Brain Tumor Detection

Medical Image Diagnosis using Convolutional Neural Network

Prexa Kamani
Computer Engineering, COE
CSULB
prexakamani.01@student.csulb.edu

Zarana Nakrani
Computer Engineering, COE
CSULB
zaranasanjaykumar.nakrani01@student.csulb.edu

Prof. Wenlu Zhang
Computer Engineering, COE
CSULB
wenlu.zhang@csulb.edu

I. Abstract

The proposed Project delve into the application of Convolutional language Network along with transfer learning to boost the performance of the image classification algorithm on Brain Tumor Dataset dataset. The dataset has 7000 images with 4 different classes. Transfer Learning, in turn, is a transferred knowledge from pre-trained models to exposing core concepts from a new data set in order to cut down the amount of required training data. This approach is the fine tuning a CNN prepared on the dataset, optimization hyper parameters and evaluation performance. We will conduct experiments in order to show that transfer learning is the most suitable way to use convolutional neural networks for feature learning for the image classification.

II. Introduction

Medical Imaging is a critical component of modern healthcare that can aid medical professionals to make more informed diagnostic decisions. Deep Learning shown great potential in aiding the health community in the detection and diagnosis of **brain tumors**. By leveraging Transfer learning algorithms, we can analyze medical images, such as **MRI** or **CT scans**, with unprecedented accuracy and efficiency. Also, it can assist in the classification of brain tumors into different subtypes. By training models on large datasets of labeled brain tumor images, deep learning algorithms can learn to distinguish between various tumor types, such as gliomas, meningiomas, or metastatic tumors. This classification capability can aid in determining the appropriate treatment approach and prognosis for patients.

We seek to train advanced Convolutional Neural Networks (CNNs) (VGG-16, ResNet50V2, DenseNet and CNN Sequential Model), with Brain Tumor images, to compare the performance metrics of these AI models to identify the underlying trends in the data and to diagnose abnormalities accurately.

A. Literature Review

Traditional Computer vision (CV) techniques have shown several limitations with respect to accuracy and performance. One of the major drawbacks is their reliance on custom or handcrafted features. In contrast, deep learning approaches such as CNNs can automatically learn relevant features directly from the image data and are superb when it comes to feature extraction.

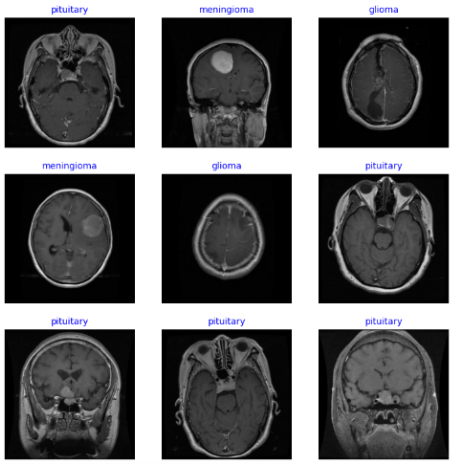

Fig. 1. Random Image Sample from Brain Tumor Dataset

B. Expected Results

In this project we are going to try various types of object-sorting programs and see how they behave on our task; we'll also play with ways to configure them to fit particular challenges, and see how much better they get when we use transfer learning.

The goal is to evaluate the models based on their metrics and draw insights into their performance on a particular dataset. Since the VGG-16 and DenseNet have deep architectures we have seen them to learn more features and perform better than the ResNet50V2 and Mymodel. This can be verified by comparing the Accuracy, Precision, Recall, and F-Score of the models.

III. Dataset

To perform disease classification from Brain Tumor, we chose the datasets publicly available from Kaggle namely, the Brain Tumor MRI Dataset.

*1) **Dataset**:* This dataset is a combination of the following three datasets : figshare, SARTAJ , Br35H .

