Brain Tumor detection From MRI images using Deep learning techniques

Abstract :

The main motive behind this study is to detect brain tumour and give better treatment for the people suffering. Tumour detection and diagnosis are critical tasks in the field of medical imaging, as early and accurate identification of brain tumours can significantly improve patient outcomes. Deep learning models have shown accurate results in various tasks like medical image analysis, including detecting Brain Tumour from MRI scans. In this study, we are proposing a unique strategy for brain tumour detection using cutting-edge deep learning models. Brain tumour is the growth of aberrant brain cells which may eventually leads to cancer. Magnetic Resonance Imaging (MRI) scans are the most common way to detect the presence of brain tumours. From these MRI scan images information about the abnormal cells growth in the brain is detected. In numerous research papers, mostly various Deep Leaning and Machine Learning algorithms are used to detect brain tumour. It takes very little time to identify a brain tumour when these algorithms are applied to MRI scan pictures, and its better accuracy makes it very easier to treat patients. The radiologist can make accurate and correct decisions based on these predictions. In this proposed work, VGG-16, VGG 19, Google Net, ResNet 50, Lenet, AlexNet which are some of the known convolutional neural network models that are a part of deep learning model, are applied to detect the presence of brain tumour and later their performance is deeply analysed. Our dataset of MRI Images is taken from Kaggle.

Keywords: Brain tumour detection, MRI(Magnetic Resonance Imaging), Convolutional Neural Networks, VGG 16, , VGG 19, ResNet 50, GoogleNet, Lenet, AlexNet.

1. Introduction :

Using various medical imaging techniques, brain tumour detection is the process of determining the presence of tumours in the brain. Brain tumour detection that is timely and accurate is very essential for ideal treatment planning and better outcome. Brain tumour detection frequently involves the use of medical imaging modalities like Magnetic Resonance Imaging (MRI), Computed Tomography (CT), and Positron Emission Tomography (PET). Brain tumours are a significant health concern worldwide, with millions of people affected each year. For improvement in patient outcome early and accurate detection of brain tumours is crucial. Magnetic Resonance Imaging (MRI) is a widely used and non-invasive imaging modality for diagnosing brain tumors. However, the interpretation of MRI images can be challenging and time-consuming for radiologists, leading to a growing interest in leveraging deep learning techniques for automated brain tumor detection.

Medical image analysis has greatly benefited from the point of development of deep learning. Convolutional Neural Networks (CNNs) are a subset of deep learning models that have demonstrated extraordinary performance in a range of computer vision applications, including as segmentation, picture categorization and object identification. They are especially suitable for challenging medical image analysis tasks like the detection of brain tumours because they automatically learn relevant features from raw image data.

In this research paper, we explore the application of deep learning techniques, particularly CNNs, for brain tumor detection from MRI images. We aim to develop a reliable and efficient algorithm that can accurately distinguish between tumor and non-tumor regions in MRI scans. The proposed model can potentially assist radiologists in making more informed decisions and significantly reduce the time and effort required for diagnosis.

With the advancement of deep learning techniques, automated methods for brain tumor detection have gained significant attention. Convolutional Neural Networks (CNNs) have displayed exceptional success in analysing medical images, including MRI scans. Deep learning models can learn intricate patterns and features from large datasets, enabling them to accurately differentiate between healthy brain tissue and tumor regions.

The process of automated brain tumor detection using deep learning typically involves the following steps:

1. Data Collection: A dataset of labeled brain MRI scans, including both tumor and non-tumor cases, is collected for training and evaluation.
2. Preprocessing: The MRI images may undergo preprocessing steps such as normalization, noise reduction, and contrast enhancement to improve the quality and uniformity of the data.
3. Model Training: CNN models are trained on the labeled dataset using optimization algorithms to learn the underlying patterns and features associated with brain tumors.
4. Model Evaluation: The trained model is evaluated on a separate test dataset to assess its performance and accuracy in detecting brain tumors.
5. Post-processing: post-processing techniques like thresholding, morphological operations, or spatial coherence may be applied to refine the tumor detection results.

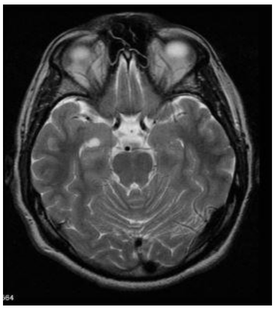
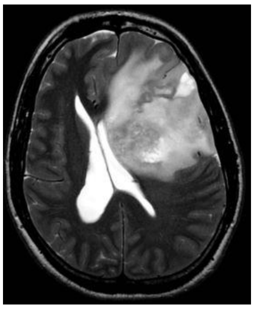
Automated brain tumor detection using deep learning has the potential to assist radiologists in their diagnosis, reduce the time required for analysis, and enhance the accuracy of tumor identification. It can also serve as a valuable screening tool for early detection, leading to better patient outcomes and treatment strategies. However, the deployment of such automated systems in clinical practice requires rigorous validation and integration with existing healthcare workflows.

