernel** is employed after each convolution to downsample the feature maps, thereby reducing computational complexity and improving translation invariance.

To compute the dimensions of the feature maps at each stage, the function *findConv2dOutShape()* was used to dynamically determines the output size after each convolutional operation, ensuring compatibility with fully connected layers.

The number of neurons in the final flattened layer is computed as:

$$H_{\text{out}} = \left\lfloor \frac{H_{\text{in}} + 2P - D(K-1) - 1}{S} + 1 \right\rfloor \tag{1}$$

$$W_{\text{out}} = \left\lfloor \frac{W_{\text{in}} + 2P - D(K-1) - 1}{S} + 1 \right\rfloor \tag{2}$$

where $H_{\text{in}}, W_{\text{in}}$ are the input dimensions, P is the padding, D is the dilation rate, K is the kernel size, and S is the stride. Pooling operations further halve the output dimensions at each layer, reducing the total number of trainable parameters in the fully connected layers. The final classification is performed through two fully connected layers. The first fully connected layer contains **100 neurons**, followed by a **dropout layer (25% dropout rate)** to mitigate overfitting. The final layer outputs predictions for two classes (**tumor vs. non-tumor**) using a **softmax activation function**.This CNN model is trained using the **Negative Log-Likelihood Loss (NLLLoss)**. This loss function computes the likelihood that a given sample belongs to the correct class and penalizes incorrect classifications logarithmically. The loss function is defined as:

$$\mathcal{L} = -\sum_{i=1}^{N} y_i \log(\hat{y}_i) \tag{3}$$

where y_i represents the true class label (one-hot encoded), and $\hat{y}_i$ is the predicted probability of that class. To optimize the network, the **Adam optimizer** is employed with a **learning rate of** 3×10^{-4}.

3.2 Model Optimization

At the same time, another model(ResNet_TUMOR) with a deeper architecture, **ResNet18**, is incorporated into this project to assess whether leveraging **pretrained weights** and **residual connections** can enhance classification performance. Unlike the default CNN model as defined above, which is trained from scratch, **ResNet18 is initialized with pretrained weights from ImageNet**, allowing for **transfer learning** to accelerate convergence and improve generalization.The optimization process involves careful selection of textbfloss functions, optimizers, and learning rate schedules to ensure that both models achieve optimal performance. The **Adam optimizer** is used for the default CNN, whereas **Stochastic Gradient Descent (SGD) with momentum = 9** is selected for ResNet18. Additionally, a **ReduceLROnPlateau scheduler** is applied to dynamically adjust the learning rate based on validation loss trends. Both models use the same loss function.

After training, the models are evaluated based on training accuracy, validation accuracy, and loss curves. The performance metrics are plotted to visualize how each model converges over epochs, providing insights into which architecture is more suitable for brain tumor classification.

4 Experiment

4.1 Model Training

Both models were trained for **20 epochs** using the same training and validation dataset splits to ensure a fair comparison. For each epoch, the training loop consisted of iterating over mini-batches of input images and their corresponding labels. During each iteration, the model performed a **forward pass** to generate predictions, followed by computation of the **Negative Log-Likelihood Loss (NLLLoss)**. This loss was then used to perform a **backward pass (backpropagation)** to compute gradients, after which model weights were updated using the designated optimizer.

4.2 Model Evaluations

To evaluate and compare the effectiveness of the two architectures, I tracked their training and validation accuracy, as well as the corresponding loss values over 20 epochs. This allowed me to assess not only their final classification performance but also their convergence behavior and potential for overfitting.The **CNN_TUMOR** model showed a steady improvement in both training and validation accuracy throughout the training process. Starting with a validation accuracy of 71.23% in the first epoch, it gradually improved to 92.62% by epoch 20. The training accuracy followed a similar upward trend, increasing from 64.99% to 93.07%. Correspondingly, the validation loss decreased from 8.80 to 2.73, indicating improved generalization. However, the gap between training and validation loss in the later epochs suggests some degree of overfitting, which could potentially be mitigated by increased regularization (e.g., higher dropout rate) or early stopping. In contrast, the **ResNet_TUMOR** model exhibited significantly faster convergence and stronger generalization. The model achieved a validation accuracy of 92.62% as early as epoch 4 and continued to improve up to 98.91% by epoch 18, outperforming the custom CNN at every stage. The training accuracy reached 98.78% by epoch 20, and the corresponding validation loss dropped from 7.15 in epoch 1 to just 0.68 in epoch 19. This rapid improvement can be attributed to the residual connections and pretrained feature representations inherent to the ResNet18 architecture.

4.2.1 Confusion Matrix

In medical setting, it is also very important to look at the **Recall** of the model as missing a tumor (false negative) can have severe consequences, potentially leading to delayed treatment or misdiagnosis. Recall is defined as:

$$\text{Recall} = \frac{\text{True Positives (TP)}}{\text{True Positives (TP)} + \text{False Negatives (FN)}} \tag{4}$$

where:

- **True Positives (TP)**: Tumor cases correctly predicted as 'Tumor'.

- **False Negatives (FN)**: Tumor cases wrongly predicted as 'No Tumor' (missed cases).

A **high recall value** indicates that the model correctly detects most of the tumor cases, minimizing the risk of missing a critical diagnosis. As we can tell from the figure2 and figure3 below, the CNN_TUMOR model achieved a recall of **0.926** and the ResNet_TUMOR model outperformed it with a recall of **0.981**, suggesting a significantly higher ability to detect tumors. In clinical applications, where minimizing false negatives is paramount, the superior recall of ResNet18 makes it a more reliable candidate for deployment in diagnostic support systems. In this study, the ResNet18 model achieved an **F1 score** of 0.99 for both healthy and tumor classes. In contrast, the CNN_TUMOR model, while still performing reasonably well with an F1 score of 0.92 for the tumor class, showed a higher degree of misclassification. The high F1 score also validates the conclusion that the ResNet18 model outperforms the custom CNN architecture. Since the F1 score harmonizes both precision and recall, a near-perfect value indicates that the model is not only highly sensitive in detecting tumors but also highly precise in avoiding false positives.