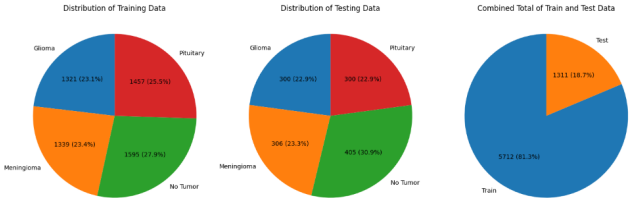

Fig. 2. Dataset Distribution

This dataset contains **7023** images of human brain MRI images which are classified into 4 classes: **Glioma**, **Meningioma**, **No tumor** and **Pituitary**.

IV. Proposed Methodology

The models used have the following characteristics:

Custom CNN: In the Custom CNN model, I begin by processing a 224 × 224 pixel input image through an initial convolutional layer with 32 filters and a resulting 224 × 224 × 32 feature map. The layer employs a 3 × 3 kernel to identify generic features. The output from this convolutional layer is then passed to a max-pooling layer, resulting in a 111 × 111 × 32 feature map as shown in as shown in "Fig. 5". Max-pooling selects the highest values from the filter-covered area. Subsequently, we apply additional convolutional layers: one with 64 filter values (resulting in a 109 × 109 × 64 feature map), another with 128 filter values, and a third with 256 filter values. Each of these convolutional layers is followed by a max-pooling layer, downsampling the feature maps. The final max-pooling layer produces a 6 × 6 × 32 feature map. After flattening the output, we feed it into a 512-dimensional fully connected dense layer with ReLU activation. A dropout layer with a rate of 0.5 is applied before the final output layer, which uses a softmax activation function.

DenseNet: It is a powerful deep learning architecture that improves on Convolutional Neural Networks (CNNs) through dense connectivity between layers.In DenseNet, each layer is directly connected to all its subsequent layers in such a way that there is a direct path from any one layer to any other. I creates a DenseNet121 model by going through the tensorflow.keras.applications module. These weights have been pre-trained using data from ImageNet (weights='imagenet'). After that, the DenseNet model's outcome goes through a Global-AveragePooling2D layer to decrease the spatial dimensions of the 3D tensors; this is then followed by an operation where this pooled output will be passed on into another dense layer having 256 cells and activation function as ReLU which takes care of recognizing intricate patterns contained within it. This last portion contains two parts; one being just the dense layer itself with 4 units in correspondence with each class that exist while the other fulfills role pertaining only at output level by providing four units alongside using a softmax activation function.

ResNet50V2: ResNet50V2 is effective for small to medi-umsized datasets and can achieve high accuracy with relatively

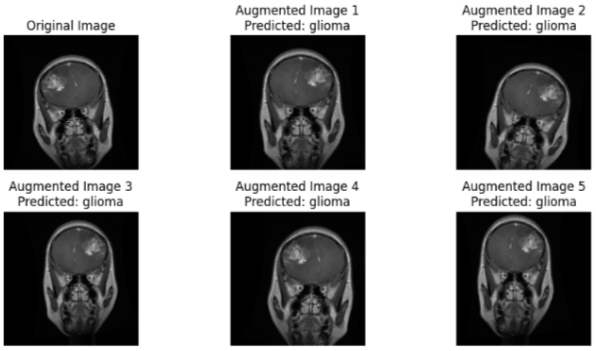

Fig. 3. Augmented Image

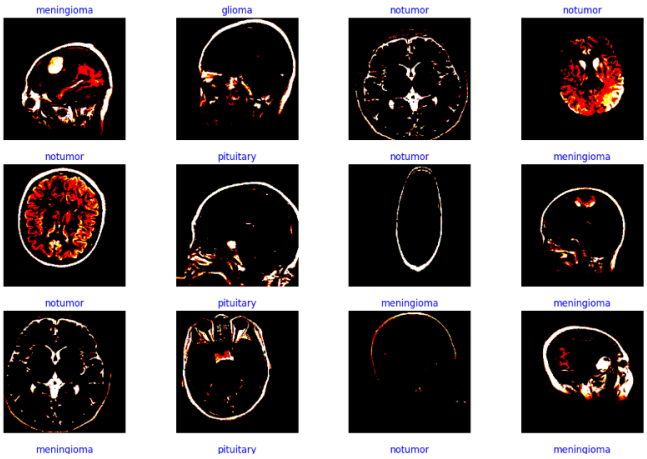

Fig. 4. Image Data Generator

low computational resources. It is known for its ability to overcome the vanishing gradient problem. It has 11,689,512 parameters, with 18 layers, and requires an input size of 224*224.

