IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Brain Tumor Detection Using Convolutional Neural Network

[1]R. S. Lavhe, [2]M. Shubham, [3]N. Ajinkya,[4]K. Vinay, [5]S. Dnyaneshwari

[1]Assistant Professor,[2,3,4,5]B.E Student

[1]Information Technology Engineering

[1]Anantrao Pawar College of Engineering and Research, Parvati Pune, India

Abstract ---Brain tumors are a serious medical issue that require prompt and precise diagnosis in order to improve patient outcomes. The use of Convolutional Neural Networks (CNNs) to automate the identification of brain cancers from medical pictures, such as MRI scans, is investigated in this research study. The suggested CNN model is suited for this challenging task since it makes use of its capacity to learn complicated visual features and patterns. To train and assess the CNN, the research technique uses a large dataset of brain MRI scans, comprising both tumor and non-tumor cases. By using methods like dropout and batch normalization, the CNN architecture is intended to prevent overfitting and capture pertinent geographical information. With pre-trained CNN models, transfer learning is used to improve the model's performance even with a small amount of training data. The outcomes of our experiment show that the CNN can successfully categorize brain pictures, with excellent recall, precision, and F1-score values. The CNN model shows promise for practical clinical applications and demonstrates strong generalization to data that has never been seen before. This study lays the groundwork for the creation of automated systems for the detection of brain tumors, which may help doctors identify these tumors early and improve patient outcomes. Moreover, it underscores the importance of deep learning methods—more especially, CNNs—in the area of medical picture interpretation and their possible wider use in the medical industry.

Keywords -Brain Tumor, Image Processing, Artificial Networks, Convolutional Neural Network

I. INTRODUCTION

Diagnostics of different medical disorders depend heavily on medical picture categorization, which is a demanding yet exciting area of image processing. Brain tumors are among the most prevalent and deadly types of cancer, with a high death rate. According to the International Organization for Research on Cancer (IARC), there have been almost a million brain tumor diagnoses. As their fatality rates rise, they rank second among cancer-related fatalities in people under the age of 34.[1] Significant technological advances in recent years have made tumor diagnosis more precise and non-invasive. A variety of anatomical regions can be examined with CT and MRI scans, which are both often employed. Because it is necessary to analyze large amounts of medical data quickly and objectively, which calls for specific computational tools, MRI-based medical image processing has drawn interest. [2] An important area of focus is automated MRI imagebased brain tumor detection. By streamlining diagnosis, this automation lessens the need for human data processing. Early and

accurate tumor diagnosis is made possible through automated analysis of MRI-based medical imaging, which ultimately improves patient care and outcomes by utilizing state-of-the-art technologies and computational approaches. This field of study has the potential to transform the diagnosis of brain tumors, highlighting the critical role that technology plays in contemporary healthcare. offered. These elements must be created by the format, which must also incorporate the following appropriate requirements.[3]

II. LITRATURE SURVEY

The urgent need to enhance patient outcomes and diagnostic accuracy has led to major advancements in the field of brain tumor detection utilizing medical imaging in recent years. In this review of the literature, we highlight a number of significant methods and strategies used in the effort to improve brain tumor detection and segmentation. Using the Euclidean distance classifier in conjunction with the Watershed algorithm was one method for detecting brain tumors [4]. Using ascending regional segmentation provided by the Watershed algorithm, this method allowed for the rapid classification of both afflicted and healthy individuals. Notably, this classification relied on measuring distances from various angles, which made tumor cell distinction possible. The extraction of characteristics from medical imaging data was investigated in another direction. The procedure comprised several stages, such as gathering images, normalizing them, analyzing their intensity, and extracting form and texture information. The key characteristics for categorization were chosen using linear discriminant analysis (LDA). To assess the results, reduced principal component analysis (PCA) was used [5]. Neural networks provide special and effective techniques for brain cancer classification, such as the Probability Neural Network (PNN) and the Back Propagation Network (BPN). The radial basis network was found to be an appropriate option for classification, and thresholding and tumor region isolation were among the image processing techniques used. The system functioned in two modes: training/learning and test/accreditation [6]. By utilizing matrix representations of MRI images, a probabilistic neural network model was created to identify brain cancers. Based on difference values, the model demonstrated remarkable accuracy rates ranging from 73% to 100% when tested using training and test datasets [7]. Recent research has shown that deep learning algorithms can be used to identify brain tumors. This method used cascaded deep learning Convolutional Neural Networks (CNNs) with two underlying networks—one for intra-tumor categorization and another for tumor localization [7]. Brain tumors have been classified as high or low risk based on clinical imaging using machine learning algorithms, specifically convolutional networks [8]. To achieve noninvasive detection, deep convolution networks based on Unet architecture have been proposed. These networks aim to identify brain cancers without using ionizing radiation by focusing on core tumor locations [9]. A segmentation algorithm for detecting glioma-based brain tumors was proposed. Deep convolutional neural networks were used in this method to address issues such as overfitting, noise reduction, and post-processing. The end result was a significant improvement in brain tumor segmentation [10].