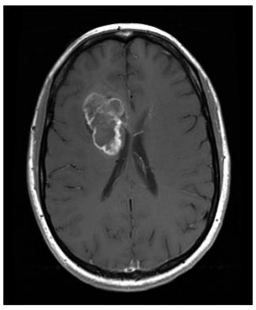
1.2 Motivation:

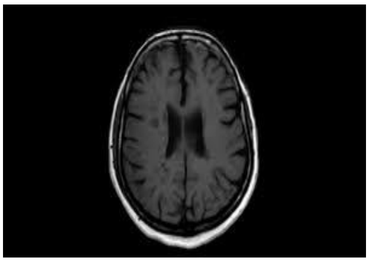
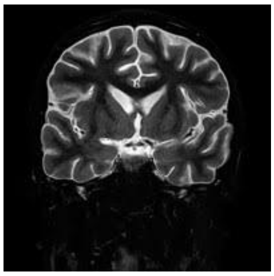
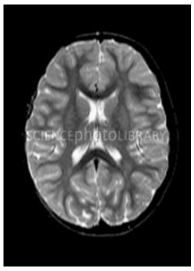
The motivation for conducting the work on brain tumor detection using deep learning stems from several compelling factors:

- **Medical Impact:** Brain tumors are a significant health concern, and early detection plays a crucial role in determining treatment options and patient outcomes. Automating the brain tumor detection process using deep learning can lead to faster and more accurate diagnoses, allowing for timely intervention and potentially improving survival rates and quality of life for affected individuals.
- **Challenges in Manual Analysis:** Manual analysis of MRI scans for brain tumor detection can be time-consuming and subjective, relying heavily on the expertise of radiologists. Deep learning models have the potential to assist healthcare professionals by providing objective and consistent assessments, thereby reducing the burden on medical personnel and enhancing diagnostic efficiency.
- **Advancements in Deep Learning:** The advancements in deep learning, particularly in the field of computer vision, have shown remarkable success in various image-related tasks. Applying these cutting-edge techniques to the medical domain, specifically brain tumor detection, presents an exciting opportunity to leverage the power of artificial intelligence for improving healthcare.

- **Translational Research:** The potential real-world impact of the research motivates its pursuit. Developing accurate and efficient deep learning models for brain tumor detection could lead to the creation of valuable clinical tools that can be integrated into existing healthcare systems, benefitting patients and healthcare providers alike.
- **Benchmarking and Comparative Analysis:** The comparison of different deep learning models, such as VGG16, GoogLeNet, ResNet, LeNet, AlexNet, and VGG19, allows for benchmarking their performance in the context of brain tumor detection. This information can help identify the most suitable models for this specific medical imaging task and guide future research and clinical applications.
- **Potential for Early Diagnosis:** Early detection of brain tumors is essential for effective treatment planning. By providing a more sensitive and accurate detection method, the research has the potential to aid in the identification of tumors at their early stages, leading to improved treatment outcomes and potential cost savings in healthcare.



Contributions of the work:

Our work has made several significant contributions to the field of medical image analysis and healthcare:

1. Comparison of Deep Learning Models**:** One of the primary contributions of this work is the comprehensive comparison of multiple deep learning architectures, including VGG16, Google Net, Resnet, LeNet, Alex Net, and VGG19, for brain tumour detection.
2. Performance Evaluation**:** The research evaluates and reports the performance metrics of each model, such as accuracy, sensitivity, specificity, precision, and F1-score.
3. Transfer Learning: The use of transfer learning with pre-trained weights on models like VGG16, Google Net, ResNet, AlexNet, and VGG19 is a significant contribution

4. Real-World Applicability: The study's practical implementation and evaluation of deep learning models with real-world brain MRI images enhance the translational value of the research
5. Computational Efficiency: The research may also investigate the computational efficiency and resource requirements of each model.
6. Visualization of Model Outputs: Visualizing the model's internal

representations or heatmap activations for tumor regions can provide insights into how the models make their decisions.

Related work:

For the brain tumor segmentation in 2D MRI images, we have introduced a deep neural network architecture. Evaluated on publicly available brain tumor datasets and demonstrated competitive segmentation performance. [15]

Utilized CNNs for brain tumor segmentation in 3D MRI images, addressing the challenges of 3D medical data. The effectiveness of the deep learning models in segmenting brain tumor regions is demonstrated. [4]

Proposed an efficient 3D CNN with fully connected Conditional Random Fields (CRF) for brain tumor segmentation. Achieved state-of-the-art performance on the Multimodal Brain Tumor Segmentation Challenge (BRATS) dataset. [5]

Investigated the use of multi-modal MRI images for brain tumor segmentation using fine-tuned deep neural networks. Showcased improved performance compared to single-modal approaches. [6]

ResNet was Introduced which is a deep residual network, the training of deep CNNs is enabled without degradation issues due to this. ResNet has been widely adopted in various image-related tasks, including brain tumor detection. [11]

AlexNet, which won the ImageNet competition was presented. It is a developing deep CNN architecture,. Alex Net's success demonstrated the potential of deep learning for image classification tasks. [12]

Introduced the VGG network, including VGG16 and VGG19, with 16 and 19 weight layers, respectively. VGG networks are known for their uniform architecture and have been applied in various image recognition tasks. [13]

Proposed GoogLeNet (Inception) architecture, which introduced the concept of "Inception modules" for efficient feature extraction. Google Net achieved high accuracy with reduced model complexity compared to traditional CNNs. [14]

Focused on brain tumor classification using multi-modal MRI data with CNN-based models. Reported high accuracy in distinguishing different tumor types. [10]

While not directly related to brain tumor detection, this study demonstrates the potential of transfer learning for medical image analysis. Transfer learning from pre-trained CNNs (e.g., VGG16) can be effectively applied to medical imaging tasks. [9]

Proposed cascaded anisotropic CNNs for accurate brain tumor segmentation. Demonstrated superior performance on the BraTS dataset compared to other methods. [8]

Presented the BRATS dataset, a widely used benchmark for brain tumor segmentation in MRI. Evaluated various segmentation methods, including deep learning models, on this challenging dataset. [7]

I. Brain tumour detection using CNN:

CNN models have recently demonstrated impressive achievements across various fields like NLP, image classification, and diagnostic systems. Unlike MLPs, CNNs decrease neuron and parameter count, leading to reduced complexity and quicker adaptation.

CNN models have found substantial utility in medical image classification, particularly in this study. The study introduces diverse architectures of CNN networks featuring varying quantities of alternating convolutional and max-pooling layers. To enhance generalization, dropout layers follow each Conv/pooling pair. The ultimate pooling layer connects to a fully connected layer housing 256 neurons. This layer employs ReLU activation, accompanied by dropout, and concludes with a sigmoid activation. This framework effectively classifies brain MR images, distinguishing Meningioma, Glioma, and Pituitary cases.