VGG-16: VGG-16, a convolutional neural network architecture, boasts 138 million parameters distributed across 16 layers, including convolutional and fully connected layers as shown in "Fig. 6" . With a straightforward design of repeating 3x3 convolutional filters and max-pooling layers, it excels in image classification tasks on medium to large datasets, requiring input images resized to 224x224 pixels. This architecture uses domain adaptation with a pre-trained model as a fine-tuning goal. Epoch, model checkpoints, early stopping, all to improve efficiency of the process, and reduce the possibility of overfitting. Also it is involved in the pipeline of data loading and processing that include data generators as shown in 'Fig. 16" to help with the efficient batch process for instance rotation, width and height shifts, shear, zoom, and horizontal flipping for the data augmentation as shown in 'Fig. 3" to increase diversity of the train data. the newspaper causes positive outcomes that become evident in the graph below, 'Fig. 4" .

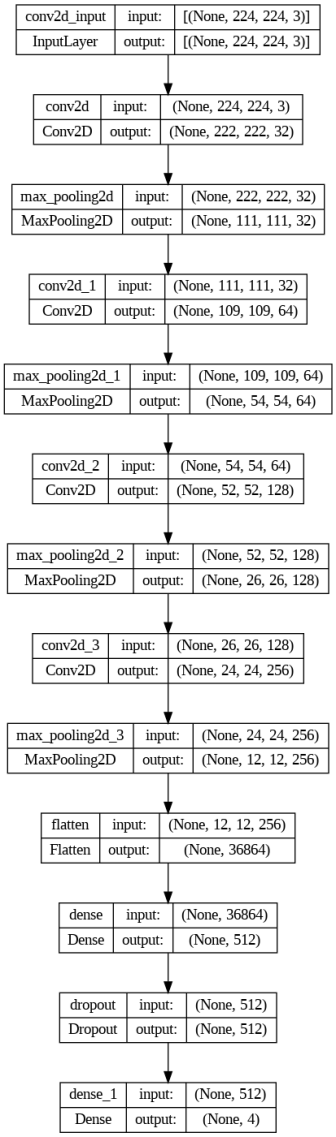

Fig. 5. Custom CNN Model

V. Experimental Setup

The 52 Gb RAM Google Colab platform based on a cloud was used to setup our environment. It was supported by T4, K80 NVIDIA Tesla and P100 GPU's. On this sophisticated platform training machine learning algorithms becomes more faster and efficient.

To set up the experiment:

1) Download the Brain Tumor MRI dataset from Kaggle.
2) Upload the downloaded dataset to your Google Drive.
3) Open Google Colab, mount your Google Drive to access the uploaded dataset.
4) Begin training your model using the dataset available in your Google Drive. You can access the dataset using the path where you uploaded it.

By following these steps, you can efficiently set up the experimental environment in Google Colab for training the

