Brain Tumor Classification using Deep Learning Algorithms

Ankita Kadam[1], Sartaj Bhuvaji[2], Sujit Deshpande[3]
[1, 2]UG Student, Department of Computer Engineering P. E. S's Modern College of Engineering, Pune, India
[3]Assistant Professor, Department of Computer Engineering P. E. S's Modern College of Engineering, Pune, India

Abstract: A Brain tumor is one aggressive disease. An estimated more than 84,000 people will receive a primary brain tumor diagnosis in 2021 and an estimated 18,600 people will die from a malignant brain tumor (brain cancer) in 2021.[8] The best technique to detect brain tumors is by using Magnetic Resonance Imaging (MRI). More than any other cancer, brain tumors can have lasting and life-altering physical, cognitive, and psychological impacts on a patient's life and hence faster diagnosis and best treatment plan should be devised to improve the life expectancy and well-being of these patients.
Neural networks have shown colossal accuracy in image classification and segmentation problems. In this paper, we propose comparative studies of various deep learning models based on different types of Neural Networks(ANN, CNN, TL) to firstly identify brain tumors and then classify them into Benign Tumor, Malignant Tumor or Pituitary Tumor. The data set used holds 3190 images on T1-weighted contrast-enhanced images which were cleaned and augmented. The best ANN model concluded with an accuracy of 78% and the best CNN model consisting of 3 convolution layers had an accuracy of 90%. The VGG16(re-trained on the dataset) model surpasses other ANN, CNN, TL models for multi-class tumor classification. This proposed network achieves significantly better performance with a validation accuracy of 94% and an F1-Score of 91.
Keywords: Artificial Neural Network(ANN), Convolution Neural Network (CNN), Transfer Learning(TL), Magnetic Resonance Imaging(MRI.)

I. INTRODUCTION

A brain tumor, known as an intracranial tumor, is a surplus growth of brain tissue in which cells grow and multiply uncontrollably, seemingly unchecked by the immune system. The foremost reason the human body is unable to recognize and fight these cells is that it cannot identify them as foreign objects. This is because these types of cells encompass the patient's DNA, which the body's immune system recognizes as natural. The World Health Organization (WHO) classifies brain tumors as grade I-IV. Where benign tumors are Grade I which are slow-growing, least harmful, and easily curable. Malignant Tumor is Grade III. These types of tumors tend to be a bit infiltrative and have chances of recurrence at a higher grade too. A pituitary tumor is a growth of abnormal cells in the tissues of the pituitary gland. While almost all Pituitary Tumors are benign because they don't spread to other parts of the body, as cancers can, they are close to the brain and may invade the central nervous system (CNS) or to other parts of the body. There is no standard grading system for pituitary tumors unlike the other above-mentioned.

As explained above, early detection of these brain tumors is extremely important as these tumors tend to metastasize and grow rapidly. Furthermore, after identification, the classification stage may be a convoluted and tedious task for physicians or radiologists in some complicated cases and completely depends on the availability of expert physicians and radiologists, which is difficult to fulfill in underdeveloped and few developing regions around the globe. These cases need experts to work on, localize the tumor, compare tumor tissues with adjacent regions, apply various filters on the image if necessary; make it more clear for human vision, and finally conclude; whether it is a tumor besides its type and grade if available.

This quick and accurate detection can be achieved through groundbreaking technological advancement in the field of Artificial Intelligence, which has shown promises of higher accuracies in the field of computer vision, image classification, and image segmentation. Deep learning is a subdivision of Machine Learning where a system of neural networks which mimic the human brain structure are trained with huge amounts of data. These types of supervised, semi-supervised, or unsupervised networks have shown tremendous potential in the fields of medical image analysis. These networks are divided into several layers, where the first layer is known as the input layer, the internal layers are known as the hidden layers and the final layer is known as the output layer. Deep learning algorithms utilize these arrangements of numerous layers of the network for feature extraction and encoding. The output of each sequential layer is the input of the succeeding one, and that helps in data abstraction as we go deep within the network. Artificial Neural Networks, Convolution Neural Networks are two popular types of neural nets that are widely used in the industry today. Generally, CNN's are preferred for image classification tasks as they implement feature selection that happens through convolving filters and pooling with the input patterns followed by a selection of the most distinguishing features and then start to train the layers of the classification network.