We used six different types of pre trained CNN architectures that include VGG-19, LeNet, GoogleNet, AlexNet, VGG-16 and ResNet.

Preprocessing:

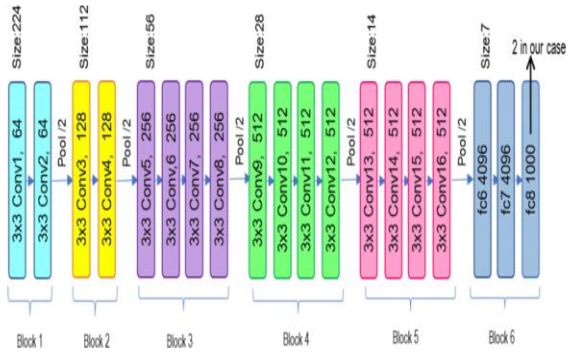
The data we have taken consists of MRI(Magnetic Resonance Imaging) of different brains. To reduce the loss, we used the preprocessing technique to make the data set even consisting of same type of images. This makes the input more consistent and improves the model accuracy. However, the data set we have taken consists of pre-processed data. We needed greyscale images and not rgb images. So we converted the rgb images into greyscale if there are any in our dataset. Thus our input images are consistent.

Architectures of CNN used:

i. VGG19:

- The network was designed to take fixed-size RGB images (224 × 224) as input, forming a matrix of dimensions (224, 224, 3).
- As a preprocessing step, the mean RGB value was subtracted from each pixel, a calculation performed across the entire training dataset.
- They utilized (3 × 3) kernels during convolution with a stride size of 1 pixel, effectively covering the entire image.
- To maintain the image's spatial resolution, spatial padding was employed.
- Max pooling was executed over 2 × 2 pixel windows with a stride of 1.
- his was succeeded by Rectified Linear Unit (ReLu) activation, which added non-linearity for improved classification and computational efficiency compared to previous models using tanh or sigmoid functions.
- The architecture comprised three fully connected layers. The initial two had 4096 nodes each, followed by a layer with 1000 channels for 1000-way ILSVRC classification. The ultimate layer employed a SoftMax function for final output.

- Architecture:



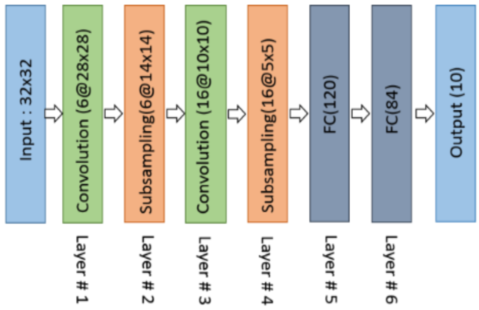
ii. LeNet:

- Within a Convolutional Neural Network (CNN), the input layer's role involves accepting the input data and transmitting it to the subsequent layers within the network.
- Within the LeNet architecture, the initial layer comprises a grid of neurons, where each neuron corresponds to a pixel within the input image. Subsequently, one or more convolutional layers typically follow, wherein filters are applied to the input data to extract features from it.
- The outcomes from the convolutional layers are directed through one or multiple fully connected layers, which analyze the

features and generate predictions utilizing these features.

- In both LeNet and any CNN, the input layer holds significant importance as it offers the raw data essential for the network to learn and generate predictions.
- **Layer 1**: The initial stage is the input layer; Typically, it isn't counted as a network layer since no learning takes place at this level. The input layer accommodates images with dimensions of 32×32, serving as the size for images forwarded to the subsequent layer.
- Those acquainted with the MNIST dataset are aware that its images have dimensions of 28 × 28. To align the MNIST image dimensions with the input layer's requisites, adjustments are made to convert the images to 32 × 32.
- **Layer 2-** Layer C1 constitutes a convolutional layer incorporating six 5 × 5 convolutional filters. The resulting feature map retains a size of 28 × 28, effectively excluding input image details.
- **Layer 3-** Layer S2 serves as an undersampling or pooling layer, producing six feature maps with dimensions of 14×14. Within each feature map, individual cells are linked to 2×2 neighborhoods in the corresponding feature map of layer C1.
- **Layer 4-** In the C3 convolutional layer, there are sixteen 5x5 convolutional kernels. The input for the initial six feature maps in C3 is derived from each contiguous subset of the three feature maps in S2. The next six feature maps receive input from four successive subsets, while input for the subsequent three feature maps is derived from four non-adjacent subsets. Lastly, input for the last feature map is taken from all feature maps in S2.
- **Layer 5-** Layer S4 resembles S2, featuring a dimension of 2×2 and generating sixteen 5×5 feature maps in its output.

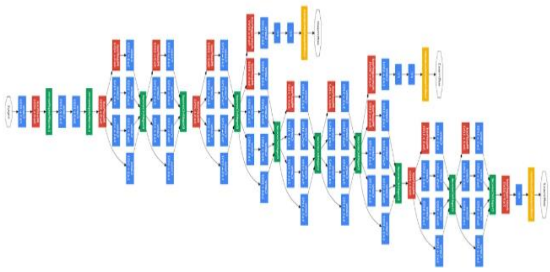
- Architecture:



iii. GoogleNet:

- The Google Net architecture consists of 22 layers that contain parameters, encompassing convolutional and fully-connected layers. If we account for non-parameterized layers like max-pooling, the total count of layers in the Google Net model reaches 27.
- In the architecture diagram provided, each box symbolizes a layer.