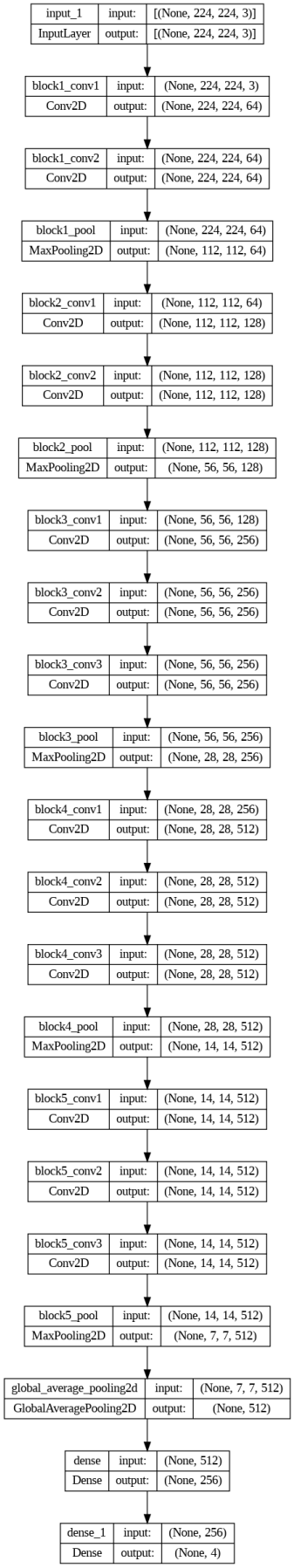

Fig. 6. VGG - 16

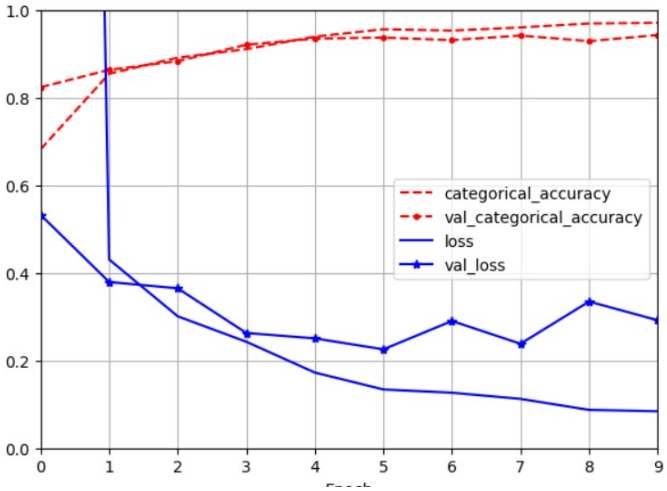

Fig. 7. Custom CNN loss and accuracy

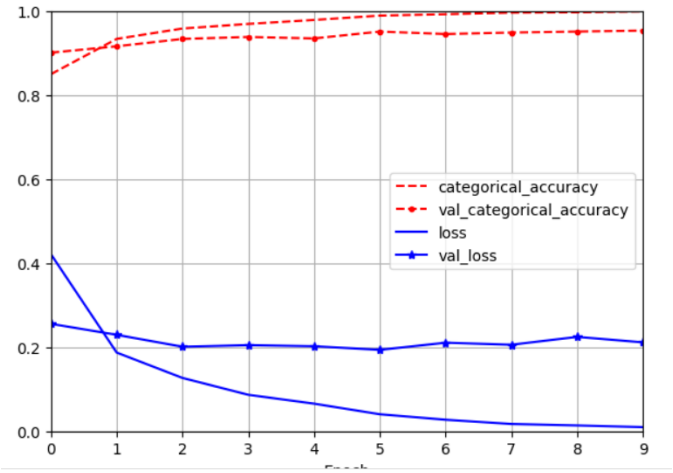

Fig. 8. DenseNet121 loss and accuracy

model.

A. *Hyperparameters*

The following Hyperparameters were used for training the models:

- Cross Entropy Loss function is used to calculate the loss between predicted and actual output.
- Early stopping and reduction callbacks of learning rate were used to prevent overfitting and optimize the training process.
- Optimizer: Adam optimizer is used to regulate the model.
- Batch Size: 32
- Epoch: 10

Drive Link for my trained model and dataset:
DatasetandTrainedModel

VI. Measurement

A. Training and Validation Measures

After the models are trained, loss and accuracy were measured on the training data and validation data. Below figures shows the graphical representation of training and validation loss and training and validation accuracy. **Custom CNN Accuracy :** shown in "Fig. 7' **DenseNet Accuracy :** shown in "Fig. 8'

The separate Loss and Accuracy graph for VGG - 16 and Resnet50V2 shown in "Fig. 12' and shown in "Fig. 10' :

The Confusion and Classification matrix for VGG 16 and Resnet50V2 shown in "Fig. ??' and shown in "Fig. 13' :

B. Test set matrices

The trained models are then tested on the test data set. The matrices that are collected are accuracy, F1 score, precision, and recall. Confusion matrix is also plotted and based on that test set matrices are measured.