In this paper, we have provided a comparative study of both ANN’s, CNN’s for brain tumor classification using the same augmented dataset. The original dataset had a total of 3190 images which were augmented and cleaned to 66960. We have designed and trained varied ANN, CNN models including various pre-trained Transfer Learning(TL) models for a comparative and exhaustive study of how different model types of network architectures compare against the four-class classification problem on the same dataset. We concluded with the best ANN model with 1 layer of 128 nodes had a test accuracy of 80%, test loss of 181.14, and F1 Score 80. The best CNN model with 3 Convolution layers, 1 dense layer of 128 nodes had test accuracy of 90%, test loss of 0.43, and F1 Score 91. The best TL model was VGG16 initialized with weights of imagenet, which was re-trained on our augmented dataset had test accuracy of 94%, test loss of 0.81, and F1 Score 94.

II. LITERATURE REVIEW

In the work ‘Brain Tumor Classification Using Convolutional Neural Networks’ proposed by J. Seetha and S. Selvakumar Raja[1] the automatic brain tumor detection is performed by using Fuzzy C Means (FCM) based segmentation, texture, and shape feature extraction and SVM and DNN based classification is carried out. They had used the ‘image net’ database for classification. So the training was performed for only the final layer. They concluded with a training accuracy of 97.5%

Ali Ari, Davut Hanbay[2] proposed a ‘Deep learning-based brain tumor classification and detection system’. The system is implemented to distinguish the tumors into two classes: benign and malignant. The proposed system has three stages, which are preprocessing the extreme learning machine local receptive fields (ELM-LRF) based tumor classification, and image processing based tumor region extraction. The system is restricted to only cranial MR images, which have a mass. The experimental system has achieved an accuracy of 97.18 %

From the Department of Biomedical Engineering, Helwan University, Cairo, Egypt the authors Hossam H. Sultan, Nancy M. Salem, Walid Al-Atabany[3] proposed ‘ Multi Classification of Brain Tumor Images’. In the work done the authors used CNN to classify three types of brain tumors are meningioma, glioma, and pituitary tumor, and identify their types and grades. Their first dataset included MRIs with three different views: axial, coronal and sagittal views. The second dataset had images on T1-weighted contrast-enhanced that include different grades of glioma (Grade II, Grade III, and Grade IV). Their proposed network had 16 layers starting from the input layer which holds the preprocessed images passing through the convolution layers and their activation functions (3 convolution, 3 ReLU, normalization and 3 Max pooling layers). They also had two dropout layers, a softmax layer and finally a classification layer that produces the predicted class. This proposed architecture had achieved an accuracy of 98.7%

III. ARCHITECTURE DIAGRAM

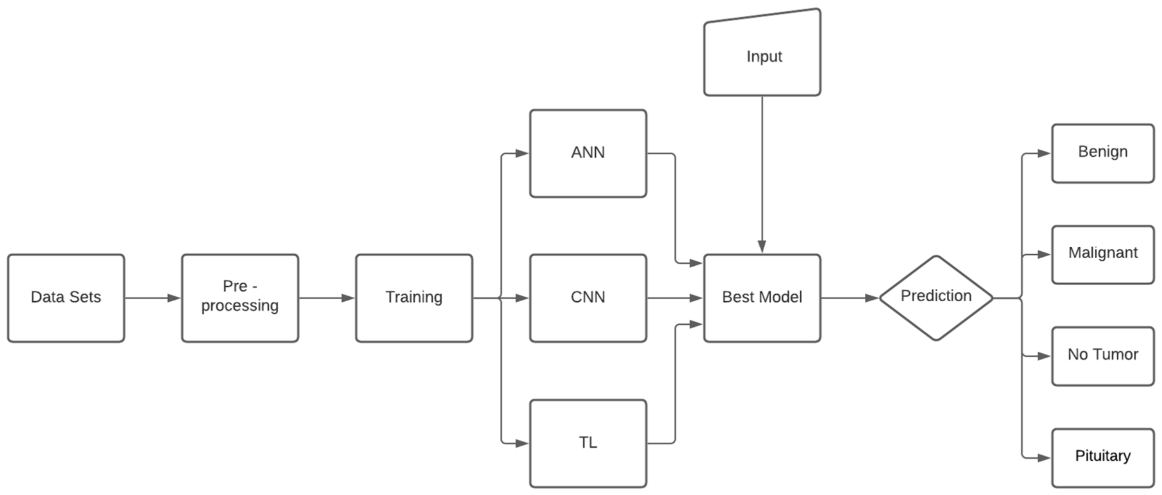

Figure 1: System Architecture

Figure 1 shows the complete system architecture diagram of the proposed method.

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 two-dimensional 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.