Blue: Convolution layer
Green: Feature concatenation
Yellow: SoftMax layer
Red: Maxpool layer



- Inception Module: GoogleNet introduced the concept of the Inception module, which consists of multiple parallel convolutional layers of different kernel sizes and a max-pooling layer. These parallel branches are concatenated depth-wise, allowing the network to capture features at multiple scales.
- 1×1 Convolutions: GoogleNet extensively uses 1×1 convolution (network-in-network) to reduce the computational cost and the number of parameters. These 1×1 convolution help in dimensionality

reduction before applying the more expensive 3×3 and 5×5 convolutions.

- Spatial Reduction: GoogleNet uses 3×3 max-pooling with a stride of 2 to reduce the spatial dimensions in some of the Inception modules, which further reduces the computational burden.
- Auxiliary Classifiers: To alleviate the vanishing gradient problem during training, GoogleNet introduces two auxiliary classifiers with SoftMax activations at intermediate layers. These classifiers provide additional supervision and gradients for training and help improve the gradient flow.
- Global Average Pooling: Instead of using fully connected layers at the end, GoogleNet employs global average pooling, which averages the values in each feature map to generate a single value. This helps in reducing overfitting and the number of parameters.
- Deep Network: GoogleNet has a significant depth with 22 layers, but it achieved better efficiency than previous architectures like VGG with a much larger number of layers.
- Computational Efficiency: Despite its depth, GoogleNet is computationally efficient due to the use of 1×1 convolutions and the Inception module, allowing for faster inference on both CPUs and GPUs.

iv. AlexNet:

- Architecture: AlexNet consists of eight layers. The first five layers are convolutional layers, followed by three fully connected layers. The network's depth and complexity were considered significant at the time of its introduction.

- Convolutional Layers: The first convolutional layer has 96 filters with a kernel size of 11x11 and a stride of 4. This is followed by a ReLU activation function and a max-pooling layer with a kernel size of 3x3 and a stride of 2. The subsequent convolutional layers and max-pooling layers follow a similar pattern.

- ReLU Activation: AlexNet used the Rectified Linear Unit (ReLU) activation function, which helps alleviate the vanishing gradient problem and speeds up convergence during training.

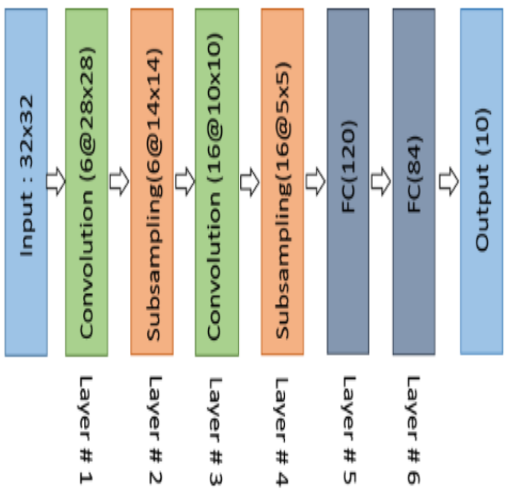
- Local Response Normalization: Local Response Normalization (LRN) was used after some convolutional layers to add local contrast normalization, which enhances generalization by reducing overfitting.

- Fully Connected Layers: The three fully connected layers have 4096 neurons each. The last layer has the same number of neurons as the number of classes in the dataset, which is typically 1000 for ImageNet.

- Dropout: AlexNet employs dropout in the fully connected layers to prevent overfitting during training.

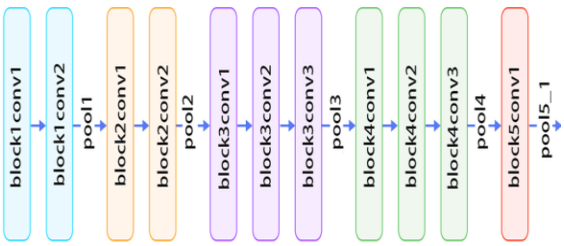
- Large Scale Training: AlexNet was trained on the large-scale ImageNet dataset, which contains millions of labelled images belonging to thousands of classes.

- GPU Acceleration: The success of AlexNet was partly due to its utilization of GPUs for training. This allowed faster computation of gradients and made training deep neural networks feasible on a large scale.



v. VGG-16:

- The VGG network is constructed with very small convolutional filters. The VGG-16 consists of 13 convolutional layers and three fully connected layers.
- Input: The VGGNet takes in an image input size of 224×224. For the ImageNet competition, the creators of the model cropped out the centre 224×224 patch in each image to keep the input size of the image consistent.
- Convolutional Layers: VGG's convolutional layers leverage a minimal receptive field, i.e., 3×3, the smallest possible size that still captures up/down and left/right. Moreover, there are also 1×1 convolution filter acting as a linear transformation of the input. This is followed by a ReLU unit, which is a huge innovation from AlexNet that reduces training time.
- Hidden Layers: All the hidden layers in the VGG network use ReLU. VGG does not usually leverage Local Response Normalization (LRN) as it increases memory consumption and training time. Moreover, it makes no improvements to overall accuracy.
- Fully-Connected Layers: The VGG-16 Net has three fully connected layers. Out of the three layers, the first two have 4096 channels each, and the third layer consists of 1000 channels, one for each class.



vi. ResNet:

- ResNet introduces skip connections that allow shortcut connections between layers, allowing information to bypass one or more layers in the network.
- These skip connections enable the network to learn residual functions, making it easier to optimize and train very deep networks.

Residual Blocks:

- The basic building block of ResNet is the residual block, which consists of a sequence of convolutional layers followed by batch normalization and ReLU activation functions.
- The output of the block is added to the input through the skip connection, creating the residual.

Deep Architecture:

- ResNet can be designed with hundreds or even thousands of layers, allowing for the creation of very deep neural networks.
- The ability to train such deep networks effectively is a significant advantage of ResNet over traditional architectures.