III. PROPOSED METHODOLOGY

The methodology, which makes use of deep learning capabilities, is intended to improve the accuracy and efficiency of brain tumor detection in medical images, specifically Magnetic Resonance Imaging (MRI). The methodology's key stages are data collection, preprocessing, model development, training, and evaluation.

A. Dataset Collection

Obtain brain MRI scans from a variety of sources, including medical databases, research institutes, and hospitals. Make sure the information you collect is diverse and represents various types of brain tumors, such as gliomas, meningiomas, and metastases. Ascertain that you have all of the necessary permissions and ethical approvals to use the data for research purposes. Patient confidentiality and data privacy are critical; anonymize or de-identify the data to protect patient information. Ascertain that the dataset contains images of both tumor and non-tumor cases. Preprocess the collected MRI images so that they can be used to train a CNN. Sample images of tumor and non-tumor cases are shown in Fig 1.

Fig.1. Image with tumor and image without tumor

B.Image Pre-processing

Resize all MRI images to the same resolution. Scaling the images to a square shape, such as 224x224 or 256x256 pixels, is a popular option. Consistent image sizes make model training easier and lower computational complexity. Normalize pixel values to a standardized scale. Scaling pixel values to a range of [0, 1] or [-1, 1] is the most common method. This step ensures that the intensity values in all images are consistent.

Apply data augmentation techniques to the preprocessed images to augment the dataset. This is usually done after resizing and normalization. To increase dataset diversity and improve model generalization, augmentation techniques such as rotation, flipping, scaling, and adding artificial noise are used. Crop the images to focus on the tumor area if your dataset includes ROI annotations for tumor regions. This reduces computational complexity and allows the model to focus on the critical regions.

C. Data Modeling

Separate the dataset in to three parts : training, validation, and testing. A typical split is 80% training, 10% validation, and 10% testing. During model development, the validation set is used to fine-tune hyperparameters. D. Model Devlopment