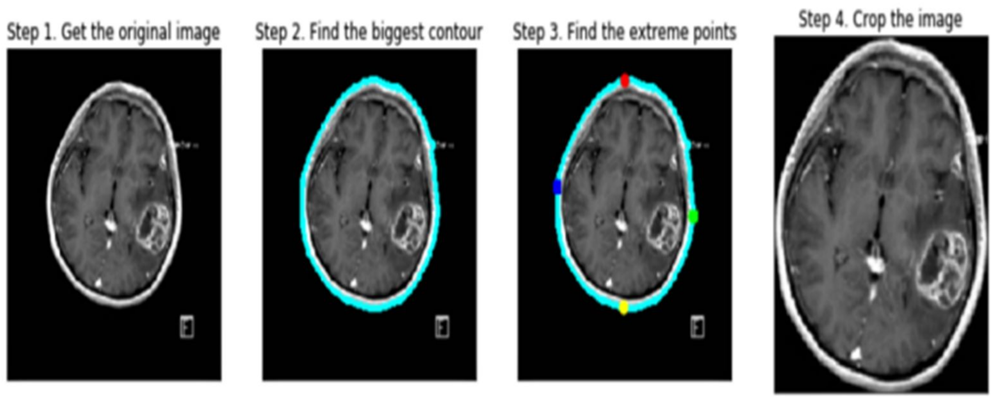
Step 1. Get the original image
Step 2. Find the biggest contour
Step 3. Find the extreme points
Step 4. Crop the image

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.

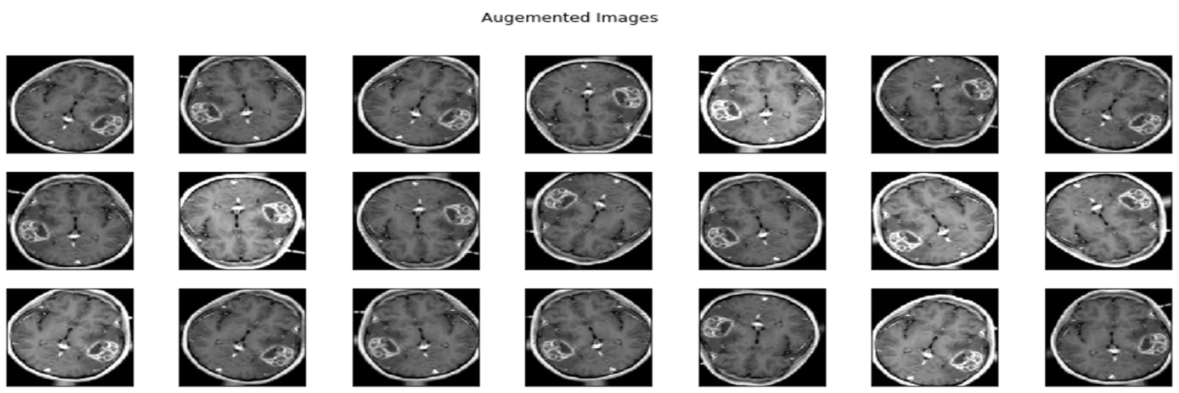

Figure 3: Augmentation

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.

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.429
Volume 9 Issue XII Dec 2021- Available at www.ijraset.com

V. APPROACH

We built and trained multiple models using ANN, CNN, TL 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.

A. *Artificial Neural Network*

Artificial Neural Networks are multi-layer fully-connected neural nets. They consist of an input layer, multiple hidden layers, and an output layer. Every node in one layer is connected to every other node in the next layer.

The training procedure for the neural network works as follows: Firstly we define the structure of the network; starting with the input layer followed by dense layers and finally an output layer. Then the input data is fed from the input layer and flows from top to bottom of the network architecture. Initially, the weights for all the nodes are randomly initialized. For every training example, perform a forward pass using the current weights and calculate the output of each node going from top to bottom. The final output is the value in the last layer. Compare this ultimate output with the actual target label in the training data, and measure the error using a loss function. Perform a backward pass propagating the error calculated from bottom to top of the network to every individual node using the technique of backpropagation. Calculate each weight’s contribution to the error, and adjust the weights accordingly using gradient descent. With multiple such epochs of reading the data, calculating the error, back-propagating the error as feedback, and then adjusting weights according to decrease the error, the network learns features of the data set.

Artificial neural networks use backpropagation as a learning algorithm to compute a gradient descent with relevancy weights. Desired outputs are compared to achieved system outputs, then the systems are tuned by adjusting connection weights to narrow the difference between the two as much as possible. Because backpropagation requires a known, desired output for every input value to calculate the loss function gradient.