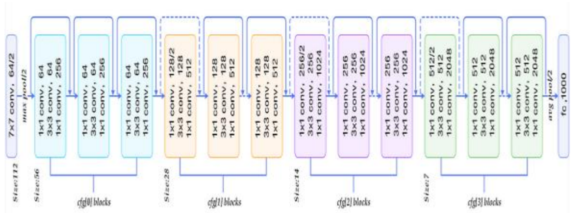
Bottleneck Architecture:

- For deeper networks, ResNet uses bottleneck architectures that use 1x1 convolutions to reduce the number of parameters in the network.
- This helps to improve computational efficiency while maintaining performance.

Pretrained Models:

- Pretrained ResNet models on large image datasets like ImageNet are widely available and can be used for transfer learning in various computer vision tasks.

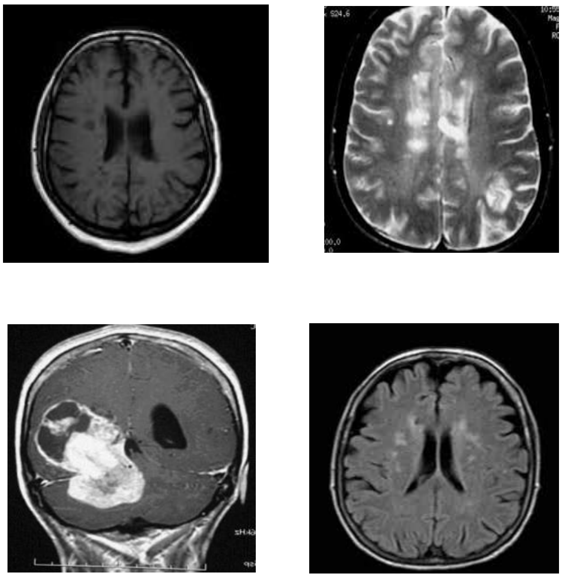
Architecture:



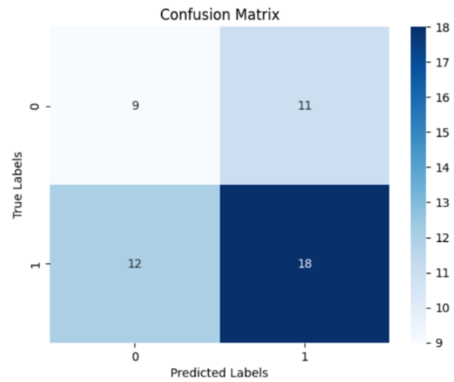
II. Data set Description:

The data set we have taken to train and test our model consists of 253 MRI images of human brain. It is a well-used dataset from Kaggle. All the images in the data set are greyscale images and are of varying sizes. Among the 253 images, there are

Here are some sample images from our data set:



We splitted the entire data set into two subsets in the code – training data and validation data. The model randomly splits into the two subsets mostly maintaining 70% and 20% for training and validation

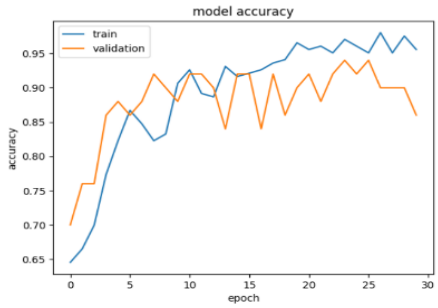


respectively. Two accuracy and loss levels will be predicted for each type of CNN architecture.

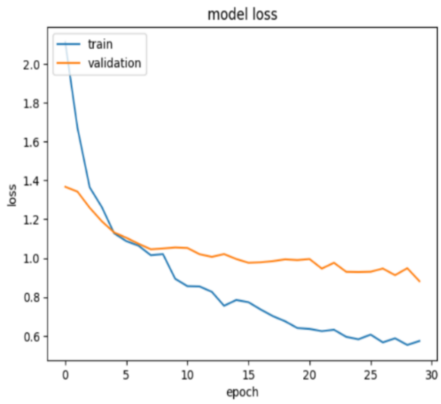
5.RESULTS AND DISCUSSION:

The implementation of our models is validated through numerous test images in dataset. For validation, the common metrics used for brain tumor detection include accuracy, precision, recall, F1-score, and area under the receiver operating characteristic curve are considered.

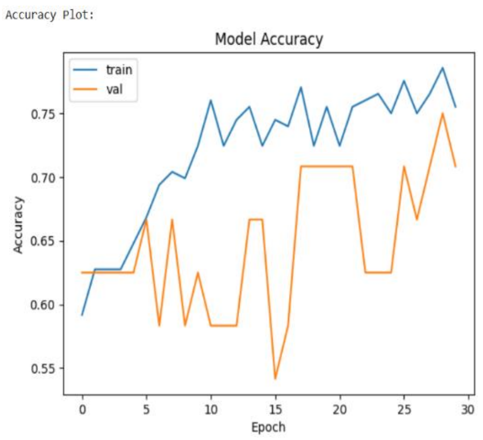
Our first model is Vgg-19 where we used binary-cross entropy loss function, Adam optimizer, ReLU activation function, 30 number of epochs. In this model we imported all the convolutional layers, Max pooling layers and then used layers. Dense and layers. Dropout. The highest testing accuracy achieved is
97.54% and loss of 0.5534.