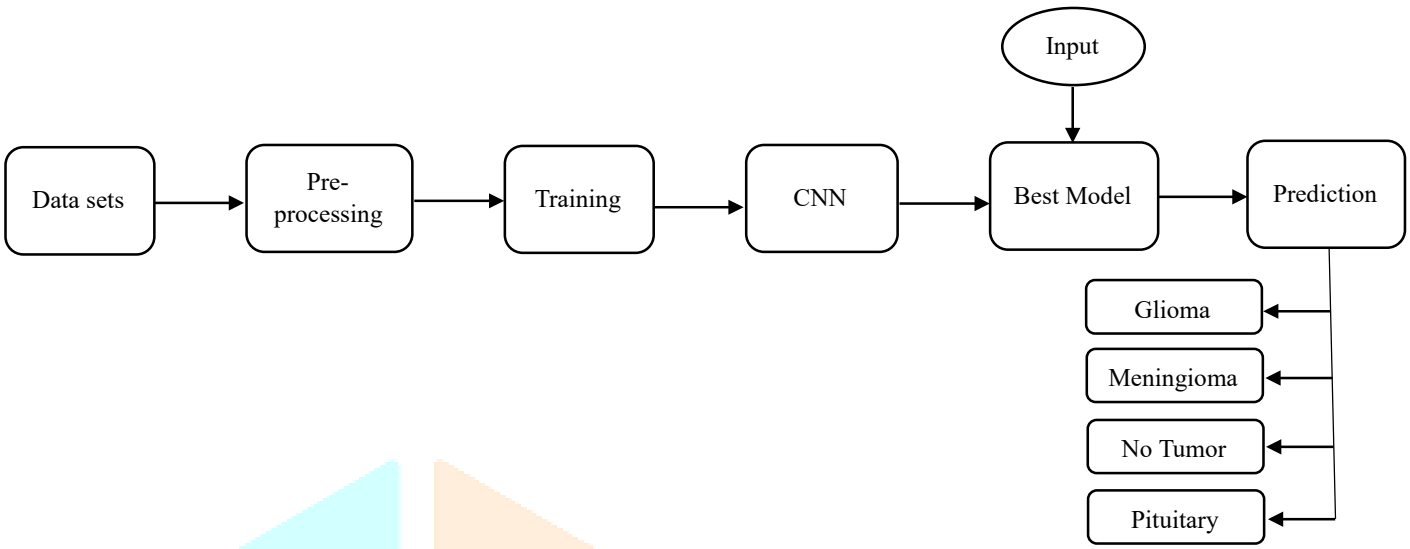
Select the type of CNN architecture and create a custom architecture, or use pre-trained models such as VGG, ResNet, or Inception and fine-tune them for the task. The architecture chosen is determined by the dataset and the complexity of the problem. Your CNN's input layer should be compatible with the preprocessed image size. Make sure the input shape corresponds to the dimensions of your resized and preprocessed MRI images. Convolutional layers should be added to your model for feature extraction. Convolutional layers learn various image features such as edges, textures, and more complex patterns. The number of convolutional layers and the number of filters in each layer can be adjusted depending on the complexity of your dataset. After each convolutional layer, use activation functions such as ReLU (Rectified Linear Unit). ReLU adds nonlinearity to the model, allowing it to learn complex patterns. Pooling layers (e.g., Max Pooling or Average Pooling) can be used to reduce the spatial dimensions of feature maps and down-sample the learned features. Pooling aids in the reduction of computational complexity.

Add dropout layers and batch normalization after convolutional layers to prevent overfitting. Dropout "drops out" a subset of neurons at random during training, while batch normalization normalizes activations, improving training stability. Flatten the feature maps and add fully connected (dense) classification layers. The number of neurons in these layers, as well as the number of dense layers, can be adjusted depending on the problem's complexity. The number of neurons in the output layer should be the same as the number of classes to predict. Typically, one neuron with a sigmoid activation function is used for binary classification (tumor vs. non-tumor). Select a suitable loss function. Binary cross-entropy is commonly used in binary classification.

Train your model on the preprocessed dataset with the loss function, optimizer, and hyperparameters you've chosen. Track training progress and performance using the validation set. Evaluate the performance of your

model using metrics such as accuracy, precision, recall, F1 score, and ROC-AUC. Ascertain that the model is producing accurate tumor predictions. Visualize the model's learned features and consider using interpretability techniques such as Grad-CAM.

E. Flowchart Of Model



IV. DATASET

Magnetic resonance imaging (MRI) is a medical imaging technique used in radiology to produce high-quality two-dimensional or three-dimensional images of the brain and brainstem. The database includes twodimensional T1-weighted contrast-enhanced images. The dataset includes three different views: axial, coronal and sagittal views. These images were classified into 4 classes. The number of images obtained for different tumor types are:

Type	Training Sample	Testing Sample	Total
Glioma	703	100	803
Meningioma	790	115	905
No Tumor	678	95	668
Pituitary	720	100	814

Table 1: Initial Dataset

A. Pre-Processing

In the Pre Processing stage, we try to create uniformity in data before feeding it to neural networks. Our images had high resolution and we downsized them to 224x224x3 this helps preserve all relevant information for our networks in fewer file sizes. The MRIs contained a black background around the central image of the brain. This black background provides no useful information for classification and would be wasted if fed to neural networks. Hence the images were cropped around the main contour. Here the biggest contour is selected and marked. Next, we find the extreme points of the contour and crop the image on those endpoints. Thus we removed most of the unwanted background and some noise present in the original image. This process is done for each image in the dataset. However, sometimes the contour mapping and cropping algorithm were not able

to correctly recognize the correct contour and wrongly cropped the image. Such images resulted in distorted images and were removed by manual inspection after the 'augmentation' phase.