For our paper, we trained a total of nine ANN models each with different network architecture. The detailed explanation of the best network is as follows: The input layer is fed an image of resolution 224x224x3. Then we have a Flattening Layer which removes all of the dimensions except for one. A flatten operation on a tensor reshapes the tensor to have a shape that is equal to the number of elements contained in tensor non including the batch dimension. After which we have a fully connected dense layer and then an activation function called ReLu. After the activation layer, we have a dropout to avoid overfitting. Finally, there is the classification layer which has four nodes, each representing one class output. The best ANN model had an accuracy of 78% and a loss of 181.14. High values of loss are bad for any model. From the results(Table 4) we can see ANN models are not that great when it comes to image classification, as these models do not have any layer for feature extraction. Hence these models concluded with low accuracies and high loss values on testing data. Below is the representation of the best ANN model along with the table containing information on all our ANN models.

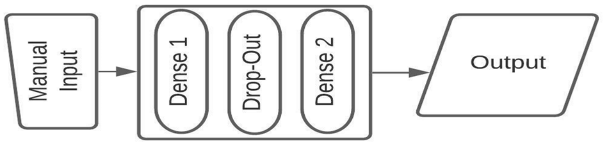

Figure 2: ANN Model Architecture

ANN Models	Test Accuracy	Test Loss	F1 - Score
32-nodes-0-dense	76	213.54	76
64-nodes-0-dense	77	229.56	77
128-nodes-0-dense	78	181.14	78
32-nodes-1-dense	24	1.38	10
64-nodes-1-dense	25	1.38	13

128-nodes-1-dense	25	1.39	10
32-nodes-2-dense	26	1.33	10
64-nodes-2-dense	24	1.3	10
128-nodes-2-dense	25	1.38	10

Table 4: ANN Models

B. *Convolution Neural Network*

Convolutional Neural Networks (CNN) is one of the variants of neural networks which have shown unprecedented accuracies in the field of image classification. The hidden layers of a CNN typically consist of convolutional layers, pooling layers, fully connected layers(dense layers), and normalization layers. A Convolution layer reads the input image and applies multiple filters to the image and a feature map is created. All these filters are initialized randomly and become our parameters which will be learned by the network subsequently during backpropagation. These filters help in extracting the right and relevant features from the input data. CNN captures the spatial features from an image. Spatial features refer to the arrangement of pixels and the relationship between them in an image. They help in identifying the object accurately, the location of an object, as well as its relation with other objects in an image. After this, it applies a ReLU function on the feature map, which helps increase non-linearity. Next, it applies a pooling layer to each feature map, which progressively reduces the spatial size of the representation to reduce the number of parameters and computation in the network. Then it flattens the pooled images into one long vector which is input for a fully connected artificial neural network. Further, the network works exactly as an ANN, where output is generated in the final layer and error is backpropagated into the network. This error then helps the network adjust the weights to minimize the error in the next consecutive epochs. For this paper, we designed and trained 27 different CNN models on our augmented data set. Figure 3 shows our proposed CNN model which is named 3-conv-128-nodes-1-dense and which has a total of 16 layers. The model begins with an input layer, which accepts images in a batch size of 32 with size 224x224x3. Then we have 3 convolution layers along with their activation functions and max-pooling layers. To prevent overfitting, a dropout layer is used followed by a fully connected layer of 128 nodes and a softmax layer to predict the output and finally a classification layer that produces output in one of the four predicted classes.

The detailed explanation of the network is as follows: The input layer reads an image in resolution of 224x224x3 and sends it down to one of the three convolution layers. The convolution layers are a bundle of convolution, activation and max-pooling layers. Firstly the convolution layer applies sliding K convolutional filters (kernels) of size (M × N) over the input images by moving the filters along the input and compute the dot product of the weights and the input. These kernels are used as features identifiers; such that kernels in the early layers detect only low-level features like (outlines, boundaries and spots), while advanced ones are used to detect more and more complex features. Each of our three convolution layers is succeeded by an Activation Layer - the activation function of a node defines the output of that node given an input or set of inputs. For our model, we have used Rectified Linear Unit (ReLu). Next in the network, we have a Pooling layer. The addition of a pooling layer is used for ordering layers within a neural network that may be repeated one or more times in a given model. The pooling layer operates upon each feature map separately and creates a new set of the same number of pooled feature maps, but the pooling layer will always reduce the size of each feature map by the pooling factor. In Max Pooling, each of the regions is represented by the filter, and we take the max of that region and create a new output matrix where each element in the new matrix is the max of a region filter from the original input. After our final max-pooling of the third convolution layer set, the output is of the size 17x17x128. In a neural network, a fully connected layer occupies most of the parameters and hence, neurons develop a co-dependency amongst each other during training which dampers the individual power of each neuron, leading to over-fitting of training data. A dropout is an approach of regularization in neural networks which helps reduce interdependent learning amongst the neurons in the fully connected layers.

