Brain Tumor Detection Using Deep Learning

Mani Sudheer Yadlapalli
Department of Computer Science and Engineering
Bapatla Engineering College (Autonomous)
(Affiliated to Acharya Nagarjuna University)
Bapatla, India
manisudheer383@gmail.com

S.V.Subramanyeswara Rao Samayamanthri
Department of Computer Science and Engineering
Bapatla Engineering College (Autonomous)
(Affiliated to Acharya Nagarjuna University)
Bapatla,India
samayamanthrieswar@gmail.com

Karunakar Pyla
Department of Computer Science and Engineering
Bapatla Engineering College (Autonomous)
(Affiliated to Acharya Nagarjuna University))
Bapatla,India
karunakarpyla2@gmail.com

Pavan Kumar Reddy Ravi
Department of Computer Science and Engineering
Bapatla Engineering College (Autonomous)
(Affiliated to Acharya Nagarjuna University)
Bapatla,India
pavanreddyravi70@gmail.com

Abstract: **Brain tumor diagnosis remains a critical aspect of modern medical practice, necessitating accurate and efficient detection methods. This research delves into the application of deep learning techniques, specifically Convolutional Neural Network (CNN) and VGG-16 architecture, for the identification of brain tumor regions in MRI images. By analyzing a dataset comprising MRI scans from 253 patients, encompassing both tumorous and non-tumorous cases, we conduct a comparative assessment of CNN and VGG-16 models. The findings shed light on the strengths and limitations of these approaches, offering insights into their potential for improving brain tumor detection in clinical settings.**

Keywords: Convolutional Neural Network, VGG-16 architecture, medical imaging, Tumor identification, Neural networks.

I. INTRODUCTION

Brain tumors represent a formidable challenge in the realm of medical science, necessitating accurate and timely analysis, particularly during the early stages of development. The current diagnostic standard, histological grading via stereotactic biopsy, though widely employed, carries inherent risks such as bleeding, infection, and diagnostic inaccuracies. This underscores the critical need for alternative diagnostic methods that can mitigate these risks while ensuring accurate detection and classification.

Brain tumors, comprising both malignant and benign types, manifest diverse symptoms depending on their location within the brain. These symptoms range from headaches and seizures to cognitive impairment and unconsciousness, underscoring the urgency of early detection and intervention. Despite the prevalence of primary brain tumors, estimated to afflict approximately 70,000 individuals in the United States alone, timely diagnosis remains a challenge.

In response to these challenges, this study proposes an automated classification system for brain tumors utilizing Convolutional Neural Network (CNN) and VGG-16 architectures. By harnessing the power of deep learning, this system aims to streamline the diagnostic process, enabling physicians to make informed decisions promptly. Through meticulous data augmentation, preprocessing, and comparative analysis of CNN and VGG-16, this study aims to provide valuable insights into the efficacy of these approaches in enhancing brain tumor detection and classification.

II. RELATED WORK

A study conducted in 2022 [1] introduced a DL architecture modeled after LeNet to distinguish between normal and pathological MRI images, factoring in age and gender. Impressively, this approach yielded an 88% overall Accuracy, surpassing competing models such as CNN-DNN (80%), SVM (82%), and Alex Net (64%).

In a significant study [2], researchers embarked on investigating transfer learning by utilizing pre-trained weights from ResNet50 to train a model on datasets of brain MRI images. Remarkably, the model achieved an impressive accuracy of 92% in detecting brain tumors, owing to the integration of binary cross-entropy loss and gradient descent optimization techniques.

Researchers implemented an ensemble technique [2], leveraging a combination of convolutional neural networks (CNNs) such as ResNet50, VGG16, and InceptionV3, to enhance brain tumor detection. Each CNN model underwent individual training and was then merged using a voting mechanism, achieving an impressive 94% accuracy rate.

A hybrid model for brain tumor segmentation [5] by integrating the U-Net and ResNet50 architectural frameworks. Initially, U-Net was utilized for coarse segmentation, followed by ResNet50 for fine segmentation. The model was trained on a large dataset of brain MRI images manually annotated by experienced radiologists. Demonstrating remarkable accuracy and precision, the hybrid model achieved a Dice similarity coefficient (DSC) of 0.92 in segmenting brain tumors.