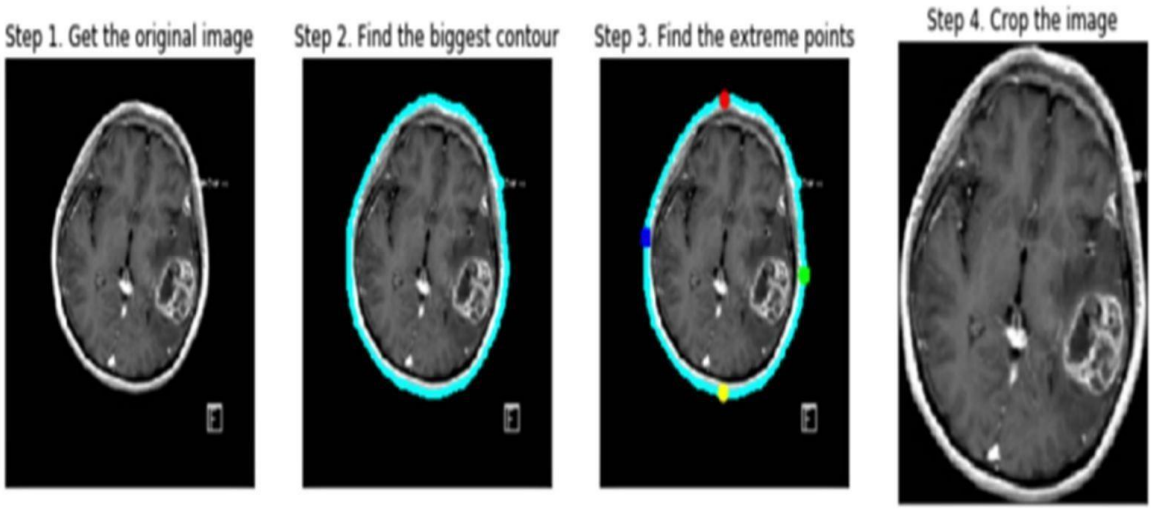


Figure 2:Contour Cropping

Here the biggest contour is selected and marked. Next, we find the extreme points of the contour and crop the image on those endpoints. Thus we have removed most of the unwanted background and some noise present in the original image. This process is done for each image in the dataset. However, note that sometimes the function may not be able to correctly recognize the correct contours and makes a mistake and wrongly crops the image. Such images should be removed by manual inspection before entering the phase. The amount of data gathered was very low and could cause the models to under-fit. Hence, we would use a brilliant technique of Data Augmentation to increase the amount of data. This technique relies on rotations, flips, change in exposure, etc to create similar images.

B. Data Augmentation

As defined in table 1 we have few images to train and test a neural network. For a network to learn to a greater extent we can increase the amount of data that is fed to it. This was done by using the technique of 'Data Augmentation'. This technique relies on rotations, flips, change in exposure, zoom, etc to create similar images. The process of augmentation was applied to every image of the dataset; this caused the number of images to increase by a factor of 21. Thus with such a huge amount of data, the chances of under-fitting of the network can be reduced. The images were shuffled and divided in a ratio of 3:1. Where 75% of data was used for training and the rest 25% used for testing and validation Figure 3 represents the augmentation we get from the above contour cropped image.