Hence after max-pooling, we have a Dropout layer. When we have such a layer during training, some number of layer outputs are randomly ignored or “dropped out.” This has the effect of making the network architecture look-like and be treated-like layers with a different number of nodes and connectivity to the prior layer. In effect, each update to a layer during training is performed with a different “view” of the configured layer. Finally, we conclude the network with a fully connected layer of 128 nodes followed by the activation layer ‘softmax’ which gets us output in one of the four classes. The accuracy of our best CNN model was 90% and the value of the loss was 0.43.

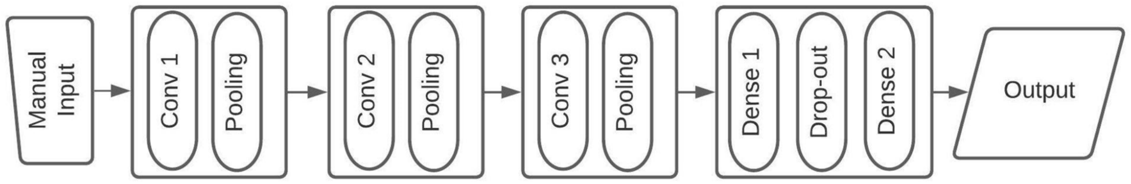

Figure 3: CNN Model Architecture

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

C. Transfer Learning

Neural networks are adaptive in nature and can be used for multiple tasks. Such a technique where a model trained on one task is re-purposed for a second task is known as Transfer Learning. A pre-trained model is chosen which was previously used for a similar task as the one we want to implement with the network for the defined purpose.

The pre-trained model can then be used as the starting point for a model on the second task of interest. This may involve using all or parts of the model(layers of the model), depending on the modeling technique used. Optionally, the model structure can be adapted or refined on the input-output pair data available for the task of interest. Here we can choose to add or subtract different layers in the model. Pre-trained models are available for everyone to use through different means. The famous deep learning Python library, Keras, provides an interface to download some popular models which comprise the network architecture and the trained weights of the neurons. These models are all CNN models. In our case, this approach is effective, because the models were trained on eclectic photographs and require the model to make predictions on a relatively large number of classes, in turn, requiring that the model efficiently learn to extract features from photographs to perform well on the problem.

For this paper, we trained five different TL models. Out of which the model '**VGG16**' had the highest accuracy. VGG16 is a convolutional neural network model proposed by K. Simonyan and A. Zisserman from the University of Oxford in the paper "Very Deep Convolutional Networks for Large-Scale Image Recognition". The model achieves 92.7% top-5 test accuracy in ImageNet, which is a dataset of over 14 million images belonging to 1000 classes. The input to the VGG16 model is images with dimensions of (224,224,3). These images first enter two of the many convolution layers. Then followed by a ReLu and max-pooling we enter the second set of two convolution layers. Then again after ReLu and pooling the data enter 3 sets of 3 layers of convolution and pooling. Then we have 3 fully connected layers. The output of the final layer is 1000 channels for 1000 classes of ILSVRC challenge, then after the output of the 3rd fully connected layer is passed to softmax layer to normalize the classification vector. For VGG16 to work on our data set we did a simple modification of replacing the final layer of 1000 class output with 4 class output. The model was architecture and associated weights of the network were loaded from Keras. After which we trained the model on our data set. After training was completed, the model was tested and we noted an accuracy of 94%, loss of 0.81 and F1 Score 94.

