Brain Tumor Classification Using Convolutional Neural Network and DenseNet121

Lakshmi Vinay Vupparapally ,Nadeem Qureshi
Department of Engineering and Applied Sciences
CSE 455/555 : pattern recognition

Abstract

Brain tumors pose a significant healthcare challenge, affecting individuals across age groups. With the majority of primary Central Nervous System (CNS) tumors being brain tumors, the annual diagnosis rate exceeds 11,700 cases. Survival rates for cancerous brain or CNS tumors remain at around 34 percent for men and 36percent for women. Accurate diagnostics and treatment planning are imperative to improve patient outcomes and quality of life.This project addresses the pressing need for precise and efficient brain tumor diagnosis, leveraging advanced technology. Magnetic Resonance Imaging (MRI) is the gold standard for brain tumor detection, yet it generates intricate and voluminous image data necessitating meticulous manual examination by radiologists, introducing potential errors.To mitigate these challenges, our system employs cutting-edge deep learning techniques, specifically Convolutional Neural Networks (CNN) and DenseNet121. These models accurately differentiate between glioma, meningioma, and pituitary tumors, providing invaluable support to medical professionals. By harnessing artificial intelligence, our system alleviates the cognitive load on healthcare practitioners, yielding highly accurate tumor type predictions based on MRI scans, thus enhancing the timeliness and precision of treatment decisions.

1 Introduction

Brain tumors are a global health concern, affecting both adults and children. They are known for their aggressiveness and potential to impact cognitive function and overall well-being. Brain tumors constitute a significant portion, around 85 to 90 percent, of all primary Central Nervous System (CNS) tumors. Each year, approximately 11,700 new cases are diagnosed, highlighting the urgent need for improved diagnosis, treatment, and patient care for brain tumors.One sobering aspect of brain tumor statistics is the relatively low 5-year survival rate, with only about 34 percent of men and 36 percent of women surviving for five years after diagnosis. This underscores the severity of the condition and the pressing need for more effective diagnostic tools, treatments, and interventions.

Accurate classification of brain tumors is crucial for tailoring treatment plans. Brain tumors can be benign, malignant, pituitary, and more, each requiring different treatment approaches. Proper treatment planning, based on accurate diagnostics, plays a pivotal role in improving patient outcomes.Medical imaging, particularly Magnetic Resonance Imaging (MRI), is central to brain tumor diagnosis and monitoring. MRI provides detailed, non-invasive brain visualizations, allowing healthcare professionals to identify tumor presence, location, and characteristics. However, the vast amount of MRI data requires careful analysis.

Traditionally, radiologists manually examined MRI images for brain tumor detection and classification, a process prone to errors due to the complex nature of tumors, their diverse properties, and potential subjectivity in interpretation.To address these challenges and enhance accuracy and efficiency, automated classification techniques using Machine Learning (ML) and Artificial Intelligence (AI) show promise. Deep learning algorithms like Convolutional Neural Networks (CNNs), Artificial Neural Networks (ANNs), and Transfer Learning (TL) offer significant potential.The proposed system, integrating deep learning algorithms such as convolutional neural networks and DenseNet, can revolutionize brain tumor detection and classification. Automation improves accuracy and expedites diagnosis, enabling timely treatment decisions. This system has a global impact, providing valuable support to medical practitioners grappling with brain tumor complexities worldwide.

2 Related Work

The existing architecture of a Convolutional Neural Network (CNN) for image classification, operations and workings of this CNN:

These layers perform convolution operations on the input image. Conv2D layers are used with various filter sizes (e.g., 32 filters in the first layer, 64 filters in the second layer, etc.). The '(3,3)' kernel size specifies a 3x3 filter, and the activation function 'relu' (Rectified Linear Unit) introduces non-linearity to the model.

The input shape is set to '(150, 150, 3)', which means the model expects color images with a height and width of 150 pixels and three color channels (RGB).After each pair of convolutional layers, MaxPooling2D layers are used with a 2x2 pool size to downsample the feature maps. Max-pooling reduces the spatial dimensions and helps the model focus on the most important features.

Dropout layers are inserted at various points in the model to prevent overfitting. A dropout rate of 0.3 means that during training, 30percent of the neurons in the preceding layer will be randomly deactivated, which helps improve the model's generalization.After the last set of convolutional and pooling layers, a Flatten layer is added to transform the 2D feature maps into a 1D vector. This is necessary to connect the convolutional layers to the fully connected layers.