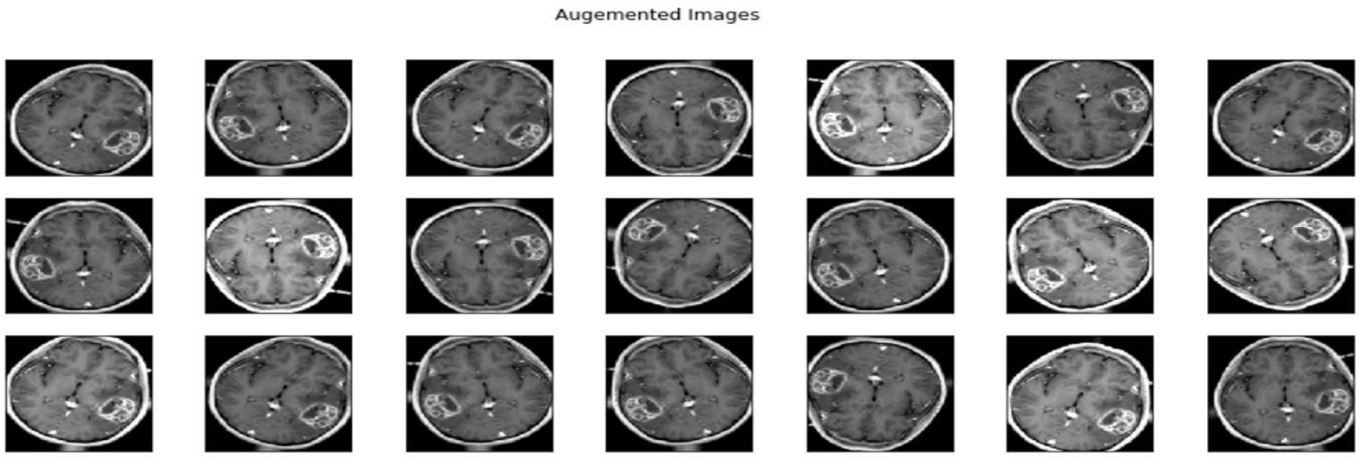


Figure 3: Augmentation

1) Training Data

No	Class	Initial Count	Augmented Count	Discarded Count	Final Count
1	Glioma	703	14763	750	14013
2	Meningioma	790	16590	1000	15590
3	No Tumor	678	14238	150	14088
4	Pituitary	720	15120	80	15040

Table 2 : Training Data

2) Testing Data

No	Class	Initial Count	Augmented Count	Discarded Count	Final Count
1	Glioma	100	2100	85	2015
2	Meningioma	115	2414	126	2288
3	No Tumor	95	1995	63	1932
4	Pituitary	100	2100	98	2002

Table 3 : Testing Data

Few images were distorted during the pre-processing and augmentation phase. These images were distorted due to improper cropping of contour. Such images were noisy and unclear, thus were not collected in the final data set. We concluded with a total of 58731 training and 8237 testing/validation images.

V. APPROACH

We built and trained models using CNN using permutation and combination of hyper-parameters like number of dense layers, number of nodes in dense layers, etc to build models which would provide promising results.

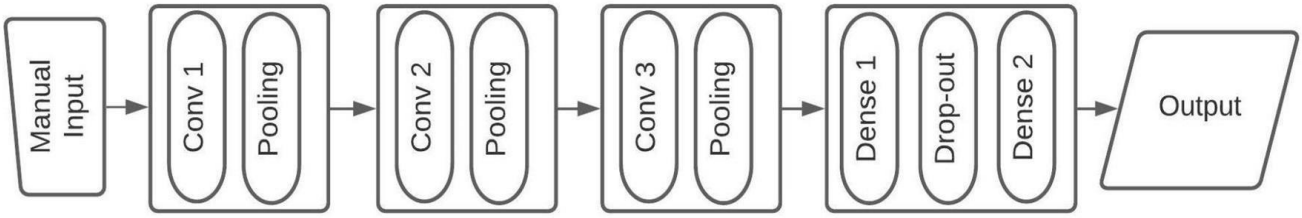


Figure 3: CNN Model Architecture

CNN Models	Test Accuracy	Test Loss	F1 - Score
1-conv-32-nodes-0-dense	86	1.77	86
2-conv-32-nodes-0-dense	85	0.76	85
3-conv-32-nodes-0-dense	88	0.48	88
1-conv-64-nodes-0-dense	85	2.39	85
2-conv-64-nodes-0-dense	89	0.55	89
3-conv-64-nodes-0-dense	89	0.55	89
1-conv-128-nodes-0-dense	84	1.36	84
2-conv-128-nodes-0-dense	82	1.01	81
3-conv-128-nodes-0-dense	86	0.95	86
1-conv-32-nodes-1-dense	24	1.38	10
2-conv-32-nodes-1-dense	88	0.59	89
3-conv-32-nodes-1-dense	86	0.58	86
1-conv-64-nodes-1-dense	26	1.36	13
2-conv-64-nodes-1-dense	88	0.48	89
2-conv-64-nodes-1-dense	88	0.54	89
1-conv-128-nodes-1-dense	85	0.73	85
2-conv-128-nodes-1-dense	85	0.77	83
3-conv-128-nodes-1-dense	90	0.43	91
1-conv-32-nodes-2-dense	24	1.38	10
2-conv-32-nodes-2-dense	24	1.38	10