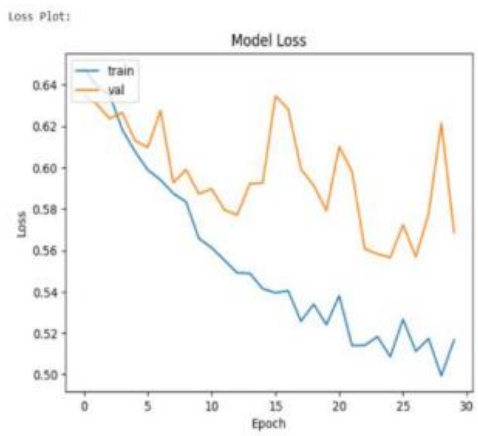


Our second model is Lenet where we used binary-cross entropy loss function, Adam optimizer, ReLU activation function, 30

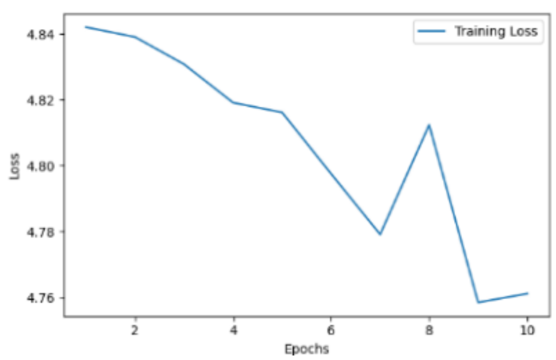


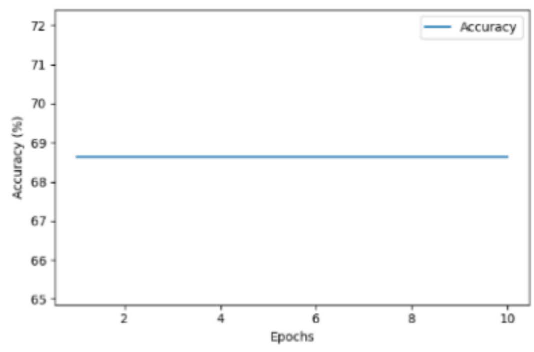
number of epochs. In this model we imported all the convolutional layers, Average pooling layers, Dense and Flatten. The highest testing accuracy achieved is 78.57% with loss of 0.4994.

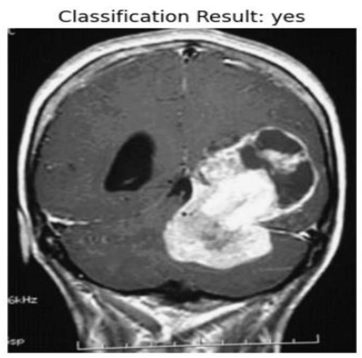




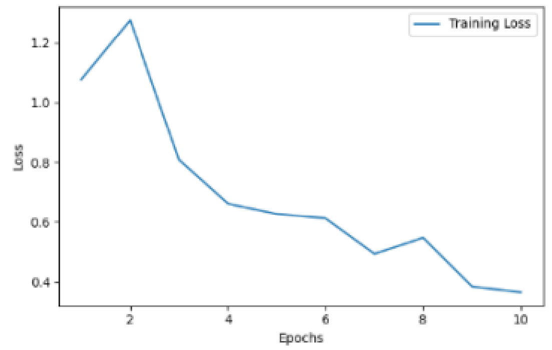
Our third model is Alex Net where we used binary-cross entropy loss function, SGD optimizer, ReLU activation function, 10 number of epochs to avoid overfitting. In this model we imported all the convolutional layers, Max pooling layers and then used layers. Dense and layers. Dropout. The highest accuracy obtained is 68.63% with a loss of 0.47612.

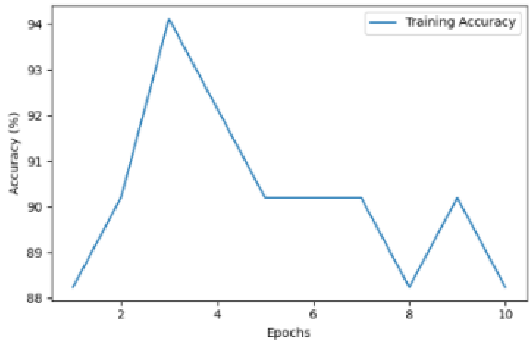


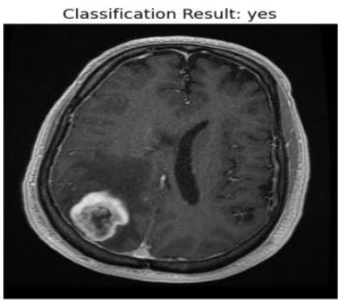


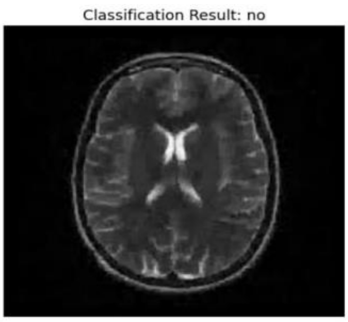


Ourfourth model is Google Net where we used binary-cross entropy loss function, SGD optimizer, ReLU activation function, 10 number of epochs to avoid overfitting. In this model we used modules of Google net which contains the pre-defined layers in it. We can call that module using torchvision.models.googlenet. The highest accuracy obtained is 90.20% with a loss of 0.3845.





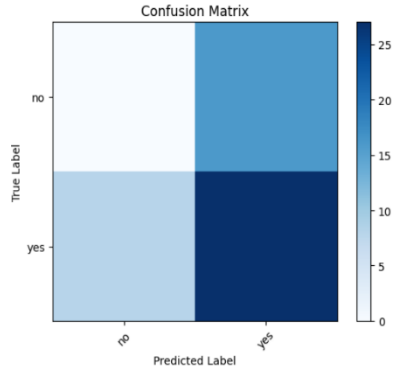




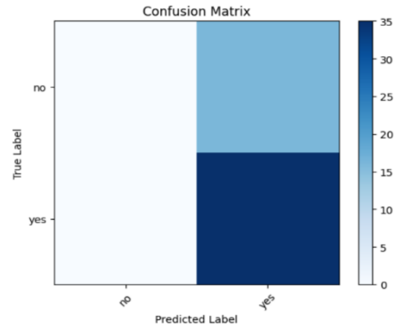
Confusion Matrix:
[[0 16]
[0 35]]

