[Team 27] ProjF: Brain Tumor Detection using Convolutional Neural Networks

Britanya Wright (bwright2) Arun Gaonkar (agaonka) Sai Harsha Nadendla (snadend2)

I. MOTIVATION

A brain tumor is one of the most hostile diseases affecting over 700,000 people within the United States alone. Once formed, it has the ability to infiltrate the remaining portions of the brain and spinal cord, resulting in death among adults and children. Caution and care are taken in the accuracy of the diagnosis, treatment, and planning to increase the life expectancy of that patient. Due to such caution being instrumental in the patient's life, radiologists and clinical experts are tasked with the tedious task of ensuring not only detecting the tumor but also segmentation. To reduce the introduction of human error when evaluating these MRI scans, Machine Learning (ML) can be implemented for automatic classification. We want to implement a model with the ability to detect and identify the presence of a brain tumor.

II. DATA SET DESCRIPTION

We have decided to use a data set that is publicly available.

The Brain Tumor Dataset, found on Kagle, consists of 3, 565 raw MRI images with brain tumors in varying locations of the brain [1]. The images are segmented into tumor types: Glioma, Meningioma, No Tumor, and pituitary. This dataset is divided into training and testing sets.

III. METHODOLOGY

The initiation for this problem is at the image classification stage where we have to identify tumor types from MRI images. We will be using a basic Convolutional Neural Network (CNN) as our baseline model and the proposed model will be a variant of this.

A CNN is an algorithm that primarily rose to prominence because of its ability to take input images, assign weights and biases to different parts of the images and differentiate them from one another.

The data is trained by iterating through the training dataset multiple times and weighing the outputs or applying a bias on the neurons based on how close the output is to the expected output.

These types of networks consist of a hidden layer, input layer, and output layer. The input layer is a tensor with shape: (number of inputs) x (input height) x (input width) x (input channels).

There are multiple layers to a CNN architecture. Hidden layers can be explained as layers that perform the convolution (a dot product of the convolution kernel with the input matrix). The purpose of the Max pooling layers reduces the dimensions while combining the outputs into a single output by using the maximum value of each neuron in the feature map. Fully connected layers connect every neuron in one layer to the next layer. Dropout acts as a regularization method to reduce overfitting and improve generalization error. A softmax, or normalization exponential function, in the activation layer, "squishes" each entry within an input to values ranging from (0,1].

1. *Baseline:* We chose to use a paper for our baseline model [6]. This paper experiments with five different variations of CNN architectures to conclusively find a high-performing model for brain tumor classification. Throughout the paper, they experiment with hyperparameter manipulation to affect the validation and training accuracy and loss. Across all their models, they chose to consistently use the adam optimizer.

   The model we chose to use as our base model was their best-performing CNN architecture. The CNN architecture is as followed:

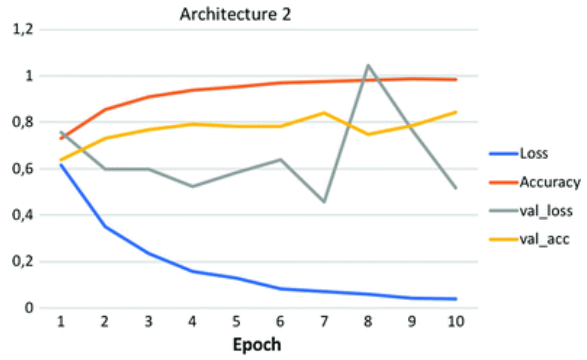
Layer (type)	Output Shape	Param #
conv2d_7 (Conv2D)	(None, 62, 62, 32)	896
max_pooling2d_7 (MaxPooling2	(None, 31, 31, 32)	0
conv2d_8 (Conv2D)	(None, 29, 29, 32)	9248
max_pooling2d_8 (MaxPooling2	(None, 14, 14, 32)	0
flatten_4 (Flatten)	(None, 6272)	0
dense_10 (Dense)	(None, 64)	401472
dense_11 (Dense)	(None, 3)	195

Total params: 411,811
Trainable params: 411,811
Non-trainable params: 0

   This model is split into many layers. Their first layer is a convolution layer with ReLu activation followed by a max-pooling layer. This order is repeated two times before then being flattened. This is followed by two fully connected layers.

The implementation of the architecture produces four parameters that determine the success of the model. Success is defined as a model able to accurately classify the input image by matching its label to the image. The four parameters are the loss and accuracy of the training and validation set. Accuracy is the percent of current guesses. Loss is the error for inaccurate predictions.

Below is a learning curve that is shown by depicting the overall performance of the model.



This model resulted in an 84.2% validation accuracy and a 98.51% training accuracy. The validation and training loss showed a decreasing trend with respect to the number of epochs as they increased. This model had medium performance due to the validation loss not showing a perfect decrease pattern.