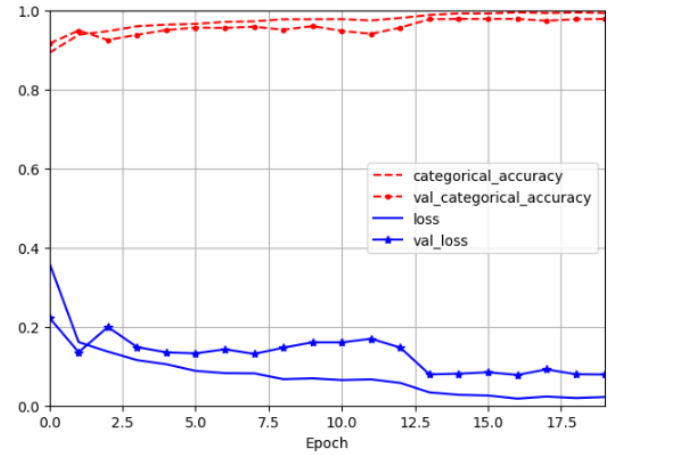

Fig. 9. VGG16 loss and accuracy

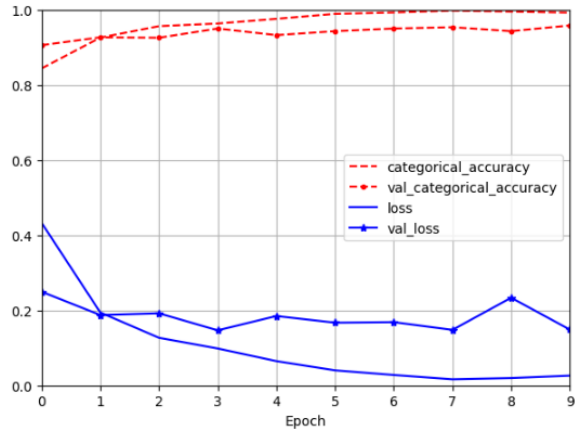

Fig. 10. ResNet loss and accuracy

Fig. 11. Resnet 50V2 Accuracy and Loss

Fig. 12. Vgg 16 Accuracy and Loss

Classification Report for ResNet50V2:

	precision	recall	f1-score	support
0	0.93	0.91	0.92	310
1	0.90	0.93	0.91	306
2	1.00	0.99	1.00	405
3	0.98	0.98	0.98	300
accuracy			0.96	1321
macro avg	0.95	0.95	0.95	1321
weighted avg	0.96	0.96	0.96	1321