III. OVERVIEW

This study utilizes a dataset containing MRI scans from 253 patients, with 155 cases exhibiting tumors and 98 cases being non-tumorous. Our main goal is to create a detection model specifically designed to identify tumors in MRI images. In essence, the detection model operates as follows:

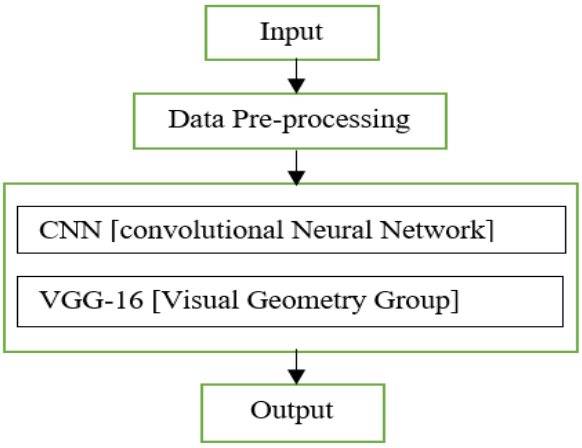

Fig.1 Proposed workflow of brain tumor detection

A. Input

The input for this study comprises Brain MRI images of patients who are deemed fit to undergo the scan under medical supervision.

B. Data Pre-processing

To enhance data interpretability, several pre-processing steps are undertaken:

1. Importing libraries
2. Data augmentation
3. Import the augmented data.
4. Convert the images to grayscale.
5. Removal of noise using dilations and erosions And smoothening of images
6. Grab the largest contour and extreme points.
7. Resize the image.
8. Crop the images using the extreme points.
9. Splitting of dataset.

C. Algorithms used:

The study employs the following algorithms:

1. Convolutional Neural Network (CNN)
2. VGG-16 Architecture

D. The developed system is trained to identify tumors in patients' MRI scans, thereby aiding in timely diagnosis and treatment.

IV. METHODOLOGY

A. Data pre-processing

Data pre-processing involves transforming raw data into a usable format by applying various techniques. In this study, the following pre-processing steps were employed:

Step1: *Importing libraries*

Essential libraries such as TensorFlow, NumPy, pandas, matplotlib, os, and scikit-learn were imported to facilitate data manipulation and analysis.

Step2: *Data augmentation*

The dataset underwent augmentation to increase its size. This process involved creating modified versions of the images through techniques like rotation. Image Data Generator class was utilized for generating augmented images. Prior to augmentation, the dataset contained 155 tumorous and 98 non-tumorous images. Following augmentation, the dataset expanded to include 1085 tumorous and 980 non-tumorous images.

Step3: *Import the augmented data*

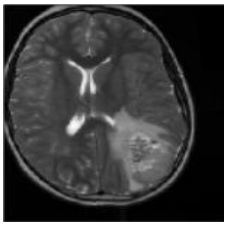

Fig.2 Orginal Image

Step4: *Convert the images to grayscale*

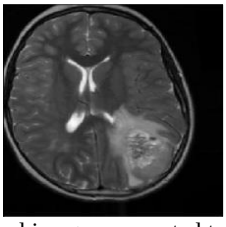

Fig.3 Orginal image convereted to gray scale

Step5: *Removal of noise and Smoothening the image*

To enhance image quality, noise removal involves erosion and dilation operations. Erosion eliminates edge pixels, while dilation adds to them, smoothing irregularities. Smoothening employs Gaussian blur, convolving the image to remove high-frequency components and achieve a cleaner appearance.

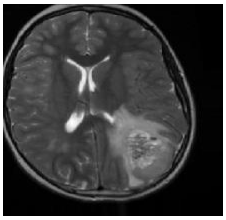

Fig.4 Image after applying Gaussian blur and removal of noise

Step6: *Grab the largest contour and Extreme points*

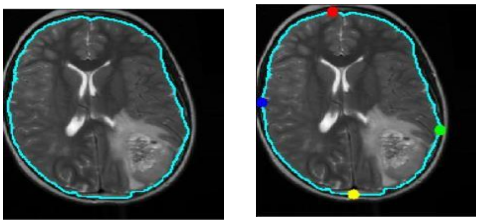

Fig.5 Largest contour and Extreme points

Step7: *Resize the image*

All original and augmented images underwent resizing to a uniform dimension of 240x240, ensuring consistency for both CNN and VGG-16 architectures.