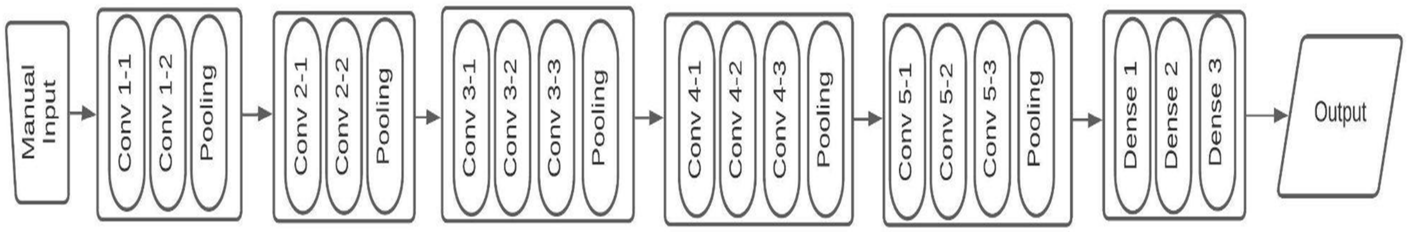

Figure 4 : TL Model Architecture

TL Models	Test Accuracy	Test Loss	F1-Score
VGG16	94	0.81	94
InceptionV3	27	1.40	19
VGG19	47	0.46	33
ResNet50	88	0.41	88
MobileNet_v2	37	1.73	30

Table 6 : TL 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 cross-entropy. 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 pre-processing, augmentation and pickling was around 50 hours and the total training time was 10 hours.

VII. CONCLUSION

In the following work, we represented different types of neural network structures and discussed how Convolution Neural Networks outperform simple Artificial Neural Networks for medical image classification of Brain Tumors into given four classes of: No Tumor, Benign Tumor, Malignant Tumor or Pituitary Tumor. Our dataset consisted of 3190 two-dimensional T1-weighted contrast-enhanced images which were then pre-processed, cropped and augmented. The dataset included three different views of MRI: axial, coronal and sagittal views. After training multiple models on this cleaned augmented data, the best ANN model had an accuracy of 78%, loss of 181.14 and a F1 Score of 78. While out of the total 27 CNN models that we trained, the best CNN model with 3 convolution layers had an accuracy of 90%, loss of 0.43 and F1 Score of 91. We also re-trained some pre-existing models from Keras library. Out of which the best TL model was - VGG16 which was re-trained on our augmented dataset had the highest accuracy among all models. On test data, VGG16 had the accuracy of 94%, a loss value of 0.81 and F1 Score of 94. Thus from the above results, we can conclude that models with convolution layers(CNN, TL) are better than models without convolution layers(ANN) for medical image classification of brain tumors using MRI images.

REFERENCES

[1] J. Seetha and S.Selvakumar Raja," Brain Tumor Classification Using Convolutional Neural Networks" in Biomedical Pharmacology Journal, September 2018, Vol. 11(3), p. 1457-1461. http://biomedpharmajournal.org/vol11no3/brain-tumor-classificationusing-convolutional-neural-networks/
[2] Ali Ari and Davyut Hanbay, "Deep learning based brain tumor classification and detection system" in Turkish Journal of Electrical Engineering Computer Sciences,2018, 26: 2275 – 2286. https://journals.tubitak.gov.tr/elektrik/abstract.htm?id=23203
[3] Hossam H. Sultan , Nancy M. Salem and Walid Al-atabany, "MultiClassification of Brain Tumor Images Using Deep Neural Network" IEEE,2019. https://ieeexplore.ieee.org/abstract/document/8723045
[4] Kaiming He Xiangyu Zhang Shaoqing Ren Jian Sun "Deep Residual Learning for Image Recognition" https://arxiv.org/pdf/1512.03385.pdf
[5] https://braintumor.org/brain-tumor-information/brain-tumor-facts/#quick-facts
[6] Sambasivarao. K "Region of Interest Pooling" https://towardsdatascience.com/region-of-interest-pooling-f7c637f409af

[7] Karan Chauhan1, Shrwan Ram2 , ”Image Classification with Deep Learning and Comparison between Different Convolutional Neural Network Structures using Tensorflow and Keras”, International Journal of Advance Engineering and Research Development. (Volume 5, Issue 02, February -2018)
[8] Himanshu Rawlani ”Visual Interpretability for Convolutional Neural Networks” https://towardsdatascience.com/visual-interpretability-for-convolutional-neural-networks-2453856210ce
[9] Karen Simonyan and Andrew Zisserman ”Very deep convolutional networks for large-scale image recognition” https://arxiv.org/pdf/1409.1556.pdf
[10] Sentdex : ”Machine Learning with python” https://www.youtube.com/user/sentdex
[11] Kaggle - Data set Brain Tumor Classification (MRI) | Kaggle
[12] Karl Weiss, Taghi M. Khoshgoftaar, DingDing Wang ”A survey of transfer learning” A survey of transfer learning | Journal of Big Data | Full Text (springeropen.com)