Performance Evaluation of Different Optimizers on Alzheimer's Disease Classification Using a Customized Convolutional Neural Network

Pallavi Saikia* and Sanjib Kr Kalita

Department of Computer Science, GauhatiUniversity,Guwahati, India
Email: saikia.pallavi@gmail.com (P.S.); sanjib959@gauhati.ac.in (S.K.K.)
*Corresponding Author

Abstract—**Alzheimer's disease is a type of dementia that usually affects elderly people. It is a neurological disorder that causes a patient to lose memory gradually over time. The brain of an Alzheimer Disease (AD) patient shrinks due to the accumulation of amyloid plaques in the neurons. As a result, the neurons, which are the basic building blocks of the brain, lose connections and cannot communicate with each other. A person can be prevented from having AD if diagnosed at the right time. So, it's very important to detect patients with mild symptoms of dementia to save them from getting AD. In this work, we have proposed a customized Convolutional Neural Network (CNN) model for classifying Alzheimer's disease. The model has been evaluated with two benchmark datasets, the Kaggle Alzheimer's dataset and the ADNI dataset. The two datasets differ in the number of images. The K-fold technique has been applied to overcome the problem of class imbalance. We have updated the model parameters using optimizers, namely Stochastic Gradient Descent (SGD), SGD with momentum, AdaGrad, AdaDelta, RMSprop, and Adam. Experimental results established that the proposed model outperforms many of the state-of-the-art models, considering the two benchmark datasets. In case of the Kaggle dataset, we have attained 99%accuracy using a customized CNN, outperforming other previous works that used a pre-trained model but still failed to produce 99% accuracy. Considering the number of images and class imbalance ADNI data*set* also outperformed other previous models by achieving 90% accuracy. The main advantage of this work is that it studies the impact of all the state-of-the-art optimizers with different epochs rather than experimenting with a particular optimizer and epoch. Optimizers have a huge impact on the performance of the model and also on the convergence time. It is an important hyperparameter that needs to be analysed further for better classification purposes.**

Keywords—**Alzheimer disease, CNN, class imbalance, K fold, optimizer**

I. Introduction

Dementia is an umbrella term that is used to describe the abnormal changes in the brain that lead to impairment of memory and cognitive function. Alzheimer's Disease (AD) is one of the types of dementia. It is a neurological and irreversible brain disorder that usually occurs in elderly people and causes a gradual loss of memory. Initially, the affected person finds difficulties recalling events, and in due course of time, it gradually increases and makes him forget even his name. The brain cells called neurons in a person with AD get damaged due to the accumulation of the abnormal protein's beta-amyloid and phosphorylated tau [1]. As the brain cells get damaged, they cannot communicate with each other, causing the brain's inability to plan, recall and concentrate. This disease mainly affects the hippocampus area which is an integral part of the brain and is associated with memory, learning and cognitive skills. AD can be detected by performing a Computed Tomography (CT) scan, Positron Emission Tomography (PET) scan, or Magnetic Resonance Imaging (MRI), which provides the anatomical details of the brain. This mental disorder is increasing day by day, and till now there has been no cure for AD, but early detection of AD might be of great help to prevent the person from going to a difficult stage. A study using the latest data from the 2023 population projections from the U.S. Census Bureau and the Chicago Health and Ageing Project (CHAP), a population-based study of chronic health conditions of older people, shows that an estimated 6.7 million Americans age 65 and older will be living with Alzheimer's dementia in 2023 [2, 3]. In India, too, a lot of people are being affected by Alzheimer's disease, and day by day the numbers are increasing. According to the Dementia India Report prepared for the Alzheimer's and Related Disorders Society of India (ARDSI) [4], the number of people with dementia in younger age groups, 60–75 years, is expected to increase steadily over time. It's very important for the medical professionals as well as the researchers to detect the disease at the right time for proper treatment and also for further analysis to predict the stage for the patients. AD has the following three stages: Alzheimer's Disease (AD), Cognitive Normal (CN), and Mild Cognitive Impairment (MCI). MCI is the initial stage

Manuscript received February 19, 2025, revised April 7, 2025; accepted June5, 2025; published July 17, 2025.

