# Diagnosis of Early Onset Alzheimer's disease via Deep Learning

Vivek Panchal
School of Computer Science:
University of Adelaide
a1774676@adelaide.edu.au

Teshreque Haq
School of Computer Science:
University of Adelaide
a1836418@adelaide.edu

## ABSTRACT

In order to save human lives in the distant future, there must be a non-invasive and cost-efficient approach to detect early onset Alzheimer's Disease. Using Deep Learning and its associated technologies, this can be detected with high precision and accuracy by analysing common features and patterns observed in MRI images. By utilising Convolutional Neural Networks (CNN) and a dataset of MRI images with different classification of Alzheimer's disease, this paper was successfully able to classify the state of Alzheimer's with high accuracy. This paper aims to further research, analyse and present a plausible solution through individual research. Furthermore, by implementing and utilising resources, the solutions will be compared to further prove validity of the findings.

## COMPUTER SCIENCE CONCEPT

**• Computer Aided Diagnosis → Detection of early onset Alzheimer's**

## 1 INTRODUCTION

The brain is the most complex part of a human body. It is the seat of intelligence, interpreter of senses, initiator of bodily movement and the controller of human behaviour [2]. With it being such a vital organ, a neuro-degenerative disorder like Alzheimer Disease (AD) can be very fatal for an individual. The progression of this disease can be categorised into four ascending stages. The first stage is known as Mild Cognitive Impairment (MCI) where there is some memory impairment and retains most of the other cognitive functional activities. The second and third stages are known as Mild AD and Moderate AD, respectively, where there is further cognitive deficit with basic functionalities like memory. The last stage is termed as Severe AD where the patient cannot be cured and death is likely to occur. Therefore, it is imperative that AD is detected before it transitions to a much more deadly phase where it cannot be cured [1].

Machine learning is sub-category of artificial intelligence that focuses on the use of data and algorithms to emulate the way humans learn, gradually improving its accuracy through well crafted algorithms. An algorithm like Convolutional Neural Network can be used to classify and detect early onset AD using magnetic resonance imaging (MRI) datasets. MRI is a non-invasive procedure that uses magnetic fields and radio waves to take detailed cross-sectional pictures of the body. MRI utilizes the fact that protons have angular momentum which is polarized in a magnetic field. This means that a pulse of radiofrequency can alter the energy state of protons and, when the pulse is turned off, the protons will, on returning to their energy stage, emit a radiofrequency signal [3]. A number of MR sequences that are sensitive to microstructural change (e.g., magnetization transfer or diffusion) have shown alterations in AD. These features can be detected using CNN and can classify normal patients and detect patients with AD through a vigorous process of cleaning and pre-processing EEG data, extracting necessary features, model selection and validation and evaluation of the findings.

### 1.2 Motivation

Alzheimer Disease (AD) is a chronic Neuro-degenerative disorder that has ranked as third most expensive disease and sixth leading cause of death in the United States [4]. There is no cure for AD once it is in its severe stage. The

use of MRI images as a diagnostic tool still proves to be a challenging part even in the current era of technology. None of the existing systems produced from previous studies have proven to be clinically or analytically validated for real-life medical purposes. Hence, objectively the system requires significant growth and improvement in order for it to be of any use in the real world. The motivation of this project is nothing but to explore as much as possible and come up with optimised CNN to detect early onset AD. With the numbers of skyrocketing and no specific cure, it is vital there is more research and focus put onto this to save lives. Although there is a very clear benefit to detect AD through the use of ML. There are several methods to detect AD, although the least invasive is through the analysis of EEG data. Consequently, this means the cost is significantly lower, making it the most viable method. Refer to table 1 below to see the comparison of the most commonly used methods [5].

**Table 1: Comparison of the methods used for diagnosis of Alzheimer disease**

| Name of the method used | Diagnosis | Progression | Non-invasiveness | Cost |
|---|---|---|---|---|
| PET | Yes | No | Yes | High |
| MRI | Yes | Yes | Yes | High |
| Biomarkers | Yes | Yes/no | No | High |
| EEG | Yes | Yes | Yes | Low |

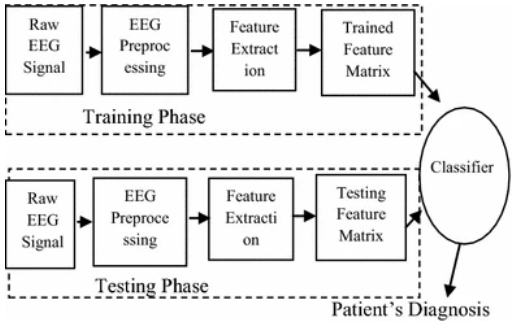
## 2.1 General Literature Review

A peer reviewed literature by Bairagi (2018) presented analysis of EEG signals in time and frequency domain for detection of Alzheimer disease in early stage using Spectral and Complexity based features of EEG by use of suitable machine learning algorithm. The study identified three main characteristics of EEG signals for Alzheimer diagnosis. These are:

- The slowing effect – EEG signals of Alzheimer patients is associated with the increase of relative power (RP) of the low frequency bands of delta and theta.
- Reduced complexity – This is measured by the use of different non-linear features such as Information Theory and it shows the increase in regularity of EEG signals in Alzheimer patients.
- Loss of synchrony measures – The Synchrony measures obtained can be impacted significantly by brain events and choices. Therefore, several Synchrony measures can be included such as the Pearson Correlation Coefficient to quantify the relationships between two or more signals.