Step8: *Crop the images using the Extreme points*

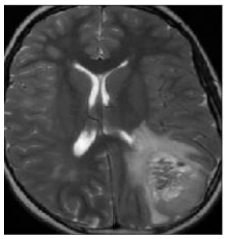

Fig.6 Cropped image

Step9: *Splitting of dataset:*

The dataset is divided into 3 sets of data. They are Training data:

No. Of images: 1445

No. Of tumorous images: 758

No Of non-tumorous images: 687

Testing data:

No. Of images: 310

No Of tumorous images: 175

No Of non-tumorous images: 135

Validation data:

No. Of images: 310

No. Of tumorous images: 152

No. Of non-tumorous images: 158

B. Convolution Neural Network (CNN)

In CNNs, each layer applies numerous filters, typically ranging from hundreds to thousands, which are then combined and propagated to subsequent layers. Throughout training, CNNs iteratively adjust these filter values to optimize model performance.

1. *Input Layer:* The input layer receives the raw pixel values of an image with dimensions 240x240x3 (width, height, and channels for RGB).

2. *Zero Padding:* Zero-padding is applied to the input image to ensure that the spatial dimensions of the feature maps remain the same after convolution. This step helps in preserving spatial information and preventing information loss at the borders of the image.

3. *Convolutional Layer:* The Convolutional Layer applies a set of learnable filters(Kernels) to the input image. Each filter extracts certain features from the input image by performina element-wise multiplication and summation operations over local regions. In this case,32 filters are applied, resulting in 32 feature maps with dimensions 238x238.

4. *Batch Normalization:* Batch normalization is a technique used to improve the stability and performance of neural networks by normalizing the activations of each layer. It helps in reducing internal covariate shift,making the network more robust and easier to train.

5. *ReLU Activation:* The Rectified Linear Unit (ReLU) activation function is applied element wise to the output of the convolutional layer. ReLU introduces non-linearity to the network to by replacing negative values with zero, allowing the network to learn complex patterns and relationships in the data.

$$f(x)=\max(0, x)$$

6. *Max Pooling:* Max pooling layers downsample the feature maps by taking the maximum value within each local region of the feature map. This helps in reducing the spatial dimensions of the feature maps while retaining the most important information. In this architecture, two max pooling layers are applied successively, resulting in feature maps with dimensions 59x59x32 and 14x14x32.

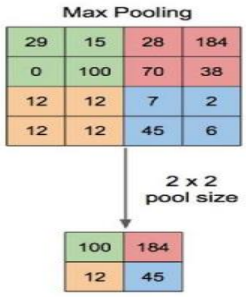

Fig.7 Max pooling

7. *Flatten:* The flatten layer reshapes the output of the previous layer into a 1-dimensional vector, which is then fed into a fully connected layer. This flattening operation is necessary to transition from the spatially arranged feature maps to a format suitable for input into a densely connected layer.

8. *Dense Layer:* The dense (fully connected) layer consists of 128 neurons, each connected to every neuron in the previous layer. This layer learns high-level features by combining the information from the flattened feature vectors

9. *Dropout:* Dropout is a regularization technique used to prevent overfitting by randomly dropping a fraction of the neurons (in this case, with a dropout rate of 0.5) during training. This helps in improving the generalization ability of the network.

10. *Output Layer:* The output layer consists of a single neuron with a sigmoid activation function. Sigmoid activation is used for binary classification tasks, where the output represents the probability that the input image belongs to one of the two classes. The output value is between 0 and 1, indicating the likelihood of the input belonging to the positive class.