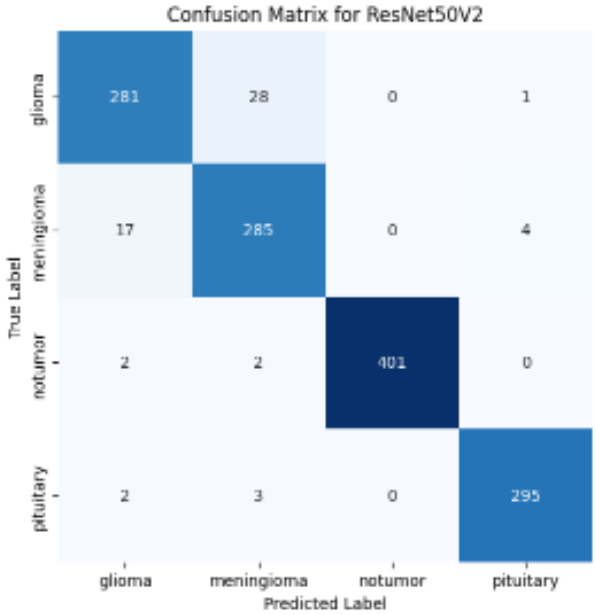

Fig. 13. Resnet Matrix

- Precision: This metric is calculated as the number of classes correctly predicted divided by the sum of the classes correctly predicted and the classes that were incorrectly predicted as correct.
- Recall: This metric is calculated as the number of classes correctly predicted divided by the sum of the classes correctly predicted and the classes that were incorrectly labeled as incorrect.
- F1-score: This is the harmonic mean of Precision and Recall and is calculated as 2 * (Precision * Recall) / (Precision + Recall)
- Accuracy: This metric is calculated as the number of classes correctly predicted divided by the total number of classes.

VII. Results Analysis, Intuitions and Comparison

Custom CNN Model: The Custom CNN Model is finally trained with deep two-dense layers with (ReLU) activation functions, followed by a dropout layer and then the final dense layer with softmax activation function that returns probabilities for categorical into one of four classes. The model includes optimizer and loss function, and metrics are used to evaluate the training results in 10 times epochs. Callbacks for monitoring precision and recall scores are also provided. Scores

Classification Report for VGG - 16:

	precision	recall	f1-score	support
0	0.99	0.94	0.96	310
1	0.94	0.98	0.96	306
2	1.00	1.00	1.00	405
3	1.00	0.99	0.99	300
accuracy			0.98	1321
macro avg	0.98	0.98	0.98	1321
weighted avg	0.98	0.98	0.98	1321

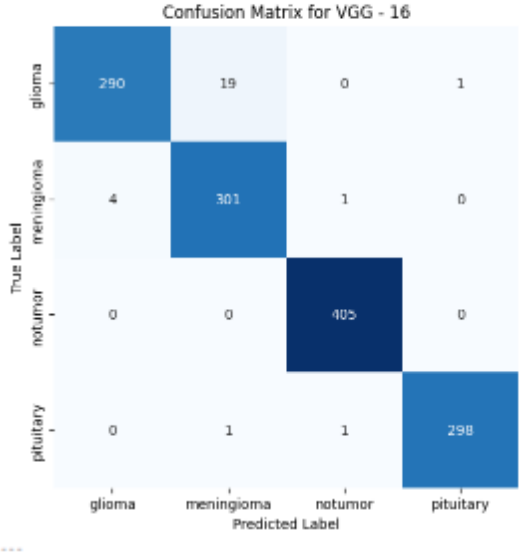

Fig. 14. VGG16 Matrix

42/42 [==============================] - 6s 133ms/step - loss: 0.2649 - categorical_accuracy: 0.9273

Fig. 15. CNN Accuracy

of high precision and recall have been shown across classes indicating to balanced performance by acheving accuracy of 92% as shown in "Fig. 15' .

Transfer Learning Model: VGG-16 did much better among other transfer learning methods by generating class accuracy of 97% test accuracy. Densenet and Resnet showed as the lower accuracy less than 94% and 95% respectively. One possible explanation for that lesser results might have been the difference between the input size of the models and the smaller images leading to elevated error rates.

With the success of VGG 16 , it is clear that its architecture of efficiency as well as transfer learning generates good results. Subtuning of pre-trained models may be the ultimate step towards even further perfection of performance

Comparison and Intution: Among all the models VGG

42/42 [==============================] - 12s 273ms/step - loss: 0.0768 - categorical_accuracy: 0.9796
VGG - 16 - Test accuracy: 0.9796, Test loss: 0.0768

Fig. 16. VGG 16 Accuracy

42/42 [==============================] - 4s 103ms/step - loss: 0.1290 - categorical_accuracy: 0.9553
ResNet50V2 - Test accuracy: 0.9553, Test loss: 0.1290

Fig. 17. Resnet50v2 Accuracy

performs the outstanding work and stood out amongst all with an accuracy of 97%.

