Alzheimer's Disease Detection using concepts of Artificial Intelligence

Group: CORE FOUR

Nihar Patel <AU1940119> Tirth Kanani <AU1920144>
Purvam Sheth <AU1940151> Mohit Prajapati <AU1940171>

Mentor: Dr. Mehul S. Raval

***Abstract*— Structural magnetic resonance imaging (sMRI)plays an important role in Alzheimer's disease (AD) detectionas it shows morphological changes caused by brain atrophy. Convolutional neural network (CNN) has been actively used to achieve better performance and accurate diagnosis for AD. Most existing conventional methods utilizes shallow CNN structures, but due to the small amount of sMRI data which limits the ability of CNN to learn high-level features. We buid a CNN network for determining the level of dementia level of Alzheimer's patient from their MRI images. Model achieved a high ROC-AUC score nearly to 99%. Testing Accuracy for identifying different classes is 85%.**

***Index Terms*— Structural magnetic resonance imaging, Alzheimer's disease detection, Convolutional 2D, Seperable Convolutional 2D**

I. INTRODUCTION

Alzheimer's disease (AD) is an irreversible neurodegenerative disease. There are over 50 million people reported around the world suffering from dementia, and this figure is expected to touch 132 million by 2050. Mild cognitive impairment (MCI) is the stage of AD from when initial symptoms are observed, and it is crucial to analyze the pathological changes in this stage. Up-till now, various studies for AD detection have been developed via sMRI which are based on analysis, since it provides an intuitive way to observe the structural changes of the brain.

II. LITERATURE SURVEY

A. Background

Current MRI-based methods can be generally categorized into four classes:

- Voxel based method
- Region-of-interest (ROI) based method
- Whole-image based method
- Patch based method

Voxel-based methods requires cortical thickness, brain tissue density and volume to measure the morphological changes of brain. While, ROI based methods technically needs prior knowledge about the regions associated with disease. These methods naturally lose some useful information. The whole-image based methods use the whole image volume, which makes harder to detect the precise structural abnormality and in patch based method, we take small patches as input, but they are highly relied on the discovery of anatomical landmarks.

Generally, conventional machine learning (ML) methods require complex preprocessing to get features for classification. Whereas, a typical deep learning (DL) based method, convolutional neural network (CNN) can hierarchically extract the most discriminative feature.

CNN has shown great power in end-to-end AD analysis. For instance, 2D CNNs have achieved excellent performance for AD detection.

We will be using some functions and dependencies of tenserflow and keres, some of the important function are as followed:

Dropout layer: Purpose of dropout layer is to minimize the effect of overfitting within a trained network.

Maxpool 2D: Objective maxpool2D is to down-sample an input representation.

Conv 2D: Conv2D are generally smaller than the input image and so we move them across the whole image.

Separable Conv 2D: A separable convolution splits a kernel into 2 separate kernels that do two convolutions: the depth wise convolution and the pointwise convolution.

Batch Normalization: Batch normalization is a technique designed to automatically standardize the inputs to a layer in a deep learning neural network.

Dense Layer: Dense Layer is used to classify image based on output from convolutional layers.

Flattening Layer: Flattening layer used to flatten the output of the convolutional layers to create a single long feature vector.

Adam Optimizer: Adam is an optimization solver for the Neural Network algorithm that is computationally efficient, requires little memory, and is well suited for problems that are large in terms of data or parameters or both. Adam is a popular extension to stochastic gradient descent.

Binary cross-entropy(Loss Function): The binary cross entropy is very convenient to train a model to solve many classification problems at the same time, if each classification can be reduced to a binary choice (i.e. yes or no, A or B, 0 or 1).

III. IMPLEMENTATION

A. Dataset

- The Dataset which we have selected is of "Alzheimer's Dataset (4 class of Images)" from Kaggle, by Sarvesh Dubey. This dataset includes preprocessed MRI images of T1 type.
- Dataset consists of two files - Training and Testing both containing a total of around 5000 images each segregated into the severity of Alzheimer's
  Classes:
  - 1. Non-Demented
  - 2. Very Mild-Demented
  - 3. Mild-Demented
  - 4. Moderate-Demented
- Preprocessed images are received by passing and applying the original MRI images through processes and techniques like Skull-Striping, Segmentation, Normalization and Spatial Smoothing.
  All final processed images are of 176X208 resolution, having 96 dpi and all are 8 bit color images.

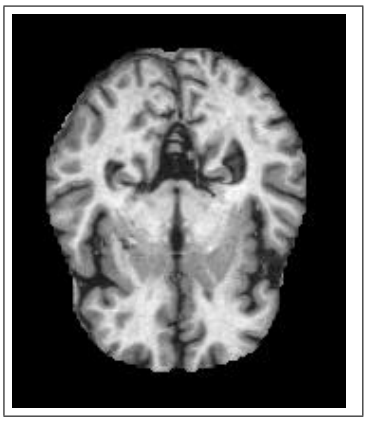

Fig. 1. Data image: Mild-Demented

- Above image is one of the sample images from the Mild-demented Folder for the training of the model.

B. Coding

- Data Loading:
  Function "tf.keras" can easily load in images from a directory. In order for this function to work, the data has to be structured in a file directory format like:
  main_directory/
  class1/
  class1_images
  class2/
  class2_images
  We will be splitting our dataset into a ration of 80:20 for training and validation of our datasets.

- Visualizing the dataset:
  For visualization we will be using dependencies from matplot library as plot. For an example we will show 12 images form our Training Dataset.

- Feature Engineering & Deciding a Matrix:
  Right now we are working with categorical and noncontinuous data, hence, we want to convert our model into one-hot encodings. One-hot encodings are a way for the model to understand that we're looking at categorial instead of continuous data.