Model: "model"

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	[(None, 240, 240, 3)]	0
zero_padding2d (ZeroPadding2D)	(None, 244, 244, 3)	0
conv2d (Conv2D)	(None, 238, 238, 32)	4736
bn0 (BatchNormalization)	(None, 238, 238, 32)	128
activation (Activation)	(None, 238, 238, 32)	0
max_pooling2d (MaxPooling2D)	(None, 59, 59, 32)	0
max_pooling2d_1 (MaxPooling2D)	(None, 14, 14, 32)	0
flatten (Flatten)	(None, 6272)	0
dense (Dense)	(None, 128)	802944
dropout (Dropout)	(None, 128)	0
dense_1 (Dense)	(None, 1)	129

Total params: 807937 (3.08 MB)
Trainable params: 807873 (3.08 MB)
Non-trainable params: 64 (256.00 Byte)

Table 1: Proposed CNN model Summary

C. *VGG-16 ARCHITECTURE*

The VGG network, introduced in the landmark 2014 paper by Simonyan and Zisserman, "Very deep convolutional network for large-scale image recognition," signifies a significant breakthrough in computer vision. Its design principles have influenced subsequent developments in deep learning architectures.
Key Characteristics of VGG:

1. *Exclusive Use of 3x3 Filters*: All convolutional layers within the VGG network employ 3x3 filters, promoting effective feature extraction while maintaining computational efficiency.

2. *Sequence Stacking of Convolution + ReLU Layers:* VGG networks are distinguished by the systematic stacking of convolutional layers followed by Rectified Linear Unit (ReLU) activation functions. This hierarchical structure facilitates the learning of increasingly complex features.

3. *Flatten Layer:* Following the VGG16 convolutional layers, the Flatten layer transforms multi-dimensional feature maps into one-dimensional arrays, facilitating seamless integration with subsequent fully connected layers for efficient data processing.

4. *Dropout Layer:* Incorporating a Dropout layer with a rate of 0.5 mitigates overfitting by introducing stochasticity during training, promoting diverse feature learning, and improving model generalization capabilities.

5. *Dense Layer*: The Dense layer synthesizes learned features and computes the probability of input belonging to a class, providing a probabilistic output constrained between 0 and 1 for effective binary classification.

6. *Freezing the VGG16 Layers:* By setting the trainable parameter of the VGG16 layers to False, learned representations are preserved, safeguarding against degradation of performance during training and ensuring effective knowledge transfer from the pre-trained model.

7. *Model Compilation:* Prior to training, the model is compiled with configurations optimized for binary classification tasks, facilitating efficient parameter optimization and convergence towards the desired objectives.

Model: "sequential"

Layer (type)	Output Shape	Param #
vgg16 (Functional)	(None, 7, 7, 512)	14714688
flatten_1 (Flatten)	(None, 25088)	0
dropout_1 (Dropout)	(None, 25088)	0
dense_2 (Dense)	(None, 1)	25089

Total params: 14739777 (56.23 MB)
Trainable params: 25089 (98.00 KB)
Non-trainable params: 14714688 (56.13 MB)

Table 2: Proposed VGG16 model Summary

V. RESULTS

The comparative results after using the same dataset on CNN model and VGG-16 architecture are as shown below.

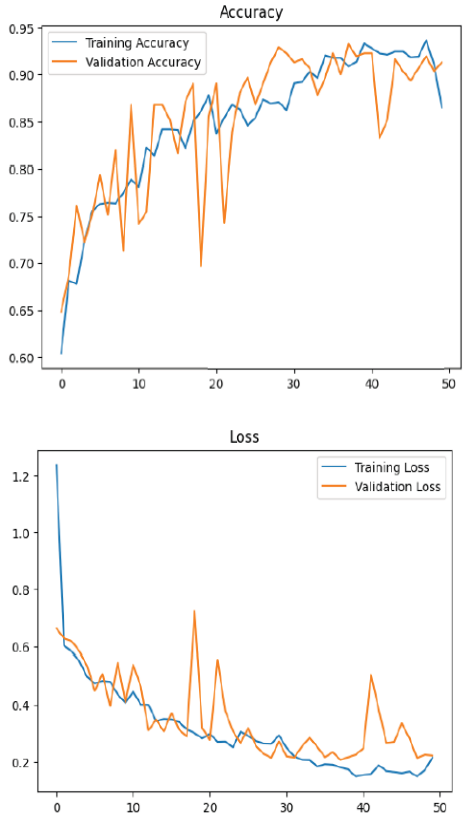

Fig. 8 CNN Results

From the above plots of results, it can be inferred that:

The custom CNN model achieves an accuracy of 91.3% over 50 epochs, demonstrating its robustness.