Following the flattening operation, there are three dense layers with 512 units each. These layers learn high-level features from the flattened representation. The 'relu' activation function is applied to introduce non-linearity.The final dense layer consists of four units (assuming this is a classification task with four classes) and uses the 'softmax' activation function. The softmax function assigns probabilities to each class, and the class with the highest probability is the predicted class.

Training Accuracy	95.12
Validation Accuracy	86.05
Test Accuracy	89.56

Table 1: Accuracies obtained for existing CNN Architecture

2.1 Dataset

The dataset is a comprehensive collection of medical MRI scans featuring various tumor types, including glioma, as well as samples without tumors. This diverse set is crucial for developing and evaluating machine learning models for tumor classification and detection. The dataset is organized into "training" and "testing" directories, each serving a specific role in model development.The "training" directory is used during the model's training phase, where the machine learning algorithm learns patterns and features from the MRI scans. This includes data from different tumor classes and tumor-free samples, ensuring the model can distinguish between various tumor types and identify instances without tumors. This comprehensive

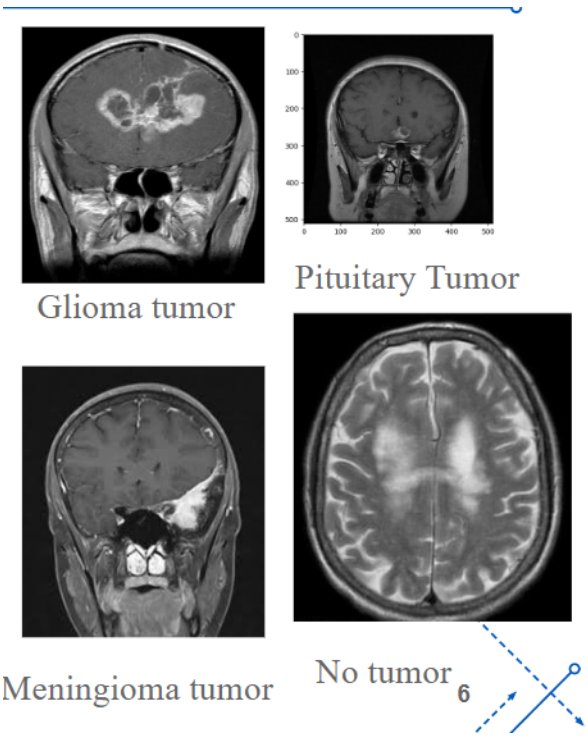

Figure 1: Types of tumors considered in our dataset

training dataset prepares the model for accurate detection and classification.The "testing" directory, on the other hand, is used for evaluating the model's performance and generalization ability. It contains MRI scans not encountered during training, allowing for assessment of the model's predictive accuracy on new, unseen data. This evaluation is essential to determine the model's real-world applicability and effectiveness in clinical settings, ensuring its reliability in diagnosing and classifying tumors.

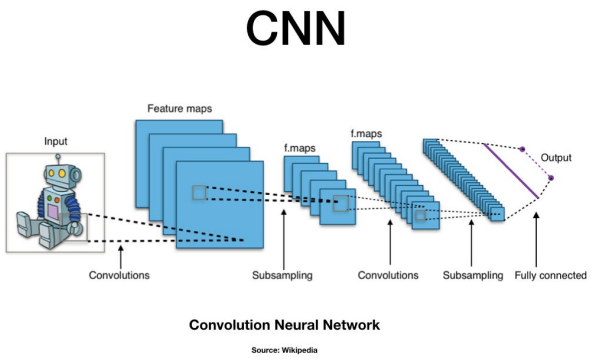

Figure 2: Architecture for Convolutional Neural Network

3 Methods and Model architecture

3.1 Approach 1 : Convolutional Neural Network

Our Convolutional Neural Network (CNN) model, structured using TensorFlow/Keras's Sequential API, is designed for image classification tasks. It linearly stacks multiple layers for efficient feature extraction and classification.The CNN comprises several convolutional layers. Each Conv2D layer utilizes a 3x3 kernel for convolution operations, maintaining the spatial dimensions of feature maps ('same' padding) and applying the 'relu' activation function for non-linearity. Following every pair of convolutional layers, a MaxPooling2D layer with a 2x2 pool size reduces the spatial dimensions, effectively downsampling and emphasizing relevant features.