- Building CNN model:
  For the making of our model more modular and easier to understand, we will create some blocks. As we are building a convolution neural network(CNN), we will create a convolution block and a dense layer block.
  The most conventional metric is to use probability.
  We have used 5 convolutional blocks which were comprised of convolutional layer, max-pooling and batch-normalization.
  On top of it we used a flatten layer and followed it by four fully connected layers. Also in between we have used dropouts to reduce over-fitting.

- Callback Functions:
  - ModelCheckpoint:
    Training requires a lot of time to achieve a good result, for that often many iterations are required. So, it is better to save a copy of the best performing model only when an epoch that improves at the metrics ends.
  - EarlyStopping:
    During training we can notice that the generalization gap (i.e. the difference between training and validation error) starts to increase, instead of decreasing. This might be a cause of overfitting, which can be solved in many ways (reducing model capacity, increasing training data, data augumentation, regularization, dropout, etc). So, we have applied a simple solution to this problem by stoping the training when the generalization gap keeps getting worse.

Here is a graphical representation of the Callback function using ModelCheckpoint and EarlyStopping.

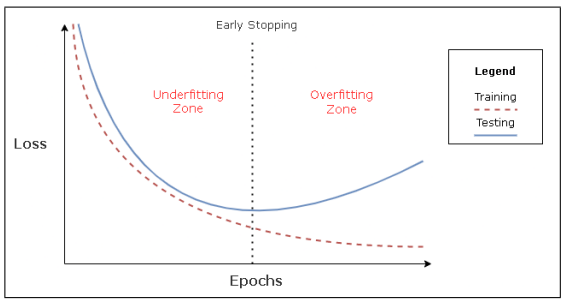

Fig. 2. Early Stopping of the model

- Training the Model:
  Learning Rate:
  To more efficiently train our model. We will be using callbacks to adjust our learning rate and to stop our model once it start converges downwards.
  We have started to train our network from a low learning rate and then further we have increased the learning rate exponentially for every batch.

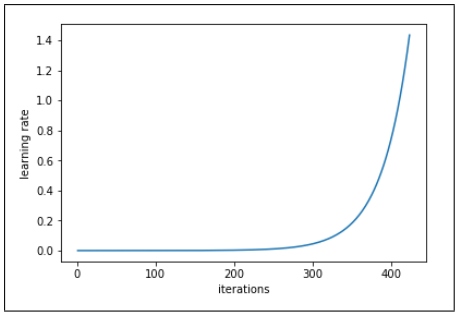

Fig. 3. Learning Rate vs Iteration left

C. Results

- Visualizing model performance:
  For visualizing the performance of the model we will graph the ROC-AUC metric and loss after each epoch for the training and validation data.
  ROC-AUC curve is a performance measurement for the classification problems at various threshold settings.

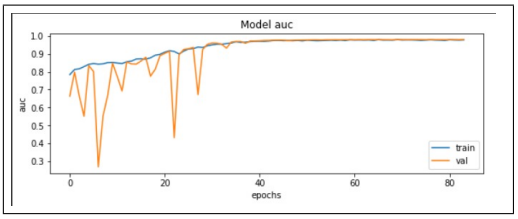

Fig. 4. Accuracy Rate vs Epoch

Above graph represents the Accuracy rate of the model at every Epoch. We have plotted Accuracy Rate at Y-axis, and Epoch at X-axis.

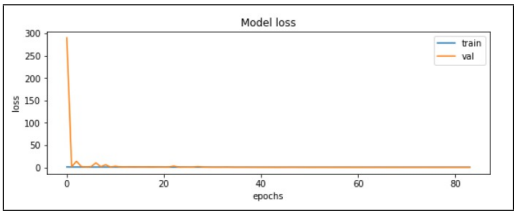

Fig. 5. Loss vs Epoch

Above graph represents the Loss rate of the model at every Epoch. We have plotted Loss Rate at Y-axis, and Epoch at X-axis.

- Predicting and Evaluating results:
  With the help of "model.evaluate()" function we will be able to see the Loss and Accuracy of the model by testing the Test Image.

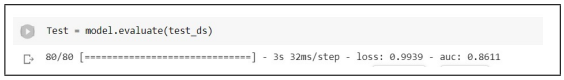

Fig. 6. Evaluation results

D. Conclusion

At the end of our model simulation and after running the evaluation part, we have achieved an testing accuracy for identification for classes of dementia level is 86% and validation accuracy of 99%. We can say that we have achieved a decent amount of accuracy. In future we will try to explore and build a 3D CNN with auto-encoder model.

REFERENCES

[1] Sagar, A. (2019, December 17). Deep Learning for Detecting Pneumonia from X-ray Images. Medium. https://towardsdatascience.com/deep-learning-for-detecting-pneumonia-from-x-ray-images-fc9a3d9fdba8

[2] Surmenok, P. (2021, April 19). Estimating an Optimal Learning Rate For a Deep Neural Network. Medium. https://towardsdatascience.com/estimating-optimal-learning-rate-for-a-deep-neural-network-ce32f2556ce0

[3] Alzheimer's disease - Symptoms and causes. (2021, June 26). Mayo Clinic. https://www.mayoclinic.org/diseases-conditions/alzheimers-disease/symptoms-causes/syc-20350447

[4] Wang, C. (2018, August 14). A Basic Introduction to Separable Convolutions - Towards Data Science. Medium. https://towardsdatascience.com/a-basic-introduction-to-separable-convolutions-b99ec3102728

[5] Alzheimer's Dataset (4 class of Images). (2019, December 26). Kaggle. https://www.kaggle.com/tourist55/alzheimers-dataset-4-class-of-images