3-conv-32-nodes-2-dense	63	0.84	57
1-conv-64-nodes-2-dense	24	1.38	10
2-conv-64-nodes-2-dense	83	0.70	83
3-conv-64-nodes-2-dense	85	0.55	86
1-conv-128-nodes-2-dense	24	0.00	10
2-conv-128-nodes-2-dense	85	0.78	87
3-conv-128-nodes-2-dense	85	0.64	85

Table 5: CNN Models

VI. MISCELLANEOUS

Google Tensorflow version 1.51.2 was used for all TensorFlow libraries. The models were compiled using a loss function known as sparse categorical cross-entropy. Sparse Categorical Cross-Entropy and Categorical Cross Entropy both compute categorical crossentropy. The only difference is in how the targets/labels should be encoded. Sparse Categorical Cross Entropy is more efficient when you have a lot of categories. In sparse categorical cross-entropy, truth labels are integer encoded, for example, (1), (2), and (3) for a 3-class problem. Also, the 'Adam' optimizer was used during the compilation. Adam is an adaptive learning rate optimization algorithm that's been designed specifically for training large datasets on deep neural networks. The algorithm leverages the power of adaptive learning rates methods to find individual learning rates for each parameter. Callbacks of tensorboard were used for collecting the logs and early stopping was also used. Early stopping helped to monitor the overall learning of the model and would terminate learning if the validation loss parameter would not improve for a few epochs. For our paper, we have found that a dropout of 30-35% was effective in avoiding overfitting of training data. The models were trained on AWS SageMaker on instance 'ml.p2.xlarge'. The GPU used for the training of models was the NVIDIA V100. The total time for preprocessing, augmentation and pickling was around 50 hours and the total training time was 10 hours.

VII. CONCLUSION

Train your model on the preprocessed dataset with the chosen loss function, optimizer, and hyperparameters. Monitor training progress and use the validation set to track performance. Evaluate your model's performance using metrics such as accuracy, precision, recall, F1 score, and ROC-AUC. Check that the model is producing accurate tumor predictions. Consider using techniques such as GradCAM to improve interpretability by visualizing the model's learned features. To improve the accuracy of CNN models, researchers can continue to refine and optimize them. This may entail experimenting with various architectures, hyperparameters, and regularization techniques. Using ensemble methods to combine predictions from multiple models can result in more robust and accurate results. The model's generalization and performance can be improved by expanding the dataset with a larger and more diverse collection of MRI images. Developing CNN-based systems capable of detecting brain tumors in real-time or near-real-time during MRI scans can provide medical professionals with immediate feedback. Collaborating with medical practitioners to integrate CNNbased brain tumor detection systems into clinical practice, ensuring their applicability and efficacy in real-world healthcare scenarios.

REFERENCES

[1] Dr. B Nithya Ramesh, Dr. Asha V, Gokul Pant, Dr. Arpana Prasad, Harshita L, "Brain Tumor Detection using CNN withResnet50," IEEE Xplore, 07 July 2023 https://ieeexplore.ieee.org/document/10169663

[2] C. Rajeshkumar, K.Ruba Soundar, M. Sneha, S.Santhana Maheswari, M.Subbu Lakshmi, R. Priyanka, "Convolutional Neural Networks (CNN) based Brain Tumor Detection in MRI Images," IEEE Xplore 14 March 2023, https://ieeexplore.ieee.org/document/10060968

[3] Bhuvaneshwari Shetty, Roshan Fernandes, Anisha P Rodrigues, P. Vijaya, "Brain Tumor Detection using Machine Learning and Convolutional Neural Network," IEEE Xplore 23 December 2022, https://ieeexplore.ieee.org/document/10060254

[4] Nisar Ahmad, Kamil Dimililer, "Brain Tumor Detection Using Convolutional Neural Network," IEEE Xplore 22 October 2022, https://ieeexplore.ieee.org/document/9932741