2. *Proposed:* Similar to the baseline model, we chose to use the adam optimizer in the learning process because of its ability to handle sparse gradients on noisy problems. We chose to include 4 convolution layers. In addition, we added a dropout layer with a rate of 0.5 sandwiched between 2 dense layers. The CNN architecture is as followed:

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 300, 300, 32)      2432
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 150, 150, 32)      0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 150, 150, 32)      9248
_________________________________________________________________
max_pooling2d_1 (MaxPooling2D) (None, 75, 75, 32)      0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 75, 75, 32)        9248
_________________________________________________________________
max_pooling2d_2 (MaxPooling2D) (None, 37, 37, 32)      0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 37, 37, 64)        18496
_________________________________________________________________
max_pooling2d_3 (MaxPooling2D) (None, 18, 18, 64)      0
_________________________________________________________________
flatten (Flatten)            (None, 20736)             0
_________________________________________________________________
dense (Dense)                (None, 128)               2654336
_________________________________________________________________
dropout (Dropout)            (None, 128)               0
_________________________________________________________________
dense_1 (Dense)              (None, 4)                 516
=================================================================
Total params: 2,694,276
Trainable params: 2,694,276
Non-trainable params: 0
_________________________________________________________________
```

IV. Model Training and Selection

A. Dataset Sampling

Original images from the dataset are resized to (300,300). From these images, batches of tensor image data are obtained using real-time data augmentation from ImageDataGenerator, which is available from the image preprocessing library of Keras. In this library, the images are randomly rotated within the specified rotation range, randomly shifted in width and height. And also zooming range and flipping direction can be specified. For our images, we used the rotation range of 30 degrees, shift range of 0.1, and a zoom range of 0.2. Random horizontal flipping of images is enabled. The resulting image tensors are then split into training and validation sets.

B. Model Training

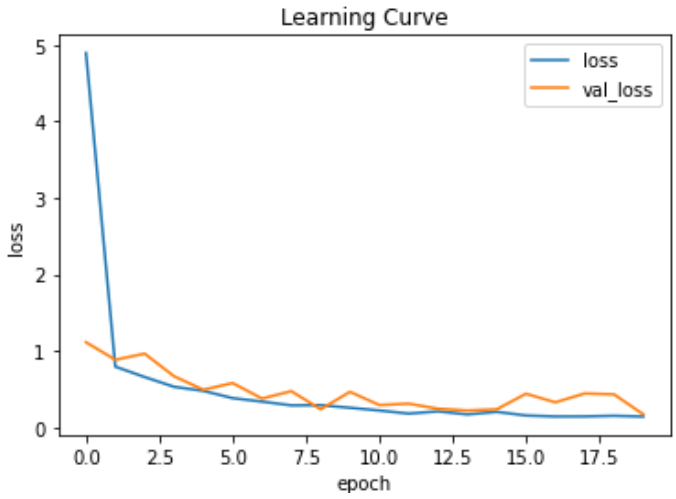
The data that is fed into the CNN model has the shape of (2583,300,300,3). First layer of the convolution layer with 32 filters and kernel size of (5,5). This layer is followed by a Maxpool layer. A convolutional layer with 32 filters and kernel size of (3,3) and followed by a maxpool layer is repeated 2 times before adding another convolutional layer with 64 filters. After adding a flattened layer, 2 dense layers are added with 128 neurons and 4 neurons respectively. Adam optimizers and categorical cross entropy are used for model compilation.

This model is trained for 20 epochs with a batch size of 32. For 20 epochs, the model took roughly 35 minutes to train, after which the model was used for testing on validation sets.

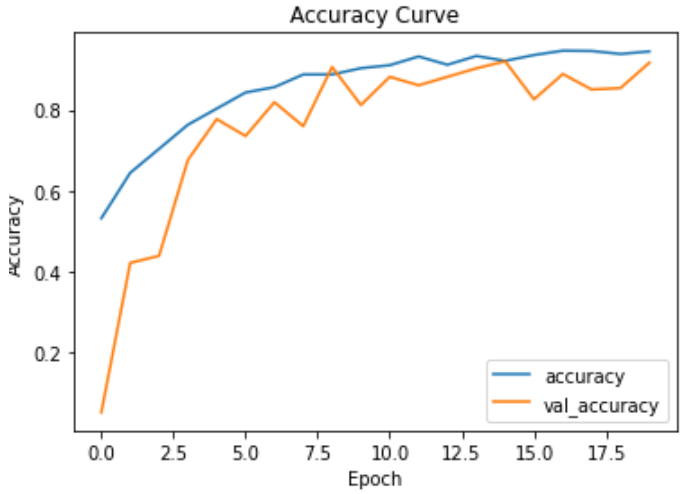
V. Evaluation

The model is trained with 2870 images using CNN. The model is then evaluated with training accuracy and test accuracy. To evaluate our model, we will be viewing the accuracy, recall, and F1-Measure score. The correct classification of brain tumors into 4 different types of tumors is also an evaluation metric.

The validation loss and training loss learning curve are shown below:



The validation accuracy and training accuracy learning curve are shown below.

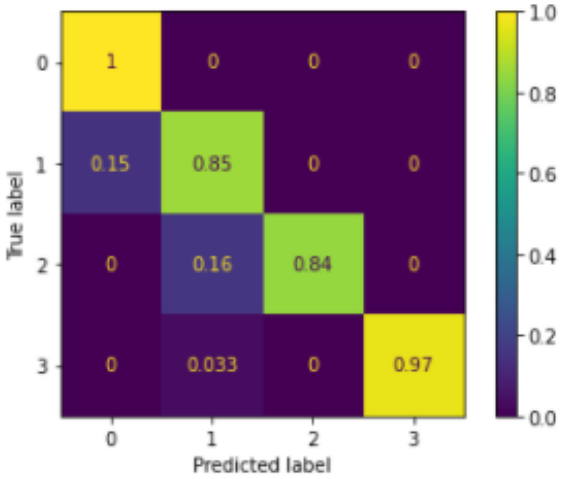