It's architecture is implemented using transfer learning by finetuning the pretrained model. Even defined callbacks for model checkpoints, early stopping to optimize the training process and prevent overfitting. In addition of that, pipeline for loading and processing the data, including data generators for efficient batch processing and data augmentation techniques such as rotation, width and height shifts, shear, zoom, and horizontal flipping in order to increase the diversity of the training data.generates good results as shown in 'Fig. 16" .

In this scenario, transfer learning offers works perfectly well as compared to custom CNN. The pre-trained models rather encode the features learned through the vast contributions of different datasets, which due to their efficient utilization can be very useful in the new tasks. They also align quicker and put less strain on the training datasets due to this method compared to training from scratch.

Whether to use custom CNNs or transfer learning is determined by variables such as size of dataset, degree of similarity of the data used in pre-training to the new dataset, the available resources or need for high accuracy. Regarding CIFAR-10, a well-designed CNN was responsible for obtaining excellent results, and transfer learning is still a valuable approach which should be appropriately considered. Let us examine a myriad of models for elites in this piece. Then, we will look up whether they are correct or full of error and see the loss they have. That's way we can narrow our understanding which one is the best.

VIII. Conclusion

In our exploration of custom architectures for image classification on Brain Tumor Dataset, we delved into the efficacy of four distinct CNNs: Custom CNN model , VGG-16 , Densenet and Resnet50v2. Our results comprised of both Custom model and Transfer learning model. Furthermore, our testing showed the possibility of transfer learning, in which the case of VGG-16 was the most noticeable as this trained model produced the highest scores with 97% . An essential component of the proposed model is the utilization of pre-trained models that was critical in achieving amazing results on Brain Tumor MRI Dataset. Therefore, the reused knowledge in machine learning can be seen as a great edifier.

On from where I stand now, future paths of research look very intriguing. The investigation of while more advanced architecture including ResNet, DenseNet, and EfficientNet is the reflection of various ways to unlock more potential. Secondly, at the same time, pretrained models as tuning solutions are also considered very promising ie. the chance to surpass the performance benchmark. Though the prime attention was about CNNs but it is also noticeable that other methodologies including SVM are worth giving a try as it is also unveiled that these methodologies may yield the competitive accuracy rates.

Therefore, this exercise pursued the objective of gaining insight into several architecture options specific to image

classification methods and reinforced that an effective learning technique lies at the heart of the performance increase.

IX. Team Members Contribution

In our collaborative effort for medical image diagnosis report we are group of 2 team members (Zarana and Prexa).

Prexa Kamani (032169694)

- Implemented **VGG-16 and Resnet50V2** transfer learning by incorporating pre-trained weights to enhance model performance.
- Set up the environment in Colab with Python, TensorFlow and Keras.
- Conducted pre-processing and through analysis of the model before training the model.
- Improved models and fine-tuned hyperparameters to achieve high performance and efficiency.
- Enhanced VGG-16 model scalabilty by adding data augmentation techniques.
- Conducted regressive experiments to compare the results and performance of the models.

Zarana Nakrani (031515313)

- Implemented the **Custom CNN Model and DenseNet**, harnessing their capabilities to further augment our diagnostic capabilities.
- Implemented DenseNet Transfer Learning by utilizing pretrained weights.
- Modified the project code to make it in a modular and reusable form.
- Visualized and Interpreted the results using graph and metrics.

Github Link

Here is the provided showcase work for Brain Tumor Detection Project using Convolutional Neural Network.

Link : ***BrainTumorImageClassificationUsingCNN***

References

1) Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 580–587, 2014
2) Samir S Yadav and Shivajirao M Jadhav. Deep convolutional neural network based medical image classification for disease diagnosis. Journal of Big data, 6(1):1– 18, 2019.