The diagnosis can be done through feature extraction of spectral features and wavelet features. Figure 1 below shows the diagnosis system for this study.

**Figure 1: Alzheimer Disease diagnosis system**

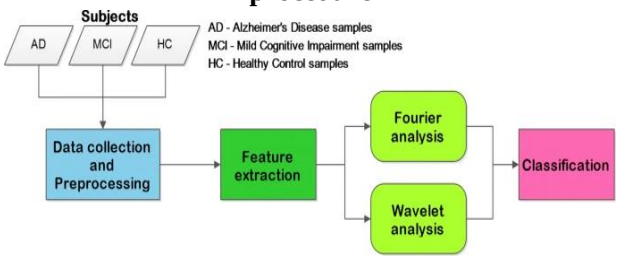



After Bairago (2018) computed the spectral and wavelet features, the results that he discovered were quite impressive to say the least. The computed values showed that the power is decreased in case of Alpha and Beta bands if the patient has Alzheimer's, which shows that there is less complexity in EEG signal. These results obtained in this literature were further verified using a database where similar results were available for comparison. The computation of mean and variance of wavelet-based coefficients calculated for both Normal and Alzheimer's disease patients were compared. It was concluded that the values of means and variance in Case of Normal patients is high, suggesting that EEG of Normal subjects is more complex than patients of Alzheimer's. This could be explained by the growth of Neurons residing in the brain regions. Overall, the study concluded that when combination of spectral and wavelet features is done together, classification rate as well as diagnostic accuracy obtained is more and it provides comparatively better results than individuals in terms of accuracy (up to 94%) [5].

In contrast, Fiscon et al (2018) took a slightly approach with their study. They have applied a procedure that exploits feature extraction and classification techniques to EEG signals, whose aim is to distinguish patient affected by Alzheimer's Disease (AD) from the ones affected by Mild Cognitive Impairment (MCI) and healthy control (HC) samples. They specifically perform analysis on time-frequency by applying both Fourier and Wavelet

Transforms. In simpler terms, these both are a form of mathematical transformation of functions depending on their spatial frequency or temporal frequency. The classification procedure of this study is as follows: (i) preprocessing of EEG signals; (ii) feature extraction by means of the Discrete Fourier and Wavelet Transforms; and (iii) classification with tree-based supervised methods. This can be visualized by figure 2 below:

**Figure 2: Flowchart of the EEG signal analysis procedure**

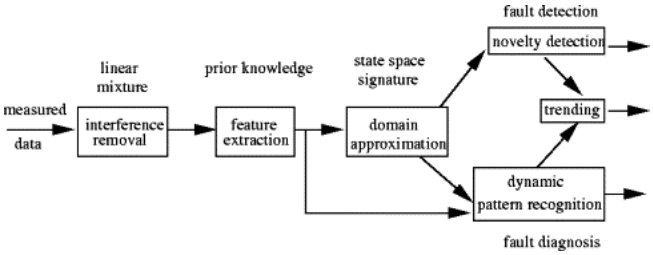



Of the two methods of analysis, Fourier and Wavelet, Wavelet feature extractions outperformed Fourier in terms of accuracy. Wavelet achieved 83%, 92%, and 79% of accuracy when dealing with HC vs AD, HC vs MCI, and MCI vs AD classification problems, respectively. This was constantly higher than Fourier by approximately 16% on average. In conclusion, although this study found Wavelet feature extraction was better than Fourier, it was performed on a relatively small sample size of 109 subjects. Therefore, it is not enough to conclude if one is truly better than the other, hence, it is suggested to use a combination of both for automatic patients classification based on their EEG signal for aiding the medical diagnosis of Alzheimer's [6].

Significant research has been done regarding automatic EEG classification and most recently, a lot of research has been done on artefact-removal from EEG data using independent component analyses (ICA). This was the exact focus of a study by Melissant et al (2005). Their study described a method for detection of EEG patterns indicative of Alzheimer's using automatic pattern recognition techniques. The main focus was on artefact removal stage based on ICA prior to automatic classification stage. ICA assumes that the electric dipoles in the cortex are modelled as independent. Artefacts are detected as independent sources based on their spectral properties. This makes that ICA can be used as a general method for artefact removal, whereas other known methods are in most cases only usable for a subset of artefacts. The study had many complex stages involving artefact removal using ICA, feature

extraction and classification. This can be seen below in Figure 3:

**Figure 3: Stages in automatic classification of Alzheimer's Disease using ICA enhanced EEG measurements**



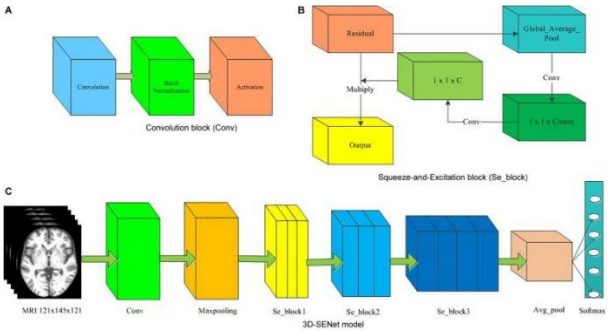

The study concluded that their method for automatic detection of Alzheimer's disease allows for detection accuracies that are comparable to the best results reported in the literature. Artefact removal with ICA prior to feature extraction and classifier training generally leads to improved detection rates, especially in the detection of early Alzheimer's disease [7].

## 2.1 Specific Literature Review

Now focusing on more specific literature reviews to narrow down on the actual topic of the project, which is to detect early onset AD using MRI images. By gathering more general knowledge from credible sources above and narrowing the focus down, it will allow for more relevant knowledge and techniques to be understood and possible be implemented in this project.

A study by Pan et al. (2020) focused on a very similar topic, with the aim to detect early onset AD through MRI images. In this study, the CNN was utilized mainly to recognize 2D images with displacement, scaling, and other non-deformed distortions. Data were reconstructed, so that an image was inputted into the CNN model as a vector for easy feature extraction and classification. The figure below shows the 3D-SENet CNN Model they used for this study.

**Figure 4: The architecture of the 3D-SENet CNN Model**




In order to overcome the possible over-fitting of the CNN model, and to add some image discrepancy, there were six methods used to generate augmented images. These include, rotation, translation, gamma correction, random noise addition, scaling, and random affine transformation. The classification performance was good for their scope of the study, which had an average classification accuracy of 84% for AD patients compared to healthy individuals. The other averages were slightly below this, although they were still pretty good but not to the level, they were expecting it to be (Pan et al., 2020).

## 3 Methodology

The methodologies used in this research project to detect and classify the level of Alzheimer's is broken down into sub-sections to explain their respective involvement and usefulness within this project.

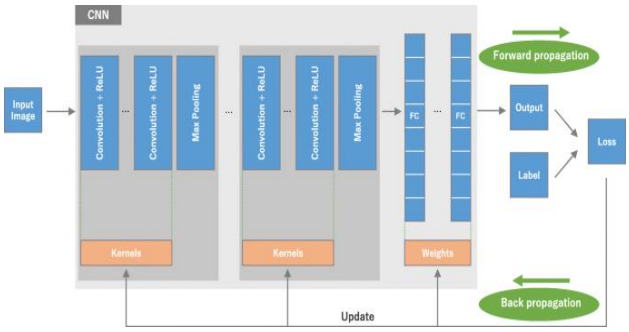
### 3.1 Deep Learning

Deep learning is a subset of machine learning, which is essentially a neural network with three or more layers. Convolutional Neural Networks (CNN) is an algorithm which takes images as input, assigns importance by using learnable weights and biases to various objects in the images allowing it to differentiate from one another. The architecture of CNN was inspired by the connectivity pattern of neurons in the human brain, especially the organization of the visual cortex. Familiarity with this state-of-the-art methodology would help not only researchers who apply CNN to their tasks in radiology and medical imaging, but also clinical radiologists, as deep learning may influence their practice in the near future (IBM Cloud Education, 2020).

### 3.2 CNN overview

CNN is a mathematical construct that is typically composed of three types of layers: convolution, pooling, and fully connected layers. The first two, convolution and pooling layers, perform feature extraction, whereas the third, a fully connected layer, maps the extracted features into final output, such as classification. A model's performance under particular kernels and weights is calculated with a loss function through forward propagation on a training dataset, and learnable parameters, i.e., kernels and weights, are updated according to the loss value through backpropagation with gradient descent optimization algorithm. ReLU, rectified linear unit. Refer to the figure below for an overview of the process (Yamashita et al., 2018).

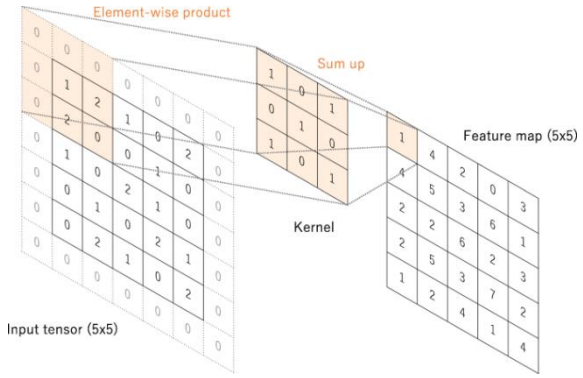
**Figure 5: An overview of a convolutional neural network (CNN) architecture**




Convolution:

A convolution layer plays a key role in CNN, which is composed of a stack of mathematical operations, such as convolution, a specialized type of linear operation. It's used for feature extraction, where an array of numbers, also known as a kernel is applied across a given input, which also is an array of numbers called a tensor. The product between each element of the kernel and the input tensor is calculated at each location of the tensor. It is then summed to obtain the output value in the respective position of the output tensor, which is called a feature map. Refer to the figure below for a visualisation of this process (Yamashita et al., 2018).