that might lead to AD in the future. The challenge is to detect whether a person with MCI can be prevented from going into the AD stage through early diagnosis. Researchers and medical professionals are working together to find solutions for AD. Classification of AD into those stages is a very challenging task, and it helps medical professionals gain insight into whether a person who is having mild symptoms of forgetfulness might develop AD in the future. Research in medical image analysis and classification has been gaining ground day by day with new inventions and methods. Earlier machine learning techniques, such as SVM, Random Forest, K-Nearest Neighbour, etc. were used in this AD classification. These methods have some drawbacks, such as being unable to handle huge amounts of data and feature extraction and selection. With the advent of GPUs and deep learning techniques, these obstacles have been overcome to a certain extent and are able to give promising results. The key medical imaging modalities are Computed Tomography (CT), Magnetic Resonance Imaging (MRI), Positron Emission Tomography (PET), Ultrasound, X-ray, and hybrid modalities. These modalities play a vital role in the detection of anatomical and functional information about the brain for diagnosis as well as research.

In this work, we have taken two completely different datasets, ADNI and Kaggle and tried comparing their results by using different optimizers. We have introduced a customized CNN model for this work. The two datasets differ in the number of classes and the number of images. The images of the ADNI dataset have undergone a set of pre-processing, but the images of Kaggle have been taken as they are from the dataset without applying any preprocessing techniques. We have also resolved the issue of class imbalance by implementing the K fold validation method, as the number of images in each of the classes differs a lot. This work will help in analysing the effect of different optimizers performance as well as how the model's performance is affected by the number of images. From the literature review, we can observe that very little research work has given importance and studied the effect of different optimizers on the model's performance. Also, much of the work has been on binary classification, which is less challenging than multiclass classification. Another important factor is that most of the works have considered an equal number of images in all the multi-classes. But we have dealt with this issue, as the number of images in each of the classes was very different.

The main contributions of the paper are listed below.

- A 2D-CNN based architecture has been proposed, considering both the ADNI and the Kaggle datasets.
- The proposed architecture has been validated with an unbalanced dataset that uses Batch Normalization (BN) and dropout for regularization and k-fold for validation.
- The proposed architecture achieved significantly high accuracy with a fixed and limited number of samples.
- Selection of the best state-of-the-art optimizer to reduce convergence time with high accuracy.
- A new corpus has been built by downloading brain images from the ADNI dataset comprising 1712 images.
- A thorough study has been done on all the important optimizers and how it impacts the model performance.

II. Literature Review

The ancient detection methods for AD classification include the use of machine learning techniques. According to a study [5] the most widely used machine learning techniques were decision trees (50%), neural networks (44%), regression (34%), SVM (34%), and Bayesian networks (20%). In [6], six different machine learning methods were applied such as decision tree, bagging, BF tree, Random Forest tree, RBF networks, and Multilayer Perceptron for the classification of Alzheimer's and Parkinson's disease. Neural network was also used along with these ML techniques. Random Forest performed best acquiring 85.17% accuracy. Comparisons of different ML techniques were conducted in [7] for detection of AD. The performances of K-Nearest Neighbors (KNN), decision trees, rule Induction, naïve bayes, random forest and Generalized Linear Model were evaluated. Generalized Linear Model (GLM) outperformed all the other classifiers with 88.4% accuracy. Extra tree classifier and decision trees are also used in the classification of AD [8]. A voting classifier was compared with the machine learning techniques Decision Tree, Random Forest, Support Vector Machine, and Gradient Boosting in [9]. Voting classifier is a method in which the probability vectors of multiple classifiers are examined and the classifier representing the highest value is chosen. In this experiment for AD detection random forest classifier performed best by achieving an accuracy of 86.92%.

TABLE I. Labels Used in Alzheimer's Disease

Abbreviation	Full name
AD	Alzheimer's Disease
CN	Cognitive Normal
MID/MD	Mild Demented
VMD	Very Mild Demented
EMCI	Early MCI
NL	Cognitive Healthy
MCI	Mild Cognitive Impairment
NC	Normal Control
MOD	Moderate Demented
ND	Non-Demented
LMCI	Late MCI
SMC	Significant Memory Concern

Recent years have witnessed that deep learning models are being widely used in image processing, and in particular, deep learning models utilizing convolutional neural networks are providing promising results in this field. The application area of CNN covers a major area of medical image analysis, including detection, segmentation, classification, and computer aided diagnosis from a wide spectrum of clinical imaging modalities. In the present study, we have surveyed some limited papers that have used CNN and its performance in the classification of AD.