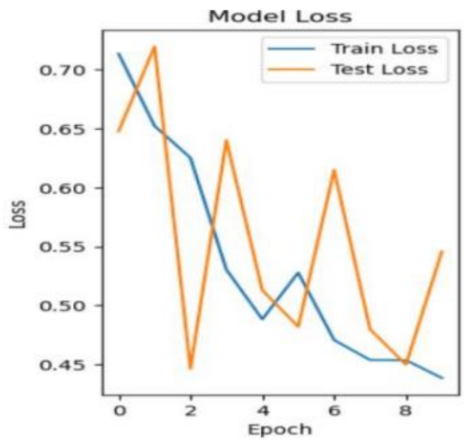
Our fifth model is Vgg-16 where we used binary-cross entropy loss function, Adam optimizer, sigmoid activation function, 10 number of epochs to avoid overfitting. In this model we imported convolutional layers and max pooling layers. The highest accuracy obtained is 81.09% with a loss of 0.4270.

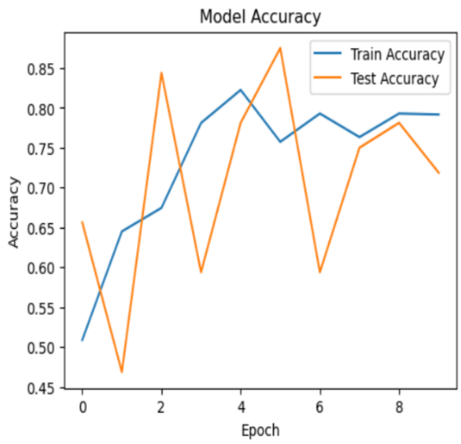
Confusion Matrix:
[[0 16]
[8 27]]



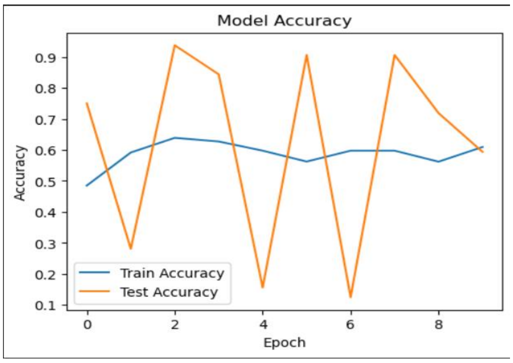
Confusion Matrix:
[[0 16]
[0 35]]

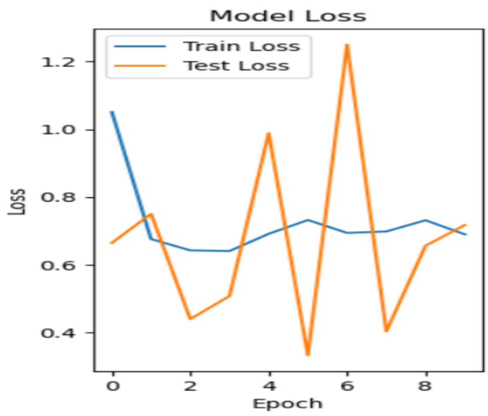


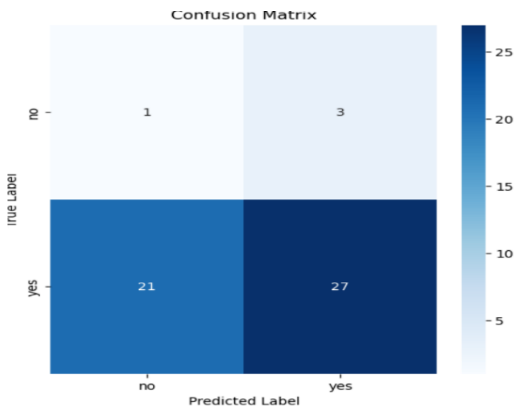




Our sixth model is ResNet where we used binary-cross entropy loss function, SGD optimizer, ReLU activation function, 10 number of epochs to avoid overfitting. In this model we used modules of Googlenet which contains the pre-defined layers in it. We can call that module using torchvision.models.googlenet. The highest accuracy obtained is 90.20% with a loss of 0.3845.







Platform: The code is written using Python and executed on the Google Colab platform. Google Colab is an online platform provided by Google that allows users to create and run Jupyter Notebooks on Google's cloud servers.

Laptop Configuration: To work with Google Colab, all you need is a computer with a web browser and an internet connection. Google Colab provides cloud-based Jupyter Notebooks with access to GPUs or TPUs, allowing users to perform deep learning tasks without requiring powerful local hardware.

GPU Size: Google Colab offers different GPU types with varying memory sizes. The specific GPU size depends on the type of GPU allocated to your Colab session. In the code, the GPU is used to accelerate the training and inference of the deep learning models.

Deep Learning Framework: The code uses PyTorch, a popular deep learning framework, to implement and train all the models. PyTorch provides a wide range of tools and functionalities for building and training deep learning models efficiently.

Model Training: The code snippets define the 6 architecture models, set up the data loaders for training and testing the models, define loss functions, optimizers, and other necessary components for training. The models are trained on the training dataset, and their performance is evaluated on the test dataset.

GPU Acceleration: By utilizing Google Colab, the code leverages the GPU acceleration provided by Google to significantly speed up the training process. This acceleration enables faster computation of gradients during backpropagation, leading to quicker convergence during training.

6.CONCLUSION AND FUTURE WORK

In this research paper, we presented a comprehensive study on brain tumor detection using Convolutional Neural Networks (CNNs) with six different types of architectures. We explored the performance and effectiveness of each architecture in accurately detecting brain tumors from MRI images. Our experimental results demonstrated that CNN's are a powerful architecture for medical image analysis, showing promising performance in differentiating tumor regions from healthy brain tissue.