**Figure 6: Demonstration of convolution operation**



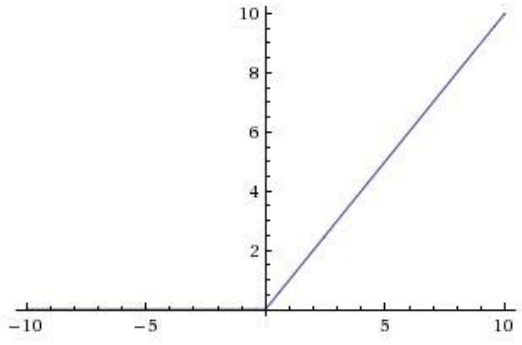

Nonlinear activation function:

The outputs of a linear operation such as convolution are then passed through a nonlinear activation function. The activation function used for all layers except the last layer in this study is ReLU (Yamashita et al., 2018). The function returns 0 if it receives any negative input, but for any positive value x it returns that value back. So, it can be written as:

$$f(x) = max(0, x).$$

Graphical representation of this function can be seen in the figure below:

**Figure: Graphical representation of ReLU activation function**



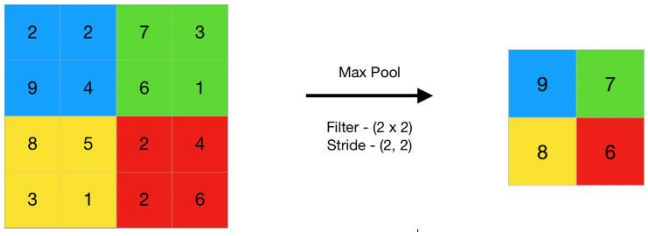

Last layer activation function:

The activation function applied to the last fully connected layer is usually different from the others. An appropriate activation function needs to be selected according to each task. An activation function applied to the multiclass classification task is a softmax function which normalizes output real values from the last fully connected layer to target class probabilities, where each value ranges between 0 and 1 and all values sum to 1. This was the last layer activation function used in this project as well (Saha, 2018).

Pooling layer:

After the convolutional layers as seen from the above structure, a pooling layer is added. The purpose of a pooling layer is to down-sample the output of the convolutional layers by summarizing the features present in a feature map. There are two different types of pooling layers, and both are utilized in this project. Firstly, max pooling selects the maximum element from the region of the feature map covered by the filter. Thus, the output after max-pooling layer would be a feature map containing the most prominent features of the previous feature map. Refer to the figure below for an example of max pooling. Finally, average pooling, as the name suggests, computes the average of the elements present in the region of the feature map covered by the filter. Therefore, it gives the average of the features present inside of the filter size (Saha, 2018).

**Figure 7: An example of max pooling operation with a filter size of 2 × 2**

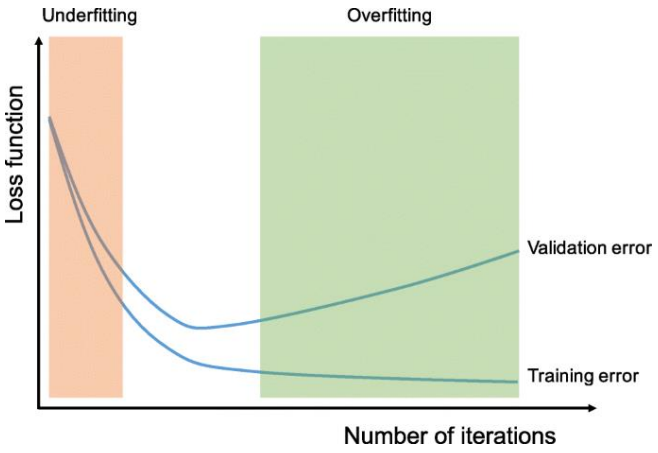



Overfitting:

Overfitting refers to a situation where a model learns statistical regularities specific to the training set, i.e., ends up memorizing the irrelevant noise instead of learning the signal, and, therefore, performs less well on a subsequent new dataset. This is one of the main challenges in machine learning, as an overfitted model is

not generalizable to never-seen-before data. There are two ways to mitigate the effect of overfitting, firstly to have more training data, however, in this case we are limited to a specific dataset. Hence, the second solution was to introduce dropout layers and was used in my method for this project. Dropout is a recently introduced regularization technique where randomly selected activations are set to 0 during the training, so that the model becomes less sensitive to specific weights in the network. A routine check for recognizing overfitting is to monitor the loss on the training and validation sets during the training iteration. If the model performs well on the training set compared to the validation set, then the model has been overfit to the training data. And if the vice versa is true, then the model is underfitted (Saha, 2018).