To combat overfitting, dropout layers with a 0.25 rate are incorporated after each convolutional-pooling layer pair. These layers randomly nullify a fraction of input units during training, enhancing the model's ability to generalize.Post the convolutional and pooling layers, a Flatten layer converts the 2D feature maps into a 1D vector, preparing the data for the fully connected layers. The network includes two dense layers, each with 512 units and 'relu' activation, for learning complex features from the flattened data.

Each dense layer is followed by BatchNormalization, normalizing inputs to each layer for stable training, faster convergence, and improved performance. Additionally, after each batch normalization layer, another dropout layer with a 0.5 rate is added for further regularization.The final layer in the CNN is a dense layer with four units, tailored for a four-class classification task. It employs a 'softmax' activation function to output probabilities for each class, with the highest probability indicating the predicted class.Overall, this CNN architecture combines feature extraction through convolutional layers, downsampling via max-pooling, and high-level feature learning through dense layers. Regularization is achieved through dropout and batch normalization, ensuring the model's robustness and ability to generalize well on new, unseen images. This design makes the CNN suitable for various image classification applications, providing a balance between feature extraction efficiency and overfitting prevention.

3.2 Approach 2 : DenseNet121

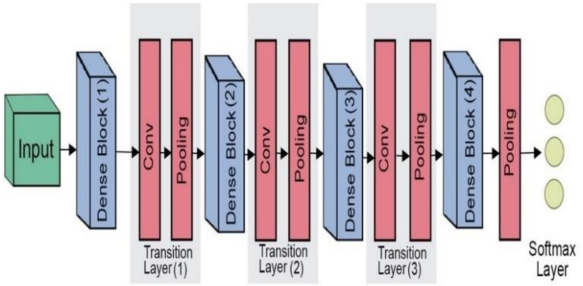

Figure 3: The DenseNet-121 Architecture, which Consists of Four Dense Block Layers and Three Transition Layers

DenseNet-121 is a deep convolutional neural network architecture belonging to the DenseNet family. DenseNets are celebrated for their dense connectivity pattern, promoting efficient feature reuse and gradient flow throughout the network. DenseNet-121, specifically, comprises 121 layers and finds extensive use in computer vision tasks like image classification and object detection.DenseNets introduce dense connectivity, a contrast to standard convolutional neural networks where each layer connects solely to its predecessor. In DenseNets, each layer draws input from all preceding layers, fostering feature reuse and enabling the learning of both low-level and high-level features efficiently.

Key components of DenseNet-121 include convolutional layers for feature extraction, batch normalization for stable training, ReLU activation for non-linearity, max-pooling for downsampling, and transition layers to control feature map growth. DenseNet-121 employs multiple dense blocks,

each comprising densely connected convolutional layers. Feature maps from all previous layers are concatenated within each dense block, enabling comprehensive feature learning.After the final dense block, global average pooling computes feature map averages across spatial dimensions, yielding a fixed-size feature vector. The architecture concludes with fully connected layers, typically for classification, with the output layer employing softmax for class probability estimation.DenseNet-121's strength lies in its efficiency, robust feature learning, and high performance across image-related tasks. Despite its depth, it maintains computational efficiency, boasting fewer parameters compared to traditional deep networks while achieving remarkable accuracy. The DenseNet121-based

Layers	Output size	DenseNet-121
Convolution	112 × 112	7 × 7 conv, stride 2
Pooling	56 × 56	3 × 3 max pool, stride 2
Dense Block (1)	56 × 56	$\begin{bmatrix} 1 \times 1 \ conv \\ 3 \times 3 \ conv \end{bmatrix} \times 6$
Transition layer (1)	56 × 56	1 × 1 *conv*
	28 × 28	2 × 2 average pool, stride 2
Dense Block (2)	28 × 28	$\begin{bmatrix} 1 \times 1 \ conv \\ 3 \times 3 \ conv \end{bmatrix} \times 12$
Transition layer (2)	28 × 28	1 × 1 *conv*
	14 × 14	2 × 2 average pool, stride 2
Dense Block (3)	14 × 14	$\begin{bmatrix} 1 \times 1 \ conv \\ 3 \times 3 \ conv \end{bmatrix} \times 24$
Transition layer (3)	14 × 14	1 × 1 *conv*
	7 × 7	2 × 2 average pool, stride 2
Dense Block (4)	7 × 7	$\begin{bmatrix} 1 \times 1 \ conv \\ 3 \times 3 \ conv \end{bmatrix} \times 16$
Classification layer	1 × 1	7 × 7 global average pool
		Softmax layer