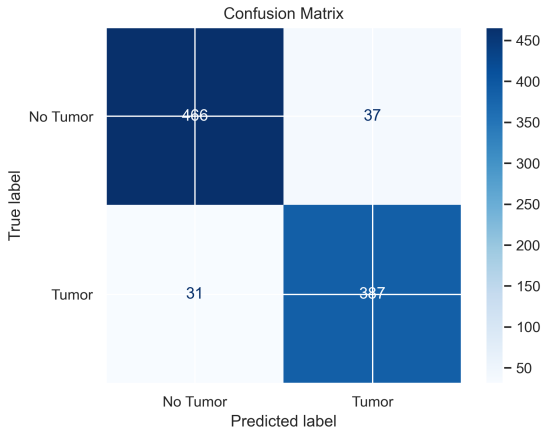

Figure 2: Confusion Matrix for CNN_TUMOR Model

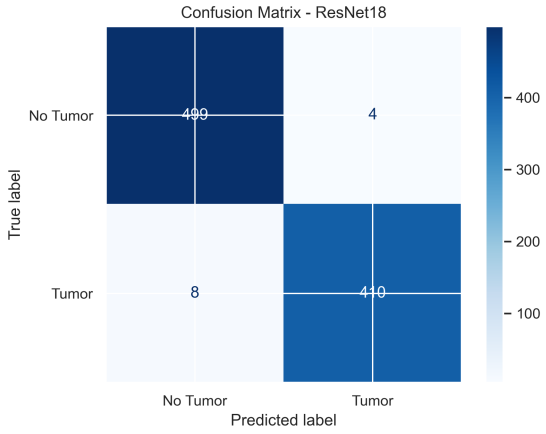

Figure 3: Confusion Matrix for ResNet_TUMOR Model

4.2.2 ROC Curve

A model that performs well will have a curve that closely hugs the top-left corner of the plot, indicating a high true positive rate with a low false positive rate across thresholds.As shown in figure 4, the ResNet18 model significantly outperforms the baseline CNN architecture in terms of its ROC curve profile. The **Area Under the Curve (AUC)** for the CNN_TUMOR model is 0.978, indicating a strong ability to distinguish between tumor and non-tumor cases. However, the ResNet18 model achieves an AUC of 0.998, which is nearly perfect. This substantial improvement demonstrates that the ResNet18 model is not only more accurate overall but also more reliable across different decision thresholds.

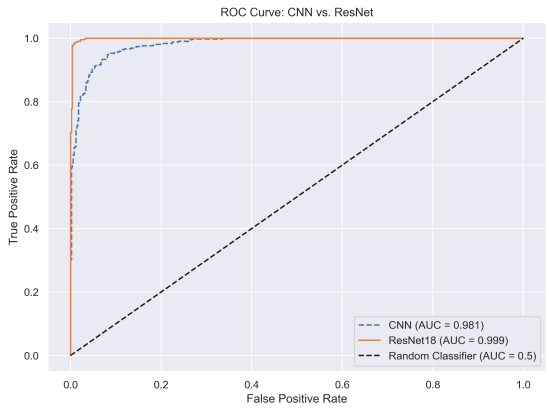

Figure 4: ROC curve

5 Conclusion

This project examines the application of **convolutional neural networks (CNNs)** for the binary classification of brain MRI images into tumor and non-tumor categories. A custom-built CNN architecture was developed and evaluated against a pretrained ResNet18 model to assess differences in performance, generalization, and clinical reliability. Through a comprehensive series of experiments—including confusion matrix analysis, ROC curve evaluation-the ResNet18 model consistently demonstrated superior performance across all evaluation criteria. Overall, the results validate the effectiveness of using deeper, pretrained models like ResNet18 for medical image classification. The findings not only demonstrate the advantages of transfer learning but also highlight the practical significance of precision-recall trade-offs in diagnostic settings. Future work may focus on further optimizing the model through explainability tools such as Grad-CAM, expanding the dataset with additional modalities, or adapting the approach for multi-class tumor classification[3] to enhance its applicability in real-world clinical workflows.

References

[1] Viradiya, P. (n.d.). Brain tumor dataset. Kaggle. Retrieved March 20, 2025, from `https://www.kaggle.com/datasets/preetviradiya/brian-tumor-dataset`

[2] Corley, I., Robinson, C., Dodhia, R., Lavista Ferres, J. M., & Najafirad, P. (2024). *Revisiting pre-trained remote sensing model benchmarks: Resizing and normalization matters*. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)* (pp. 3162-3172). Retrieved from `https://openaccess.thecvf.com/content/CVPR2024W/PBVS/papers/Corley_Revisiting_Pre-trained_Remote_Sensing_Model_Benchmarks_Resizing_and_Normalization_Matters_CVPRW_2024_paper.pdf`

[3] Krishnan, R., P G, G., Sujith, G., T, A., Abhishek, S. (2024a). Enhancing brain tumor diagnosis: A CNN-based multi-class classification approach. 2024 IEEE International Conference on Interdisciplinary Approaches in Technology and Management for Social Innovation (IATMSI), 1–6. `https://doi.org/10.1109/iatmsi60426.2024.10503231`