**Figure 8: Shows the correlation of overfitting and underfitting with training and validation error**



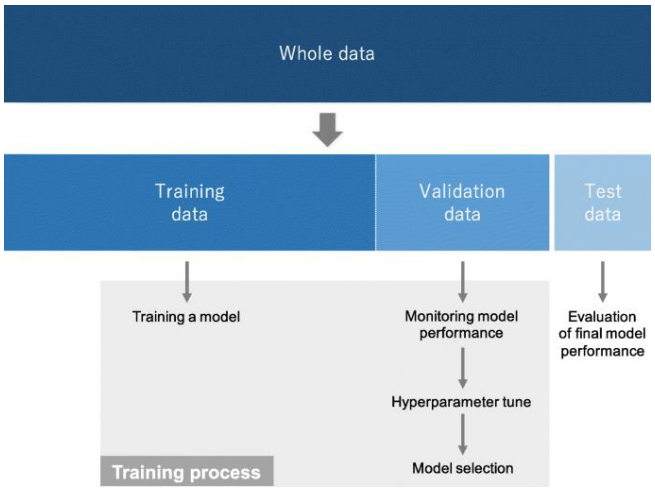

Transfer Learning:

Since the dataset that will be utilized for this project is relatively small, transfer learning is the most effective strategy that can be implemented to train it. Transfer Learning is a strategy to train a network on a small dataset, where a network is pretrained on an extremely large dataset, such as ImageNet. The underlying assumption of transfer learning is that generic features learned on a large enough dataset can be shared among seemingly disparate datasets. This portability of learned generic features is a unique advantage of deep learning that makes itself useful in various domain tasks with small datasets (Saha, 2018).

Dataset:

A dataset is typically split into three sets: a training, a validation, and a test set. A training set is used to train a network, where loss values are calculated via forward propagation and learnable parameters are updated via backpropagation. A validation set is used to monitor the model performance during the training process, fine-tune hyperparameters, and perform model selection. A test set is ideally used only once at the very end of the project in order to evaluate the performance of the final model that is fine-tuned and selected on the training process with training and validation sets. A similar method is also implemented with the dataset used for this project. Refer to the figure below for a visualization of this process. The dataset used in this study has four classifications, Mild demented, moderate demented, non-demented and very mild demented.

**Figure 9: A dataset being split into three categories**




## 4 Experimental Setup

All the code produced for the purpose of this project is done through with the help of online research, only tutorials, problem solving and helpful code snippets. These are all referenced in the reference list below.

The following exact setup was used to create, produce and test all of the code in this research project. All of this

was performed a Windows 10 Home 64-bit machine with:

- 12th gen Intel i7-12700k
- NVIDIA GeForce RTX 3080
- 32GB of RAM.

These specs are important to mention, especially for deep learning as it relies highly on the use of the graphics card in correlation with the CPU to train a dataset.

Firstly, python needs to be installed into the machine through the installer provided on its official website Visual Studio Code for windows was downloaded as the main and only IDE to write, test and compile python code to keep consistent results through the project without any random or systematic errors. Visual Studio IDE was setup python interactive window with Jupyter Code Cells.

Next, Anaconda Distribution software was downloaded, and along with it, a command prompt called Anaconda Prompt. This Prompt was used to create a new python 3.9 environment to be used for this project.

In preparation to train the dataset and utilize the GPU:

- CUDA Toolkit version 11.2
- cuDNN version 8.1

need to be installed through the Anaconda Prompt as these are the versions compatible with TensorFlow. CUDA is an application programming interface that allows software to use graphics processing units. cuDNN is Deep Neural Network library (cuDNN) is a GPU-accelerated library of primitives for deep neural networks.

Furthermore, in order to reproduce the code and analysis in this project, the following python libraries were installed through the Anaconda Prompt in the newly created separate environment:

- TensorFlow-gpu
- Keras
- Matplotlib
- Split-folders**.**

Specifically, the 'gpu' version of TensorFlow was installed allowing the inference to run on the GPU instead of CPU. Since the GPU is far superior, especially in this case compared to the CPU, it will train the dataset much faster and efficiently.

In total, there were three CNN models being tested in this project. Two CNN models were developed to produce the highest accuracy as possible as this is the ideal goal of the project. It was believed that keeping some factors constant to test one or two parameter tuning isn't ideal for CNN networks because there are so many factors that affect it and they all have to be synchronized for the best accuracy. Therefore, both of these two models do not contain any constant factors rather, they're focused on producing the best possible outcome, which is high classification accuracy. This is possible by changing the four main factors, tune parameters, image data augmentation, deeper network topology and handling overfitting and underfitting. And the third CNN model is using Transfer Learning with the pre-trained model called EfficientNetB0.