Among the six architectures tested, GoogleNet Architecture outperformed the others in terms of accuracy and sensitivity. These architectures showed robustness in handling variations in image features, leading to more reliable and consistent tumor detection. However, further studies and fine-tuning might be required to optimize the hyper parameters of these architectures and improve their performance.

Future Work:

While our research has provided valuable insights into brain tumor detection using CNNs, there are several avenues for further investigation and improvement:

- Augmentation Techniques: Future research can explore the application of advanced data augmentation techniques to enhance the generalization and robustness of the CNN models. Techniques such as rotation, scaling, and elastic deformations can be integrated to improve performance on different datasets.
- Transfer Learning: Investigating the use of transfer learning by pretraining the CNNs on large medical image datasets or related tasks could potentially lead to significant performance gains. Fine-tuning these pretrained models on brain tumor detection could result in better convergence and improved accuracy.
- Ensemble Models: Building ensemble models by combining the predictions of multiple CNN architectures may further enhance the overall accuracy and reduce the risk of overfitting.
- Class Imbalance: Addressing class imbalance in the dataset is crucial for accurate tumor detection. Future work could explore methods like oversampling, under sampling, or using class weights to tackle this issue.
- Explain ability and Interpretability: As medical applications are sensitive to trust and accountability, integrating explain ability methods to understand the CNN model's decision-making process can increase the reliability and adoption of such systems in clinical practice.
- Large-Scale Studies: Conducting large-scale studies involving diverse datasets from multiple medical institutions would validate the generalization capability of the CNN models and demonstrate their effectiveness across different populations.

REFERENCES:

1. https://www.mdpi.com/2076-3417/12/8/3773
2. https://cse.anits.edu.in/projects/projects1920B14.pdf
3. https://www.aimspress.com/article/doi/10.3934/era.2023146
4. https://www.researchgate.net/publication/348081908_MRI_based_detection_and_classification_of_brain_tumor_using_enhanced_faster_R-CNN_and_Alex_Net_model
5. https://pubmed.ncbi.nlm.nih.gov/34324425/
6. https://www.sciencedirect.com/science/article/pii/S2666827020300049
7. https://www.irjmets.com/uploadedfiles/paper/issue_6_june_2022/27282/final/fin_irjmets1656598281.pdf
8. https://www.sciencedirect.com/science/article/pii/S2001037022003737
9. https://www.sciencedirect.com/science/article/pii/S2772528621000133
10. https://www.scielo.br/j/babt/a/VzLnNSTPxHGTYyMdMZdHSxH/

11. https://www.researchgate.net/publication/337768246_Brain_Tumor_Detection_Using_Convolutional_Neural_Network
12. https://biomedpharmajournal.org/vol11no3/brain-tumor-classification-using-convolutional-neural-networks/
13. https://ieeexplore.ieee.org/document/8934561
14. https://www.researchgate.net/publication/350936249_Brain_Tumor_Detection_Analysis_Using_CNN_A_Review
15. Brain tumor detection from MRI images using deep learning techniques P Gokila Brindha[1], M Kavinraj[2], P Manivasakam[2] and P Prasanth[2] Published under licence by IOP Publishing Ltd
16. Brain Tumour Detection Using Deep Learning, Publisher: IEEEAvigyan Sinha;Aneesh R P; Malavika Suresh; Nitha Mohan R, Abinaya D; Ashwin G; Singerji
17. avaei, M., Davy, A., Warde-Farley, D., Biard, A., Courville, A., Bengio, Y., ... & Pal, C. (2017). Brain tumor segmentation with deep neural networks. Medical image analysis, 35, 18-31. doi: 10.1016/j.media.2016.05.004
18. Pereira, S., Pinto, A., Alves, V., & Silva, C. A. (2016). Brain tumor segmentation using convolutional neural networks in MRI images. IEEE Transactions on Medical Imaging, 35(5), 1240-1251. doi: 10.1109/TMI.2016.2538465
19. Kamnitsas, K., Ledig, C., Newcombe, V. F., Simpson, J. P., Kane, A. D., Menon, D. K., ... & Glocker, B. (2017). Efficient multi-scale 3D CNN with fully connected CRF for accurate brain lesion segmentation. Medical Image Analysis, 36, 61-78. doi: 10.1016/j.media.2016.10.004
20. Islam, J., Zhang, Y., Xie, W., & Yang, J. (2019). Brain tumor segmentation in multi-modal MR images based on fine-tuned deep neural networks. Computers in Biology and Medicine, 110, 20-31. doi: 10.1016/j.compbiomed.2019.04.017
21. Menze, B. H., Jakab, A., Bauer, S., Kalpathy-Cramer, J., Farahani, K., Kirby, J., ... & Kaus, M. R. (2015). The multimodal brain tumor image segmentation benchmark (BRATS). IEEE Transactions on Medical Imaging, 34(10), 1993-2024. doi: 10.1109/TMI.2014.2377694
22. Wang, G., Li, W., Ourselin, S., & Vercauteren, T. (2019). Automatic brain tumor segmentation using cascaded anisotropic convolutional neural networks. In International Conference on Medical Image Computing and Computer-Assisted Intervention (pp. 127-135). Springer, Cham. doi: 10.1007/978-3-030-32254-0_14
23. Sun, C., et al. (2018). Automatically diagnosing Alzheimer's disease from MRI scans by a deep transfer learning-based method. Frontiers in Aging Neuroscience, 10, 439.
24. Abdullah-Al-Wadud, M., et al. (2018). Brain tumor classification from multi-modal MRI using CNN. Proceedings of the International Joint Conference on Neural Networks, 1-8.
25. He, K., et al. (2015). Deep residual learning for image recognition. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 770-778.
26. Krizhevsky, A., et al. (2012). ImageNet classification with deep convolutional neural networks. Advances in Neural Information Processing Systems (NIPS), 25, 1097-1105.
27. Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.
28. Szegedy, C., et al. (2015). Going deeper with convolutions.