Our model is said to be a "good fit" because the accuracy of the training and validation set increases for every epoch of training. There is no sign of overfitting with the accuracy. The validation loss for both the training and validation set decreases. The overall validation accuracy is 91.68% and the training accuracy is 94.39%.

Other metric values are calculated and tabulated below. This shows the precision, recall, and F1-score of each class.

	precision	recall	f1-score	support
0	0.85	1.00	0.92	69
1	0.88	0.85	0.86	82
2	1.00	0.84	0.91	44
3	1.00	0.97	0.98	92
accuracy			0.92	287
macro avg	0.93	0.92	0.92	287
weighted avg	0.93	0.92	0.92	287

The confusion matrix is plotted for better visualization purposes and is as shown below.



Of all the other classes, the model identified Glioma as most accurate, followed by the pituitary tumor, meningioma, and finally no-tumor.

To compare both the baseline and the proposed model we did a chart:

Comparison	Baseline CNN	Proposed CNN
Input image size	62 x 62	150 x 150
Output Classes	3	4
No. of Convolutional layers	2	4
Dropout Layers	0	1
Final Training Accuracy	98.51 %	94.39 %
Final Training Loss	0.03	0.13
Validation Accuracy	84.19 %	91.68 %
Validation Loss	0.55	0. 38

Two notable differences between the two models are the image resolution and output classes. The proposed model predicts 'no tumor' while the baseline model does not. There are four classes for the proposed model while there are only three classes for the baseline model. In addition, there are 2 additional convolutional layers in the proposed model with a different filter size than that of the baseline model. A dropout layer was also included to reduce overfitting during training.

After training the model, we saw some more differences. The baseline model is better when compared with training loss and accuracy but the proposed model is better for testing the model.

We can conclude that of the two CNN model architectures, the proposed model performed better due to a higher validation accuracy.

VI. Conclusion

In this work, we have used an optimal CNN architecture to classify Meningioma, Pituitary, Glioma, and No Tumor while the baseline only classifies it into the first 3 types. While the training loss and accuracy were slightly higher in the baseline model, when testing the model, the proposed model had better performance. Future scope includes using Faster RCNN for image segmentation to locate the position of the tumor in the images. Originally we planned to include this as part of our work, however, due to our dataset not including bounding boxes (which were necessary for Faster RCNN), we were not able to execute this task.

VII. References

[1] S. Bhuvaji , A. Kadam, P. Bhumkar, and S. Dedge, “Brain tumor classification (mri),” Kaggle, 24-May-2020. [Online]. Available: https://www.kaggle.com/sartajbhuvaji/brain-tumor-classification-mri

[2] RADHAMADHAB DALAI, July 1, 2021, " Brain Tumor Dataset", IEEE Dataport, doi:https://dx.doi.org/10.21227/2qfw-bf10.

[3] S. Ankireddy, “Assistive diagnostic tool for brain tumor detection using computer vision,” arXiv.org, 17-Nov-2020. [Online]. Available:https://arxiv.org/abs/2011.08185

[4] S. Saha, “A comprehensive guide to Convolutional Neural Networks - the eli5 way,” *Medium*, 17-Dec-2018. [Online]. Available: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53.

[5] F. Murat, “The architecture of the 1D-CNN simple ... - researchgate.net,” *ResearchGate*. [Online]. Available: https://researchgate.net/figure/The-architecture-of-the-1D-CNN-simple-model-for-intrusion-detection-The-network-consists_fig1_340697891.

[6] Abiwinanda N., Hanif M., Hesaputra S.T., Handayani A., Mengko T.R. (2019) Brain Tumor Classification Using Convolutional Neural Network. In: Lhotska L., Sukupova L., Lacković I., Ibbott G. (eds) World Congress on Medical Physics and Biomedical Engineering 2018. IFMBE Proceedings, vol 68/1. Springer, Singapore. https://doi.org/10.1007/978-981-10-9035-6_33