<u>Common between all three CNN models:</u>

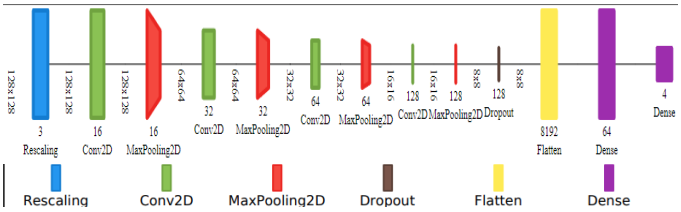
Firstly, all the necessary python libraries mentioned earlier were imported. Splitfolders's 'ratio' function was utilized to divide the dataset into 80% train, 10% validation and 10% test. Then the output folder of this was loaded into the dataset format for Keras as a preprocessing layer by using the 'image_dataset_from_directory' function. The image size was set to 128x128 with the batch size being 64. After their individual CNN model was constructed, Early Stopping and Model Checkpoint were set. Early Stopping's patience was set to 6 and min_delta to 0.01. The Model Checkpoint was saving only the best accuracy by using the 'save_best_only' and setting it to true. Then the model was compiled with loss being "sparse_categorical_crossentropy' and using 'Adam' as the optimizer. The model was trained with the 'model.fit' function setting the epochs to 30. For analytics, evaluate function was used to find out the accuracy of the model

on the test dataset. Furthermore, matplotlib's plot and title functions were used to generate necessary graphs. For example, accuracy of training data vs accuracy of validation data. Finally, matplotlib's figure function was utilized to create a figure size of 20 x 20. Inside the figure was predict function to predict and classify what kind of Alzheimer's it is based on the probabilities provided by softmax. This figure would display a 4 by 4 plot of MRI images, stating the actual classification on top of the image and the predicted classification towards the left with it being colored in green if it was correct and red if it was incorrect.

First CNN Model:

This model firstly uses 'keras.layers.experimental.preprocessing.Rescaling' function to rescale the input values to a range of 1/255. It is a four-layer Conv2D layer, with each layer doubling the size of filter from the previous one starting from 16 filters on the first layer till 128 filters on the last layer. After each Conv2D layer, there is a max pooling layer of size 2 by 2. Each Conv2D layer is using kernel size of 3 by 3, activation function ReLU and padding as 'same'. It then has a dropout layer of 25%. Furthermore, the layer is flattened and then a dense layer of 64 units and the activation function ReLU is applied. And lastly, the last dense layer of 4 units has the activation function 'softmax'. This model can be seen in the figure below:

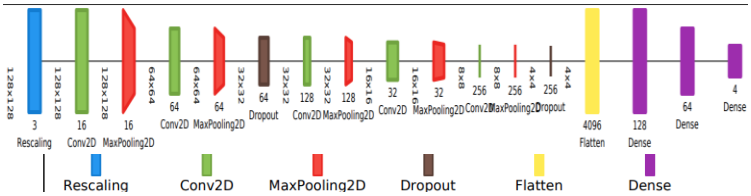
**Figure 10: Visual representation of the first CNN Model**




Second CNN Model:

This second model is slightly different than the first one with deeper network topology and slightly different use of parameters. It firstly uses 'keras. layers.experimental.preprocessing. Rescaling' function to rescale the input values to a range of 1/255. It is a five-layer Conv2D layer, with the following number of filters respectively in order, 16, 64, 128, 32, 256. After each Conv2D layer, there is a max pooling layer of size 2 by 2. Each Conv2D layer is using kernel size of 3 by 3, activation function ReLU and padding as 'same'. Each layer is also using a kernel initializer called 'he_normal'. It has a dropout layer of 20% between the second and third layer in the CNN network. It then also has a dropout layer of 25% after the network is finished. Furthermore, the layer is flattened and then a dense layer of 128 and then 64 units and the activation function ReLU is applied. And lastly, the last dense layer of 4 units has the activation function 'softmax'. It used the optimizer 'rmsprop' to compile the model. This model can be seen in the figure below:

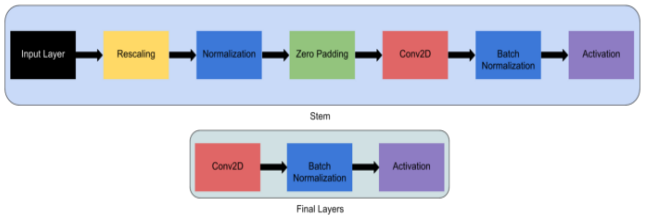
**Figure 11: Visual representation of the second CNN Model**




Third CNN Model (EfficientNetB0):

EfficientNetB0 is a convolutional neural network that is trained on more than a million images from the ImageNet database. The network can classify images into 1000 object categories, such as keyboard, mouse, pencil, and many animals. As a result, the network has learned rich feature representations for a wide range of images. The model contains 237 layers in total and it is made up of the same layers with different parameters. A basic demonstration of this model can be seen below:

**Figure 12: Visual representation of the EfficientNetB0 Model**




## 5 Results

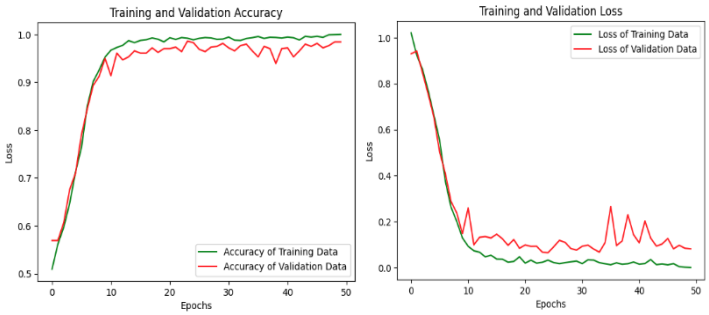
First CNN Model Result:

The accuracy of this model on the test data set after running it for 50 epochs came down to 99.53%.