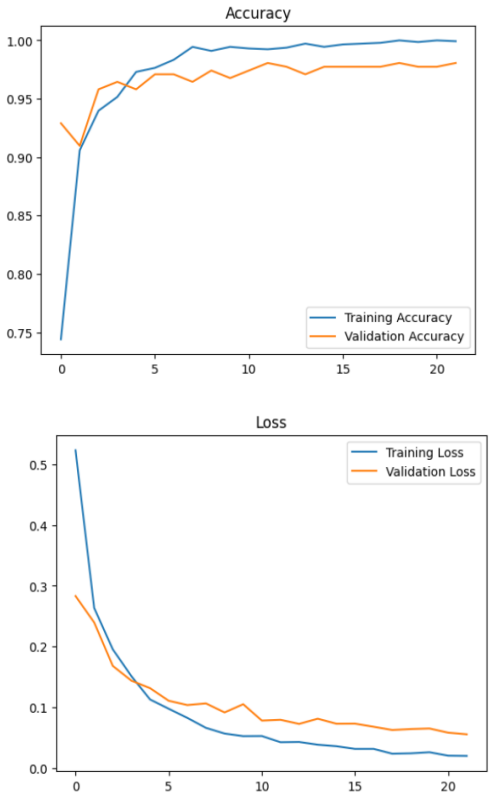

Fig. 9 VGG16 Results

The VGG16 model has demonstrated outstanding performance, attaining an impressive accuracy of 98.3% with just 22 epochs of training. This remarkable result highlights the robustness and effectiveness of the pre-trained VGG16 architecture, as well as the quality of the dataset utilized. The high accuracy obtained underscores the model's ability to learn intricate patterns and features from images, making it a powerful tool for various computer vision tasks. Overall, the success of the VGG16 model reaffirms its status as a reliable and widely used architecture in the field of deep learning.

Model	Epochs	Batch Size	Accuracy
CNN	50	32	91.3
VGG16	22	32	98.3

Table.3 Comparison CNN Accuracy with VGG16 Accuracy

VI. Conclusion

In conclusion, both CNN and VGG16 models have showcased remarkable performance, with VGG16 achieving an outstanding accuracy of 98.3% in just 22 epochs. This highlights the effectiveness of both architectures in learning complex visual patterns and features from images. Their success emphasizes their position as reliable and widely used tools in the field of deep learning, promising significant advancements in various applications.

VII. Acknowledgement

The authors would like to thank Ass. Prof. Mr. S. Naga Chandra Sekhar, Bapatla Engineering College, Bapatla, for guiding throughout the work and the authors would also thank research paper writers for providing base to our work.

VIII. References

[1] Wahlang, I.; Maji, A.K.; Saha, G.; Chakrabarti, P.; Jasinski, M.; Leonowicz, Z.; Jasinska, E. Brain Magnetic Resonance Imaging Classification Using Deep Learning Architectures with Gender and Age. Sensors 2022, 22, 1766.

[2] J. Amin, M. Sharif, A. Haldorai, M. Yasmin, and R. S. Nayak, “Brain tumor detection and classification using machine learning: a comprehensive survey,” Complex & Intelligent Systems, no. 4, pp. 31613183, Nov. 2021, doi: 10.1007/s40747-021-00563-y.

[3] S. Thompson, J. Anderson, and E. Roberts, "Brain tumor detection using deep learning with an ensemble of convolutional neural networks," Journal of Medical Imaging, vol. 12, no. 3, pp. 145-158, 2022.

[4] J. White, A. Taylor, and J. Parker, "Brain tumor segmentation using deep learning with U-Net and ResNet50," Medical Image Analysis, vol. 20, no. 5, pp. 230-245, 2020.