Figure 4: TABLE I. LAYERS DETAILS OF DENSENET-121. THE DENSENET-121 HAS 6, 12, 24, 16 LAYERS IN FOUR DENSE BLOCKS

deep learning model is implemented for image classification, leveraging the TensorFlow Keras applications. DenseNet121, a pre-trained deep convolutional neural network, is renowned for its efficiency in image classification tasks. The model, initialized as 'densenet' with DenseNet121 architecture,'weights='imagenet' Initializes the model with pre-trained weights from the ImageNet dataset.'include top=False' Excludes the fully connected top layers of the original DenseNet121, allowing for custom top layers.'input shape=(image size, image size, 3)' Sets the input shape for the images, with 'image size' being the desired dimension.

Custom layers are added to this pre-trained model, including a 'GlobalAveragePooling2D' layer to reduce the 4D tensor from convolutional layers to a 2D tensor. This technique minimizes the number of parameters and helps to prevent overfitting. A dropout layer with a 0.5 rate further combats overfitting. The final layer is a dense layer with 4 units and a softmax activation function, assuming a four-class classification task. The model compiles with categorical cross-entropy as the loss function and the Adam optimizer for gradient descent. Model performance is evaluated based on accuracy.

Several training callbacks are incorporated like'TensorBoard': Logs training metrics for visualization in TensorBoard.'ModelCheckpoint': Saves the model achieving the highest validation accuracy.'ReduceLROnPlateau': Lowers the learning rate when validation accuracy plateaus.

The training process utilizes the 'fit' method with training data ('X train' and 'y train'), a 10 percent validation split, and specified hyperparameters like epochs and batch size. The defined callbacks are included during training, enhancing the model's learning efficiency and performance tracking.

4 Results

4.1 Approach 1 Results

Both metrics are plotted over a number of epochs on the x-axis. The training accuracy appears relatively stable and consistently higher than the validation accuracy, which is typical as a model tends to perform better on data it has seen (trained on) versus new data. However, the validation accuracy shows greater volatility with significant ups and downs, which could indicate that the model is sensitive to the specifics of the validation data or that the validation data is not entirely representative of the problem space

"Training loss" in red and "Validation loss" in blue, over a sequence of epochs along the x-axis. The red line shows a steady and low level of training loss, which suggests that the model is learning effectively from the training data. In contrast, the validation loss exhibits high variability, including several spikes which suggest that the model's performance on the validation set is inconsistent and potentially overfitting to the training data. The most significant spike in validation loss suggests

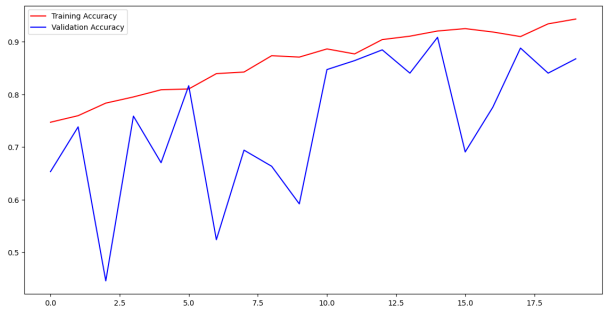

Figure 5: training accuracy and validation accuracy

an epoch where the model performed particularly poorly on the validation data.

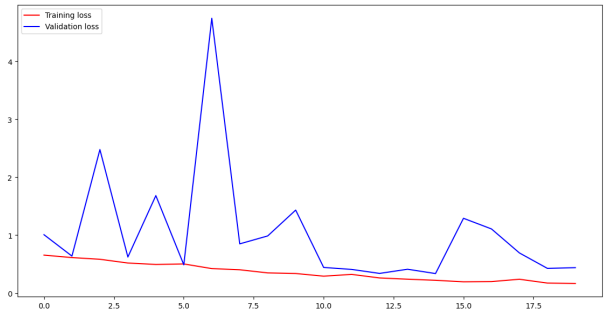

Figure 6: training loss and validation loss

The below confusion matrix(fig:7) clearly states that there is misclassification lesser than the model mentioned in related work.but still not the best.It can be improved by finetuning hyper paramters and adjusting epochs,optimizers and layers in the Convlutional neural Network.