**Graph 1: Training data accuracy and loss of the first CNN Model**


From graph 1 above, it is seen that as the accuracy of training data increases, the loss of training data decreases at approximately the same rate. Suggesting that the model is working appropriately.

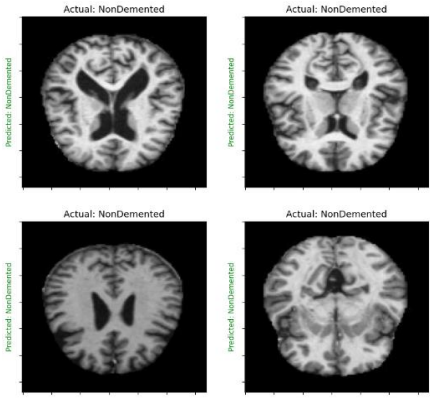
**Graph 2: Training and Validation accuracy (left) Training and validation loss (right)**




As seen from graph 2 above, the training and validation loss (graph on the right) is decreasing until about 10 epochs, and then from there it is slowly plateauing down. Although there are some visible spikes in the loss of validation data, towards the end, it may suggest the model is slightly overfitting. A similar pattern is also seen in the training and validation accuracy graph on the left.

From the prediction figure below, a randomly selected four MRI images from the test set were generated. The actual classification is seen on top of each image and the predicted is seen towards the left of the image. If it is coloured in green, the prediction was accurate, otherwise it is red. It is seen all of the four predictions were correct in the figure below.

**Figure 13: Predictions of the first CNN Model**



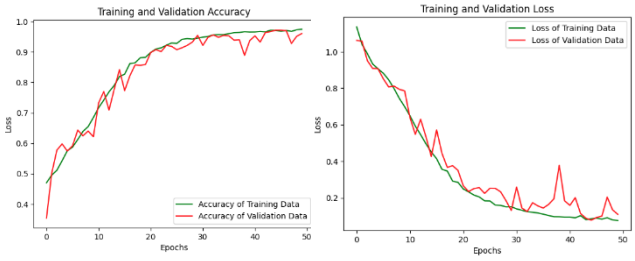

Second CNN Model Result:

The accuracy of this model on the test data set after running it for 50 epochs came down to 96.57%.

**Graph 3: Training data accuracy and loss of the second CNN Model**

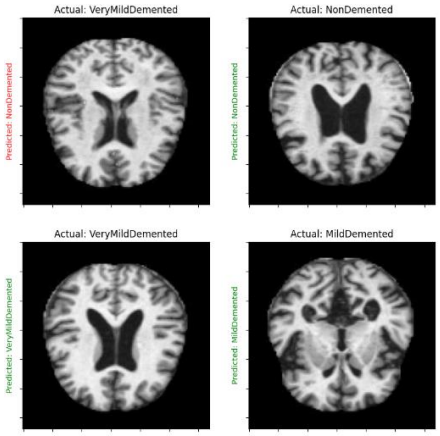

From graph 3 above, it is seen that as the accuracy of training data increases, the loss of training data decreases at approximately the same rate. Suggesting that the model is working appropriately.

**Graph 4: Training and Validation accuracy (left) Training and validation loss (right)**



As seen from graph 2 above, the training and validation loss (graph on the right) is steadily decreasing until about 35 epochs, and then from there it is slowly plateauing down. Although there are some visible spikes in the loss of validation data, towards the end, it may suggest the model is slightly overfitting. A similar pattern is also seen in the training and validation accuracy graph on the left.

**Figure 14: Predictions of the second CNN Model**

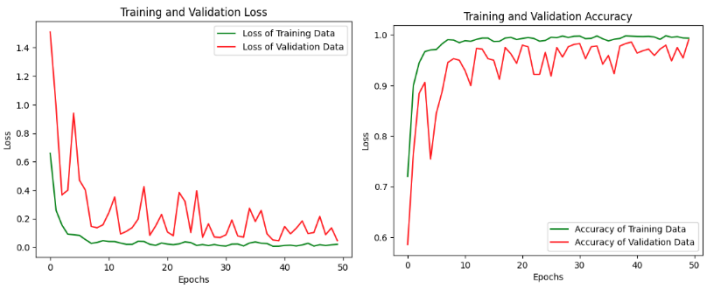



From the predictions figure above, it is seen that 3 out of 4 images were predicted correctly. Although, since the accuracy is 96.57%, you can expect for most of the predictions to be accurate.

EfficientNetB0 CNN Model Result:

The accuracy of this model on the test data set after running it for 50 epochs came down to 97.66%.

**Graph 5: Training and Validation accuracy (left) Training and validation loss (right**




As seem from graph 5 above, the validation loss and accuracy aren't stable relative to the training loss and accuracy, suggesting that the model is overfitting.

Table 1: Summary of all three models

| Model | Accuracy | Optimiser used | Number of layers |
|---|---|---|---|
| First | 99.53% | Adam | 4 |
| Second | 96.57% | rmsprop | 5 |
| EfficientNetB0 | 97.66% | Adam | 237 |

From the summary table above, the results for this project were quite impressive. The first two models produced an exceptional accuracy, although it also showed that a deeper and bigger CNN network does not equate to a higher accuracy. The second CNN model was bigger than the first although it produced a lower accuracy. It is also to note that the optimizer used in the second model was different to the first, which may have had an impact on the accuracy. This is further proved by using a pre-trained 237-layer model which produced a lower accuracy compared to the simplest network which was the first model.

## 6 Conclusion

In conclusion, the results were quite impressive based on the accuracy produced. Especially the first model was the most impressive as it was the simplest network and produced the highest accuracy. That suggests that there is more to CNN networks than what we know. Having a bigger network like EfficientNetB0 with 237 layers didn't necessarily yield the highest accuracy. Therefore, with each project, there needs to be a customized model optimised for that specific project in order to produce the highest accuracy. For future work, a larger training dataset could be utilized alongside more parameter tuning to take this to the next level.

## REFERENCES:

[1] Alzheimer's Disease and Dementia. 2022. Stages of Alzheimer's. [online] Available at: <https://www.alz.org/alzheimers-dementia/stages> [Accessed 9 August 2022].

[2] Hopkinsmedicine.org. 2022. Brain Anatomy and How the Brain Works. [online] Available at: <https://www.hopkinsmedicine.org/health/conditions-and-diseases/anatomy-of-the-brain#:~:text=What%20is%20the%20brain%3F,central%20nervous%20system%2C%20or%20CNS.> [Accessed 16 August 2022].

[3] Houmani, N., Vialatte, F., Gallego-Jutglà, E., Dreyfus, G., Nguyen-Michel, V., Mariani, J. and Kinugawa, K., 2018. Diagnosis of Alzheimer's disease with Electroencephalography in a differential framework. PLOS ONE, 13(3), p.e0193607.

[4 ]Alzheimer's & Dementia, 2021. 2021 Alzheimer's disease facts and figures. 17(3), pp.327-406.

*[5] Bairagi, V., 2018. EEG signal analysis for early diagnosis of Alzheimer disease using spectral and wavelet based features. International Journal of Information Technology, 10(3), pp.403-412.*

[6] Fiscon, G., Weitschek, E., Cialini, A., Felici, G., Bertolazzi, P., De Salvo, S., Bramanti, A., Bramanti, P. and De Cola, M., 2018. Combining EEG signal processing with supervised methods for Alzheimer's patients classification. BMC Medical Informatics and Decision Making, 18(1).

[7] Melissant, C., Ypma, A., Frietman, E. and Stam, C., 2005. A method for detection of Alzheimer's disease using ICA-enhanced EEG measurements. Artificial Intelligence in Medicine, 33(3), pp.209-222.

[8] Fiscon, G., Weitschek, E., Cialini, A., Felici, G., Bertolazzi, P., De Salvo, S., Bramanti, A., Bramanti, P. and De Cola, M., 2018. Combining EEG signal processing with supervised methods for Alzheimer's patients classification. BMC Medical Informatics and Decision Making, 18(1).

[9] Fiscon, G., Weitschek, E., Cialini, A., Felici, G., Bertolazzi, P., De Salvo, S., Bramanti, A., Bramanti, P. and De Cola, M., 2018. Combining EEG signal processing with supervised methods for Alzheimer's patients classification. BMC Medical Informatics and Decision Making, 18(1).

[10] By: IBM Cloud Education (no date) What is deep learning?, IBM. Available at: https://www.ibm.com/cloud/learn/deep-learning (Accessed: October 31, 2022).

[11] Pan, D. et al. (2020) "Early detection of alzheimer's disease using Magnetic Resonance Imaging: A novel approach combining convolutional neural networks and ensemble learning," Frontiers in Neuroscience, 14. Available at: https://doi.org/10.3389/fnins.2020.00259.

[12] Saha, S. (2018) A comprehensive guide to convolutional neural networks — the eli5 way ... Available at: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53 (Accessed: October 29, 2022).

[13] Yamashita, R. et al. (2018) "Convolutional Neural Networks: An overview and application in Radiology," Insights into Imaging, 9(4), pp. 611–629. Available at: https://doi.org/10.1007/s13244-018-0639-9.

## GitHub repository:

https://github.com/Vivek6996/adv.git

## ACKNOWLEDGMENTS

This work was supported and monitored under the supervision of Tashreque Haq.