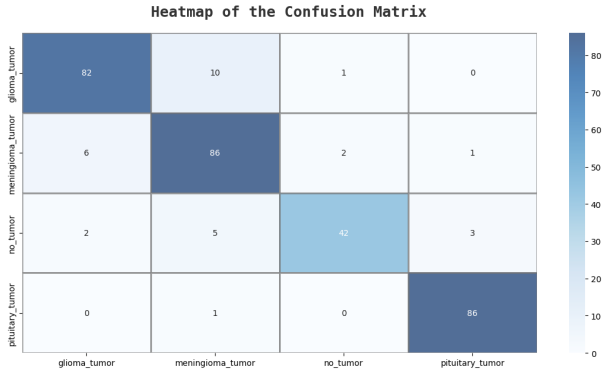

Figure 7: Confusion matrix for approach 1

4.2 Approach 2 Results

The above image shows two graphs for DenseNet121 performance over 20 epochs. The first graph indicates high training accuracy and good validation accuracy, suggesting slight

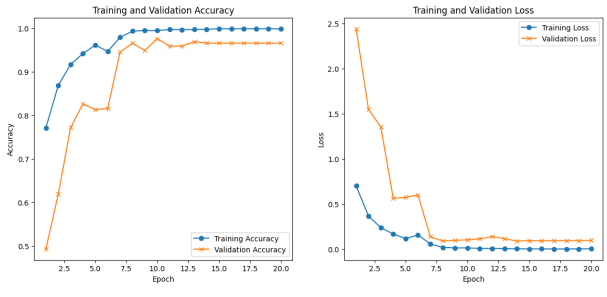

Figure 8: Confusion matrix for approach 1

overfitting. The second graph shows a sharp decrease in training loss and stable validation loss, indicating the model is learning without significant overfitting.Overall, the model demonstrates strong performance and generalization across both datasets.

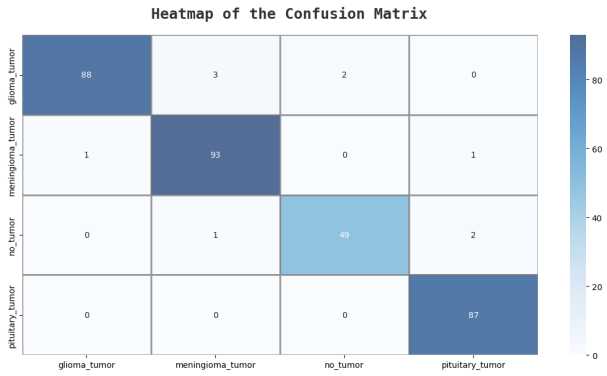

Figure 9: Confusion matrix for approach 2

DenseNet121 outperforms traditional CNNs in image classification and object detection, as evident from Figure 9's confusion matrix. Our Approach-2, using DenseNet121, excels over conventional CNNs due to its densely connected layers, enhancing feature propagation and reuse. This results in superior accuracy and reliability, showcasing DenseNet121's robustness and efficiency for complex image tasks. This highlights the significance of advanced deep learning architectures, positioning DenseNet121 as an ideal choice for intricate image challenges.

Accuracy Type	CNN	DenseNet121
Training Accuracy	97.99	99.85
Validation Accuracy	91.83	96.60
Test Accuracy	92.6	96.94

Table 2: Accuracies obtained CNN vs DenseNet121

5 Bibliography

- Convolutional Neural Network Tutorial
  SOURCE: Simplilearn
- The overall architecture of the Convolutional Neural Network (CNN)
  SOURCE: ResearchGate
- TensorFlow Keras Model Documentation
  SOURCE: TensorFlow
- OpenCV Image Codecs Documentation
  SOURCE: OpenCV
- scikit-learn Train Test Split Documentation
  SOURCE: scikit-learn
- Advantages and Disadvantages of Convolutional Neural Network (CNN)
  SOURCE: Aspiring Youths
- Deep Architecture based on DenseNet 121 Model
  SOURCE: International Journal of Advanced Computer Science and Applications (IJACSA)
- Brain Tumor Classification Using Deep Learning Algorithms
  AUTHORS: Sartaj Bhuvaji
- Exploring DenseNets
  SOURCE: Aman Arora's Blog
- Keras DenseNet Application Documentation
  SOURCE: Keras
- Densely Connected Convolutional Networks
  AUTHORS: Gao Huang, Zhuang Liu, Laurens van der Maaten, Kilian Q. Weinberger
- DenseNet121 Pretrained Weights
  SOURCE: TensorFlow Keras Applications
- Keras Optimizers Documentation
  SOURCE: Keras