The two classification types are binary and multiclass classification. In this survey, we have considered only the papers that have implemented multiclass classification. The different labels used for AD classification have been depicted in Table I.

Preprocessing is a major step, as raw MRI images consist of a lot of noise and need to be processed before feeding them to the CNN model. R. Jain *et al.* [10] performed Alzheimer's disease classification on 4800 sMRI's which were generated by using the image entropy of 150 subjects taken from ADNI. Both three way (AD vs CN vs MCI) and 2-way classification (AD vs CN, AD vs MCI and CN vs MCI) were conducted using VGG16 as the base model. The test accuracy of the three-way classification was 95.7% at 50 epochs with a batch size of 40. Kaggle's Alzheimer's dataset has been used in by Yildrim *et al.* for 4-way classification [11]. A customized ResNet50 model was used by removing the last 5 layers and adding 10 more new layers. An accuracy of 90% was achieved without any data pre-processing. Fuadeh *et al.* [12] used AlexNet architecture for Alzheimer's disease classification. A total of 664 images from Kaggle's Alzheimer's dataset were used that consisted of 200 images for Non-Demented, Very Mild Demented and Mild Demented conditions respectively and 64 images of Moderate Demented conditions. AlexNet architecture was used with different learning rates. The best accuracy was found to be 95% with 0.0001 as the learning rate and Adam as optimizer. ResNet101 was used by Bhuvaneshwari *et al.* [13] for Alzheimer's disease classification using the ADNI dataset consisting of 240 images only. Each of the classes AD, CN and MCI consisted of 80 images. Image segmentation and skull stripping were used as part of image preprocessing. An accuracy of 96.3% was achieved in the three-way classification. Experiments were done using CNN and VGG16 for 4-way Alzheimer's disease classification by Ajagbe *et al.* [14]. Kaggle's Alzheimer's dataset was utilized and it achieved an accuracy of only 0.71 in CNN and 0.77 in the VGG models. Murugan *et al.* [15] developed a model DEMNET which consisted of convolutional layers followed by max pooling, batch normalization and dropouts. Different dropout values 0.7, 0.5 and 0.2 were used for the first, second and third dense layers respectively. The SMOTE technique was used for handling class imbalance. Kaggle's Alzheimer's dataset achieved an accuracy of 94% at 50 epochs with RMSprop as optimizer. Experiments on two Alzheimer's datasets from Kaggle consisting of 6400 images and 6330 images was conducted by Sharma *et al.* [16]. The first dataset consisted of 4 labels and the second dataset consisted of 3 labels. VGG 16 was used as the pre-trained model and Adam was used as the optimizer. An accuracy of 90% and 71% was achieved on dataset1 and dataset2 respectively. Shanmugam *et al.* [17] experimented using 7800 images from the ADNI dataset consisting of 5 labels: AD, LMCI, MCI, EMCI, CN. Three pretrained models were used namely: GoogleNet, AlexNet and ResNet. SGD with momentum value of 0.9 and learning rate of 10^{-4} was applied to the transfer learning models. ResNet outperformed GoogleNet and AlexNet by acquiring an accuracy of 98.63% at 100 epochs.Classification of AD was conducted with different classifier models like LeNet, AlexNet, VGG-16, VGG-19, Inception-V1, Inception-V2, Inception-v3, ResNet-50, ResNet-101, ResNet50-V2, ResNet152-V2, InceptionResNet, MobileNet, MobileNet-V2, EfficientNet-B0, EfficientNet-B7, Xception, NasNet-A, NasNet-C, and DenseNet-121 by Hazarika *et al.* [18]. Both two way and three-way classification were done using the ADNI dataset with labels AD, CN and MCI. Improved DenseNet121, where all the convolutional layers of the architecture were replaced by depthwise convolutional layers, outperformed all the other models achieving 88%accuracy.

Two experiments regarding Alzheimer's disease classification were conducted by Marwa *et al.* [19]. The first experiment considered the classification of CN, AD and MCI and the second classification was considered as a local classification which considered classification of MCI into a Very Mild Dementia (VMD), Mild Dementia (MD), and Moderate Dementia (MoD). The OASIS dataset consisting of 6400 images was used. Image normalization as well as data augmentation was conducted on the images. A CNN model was used with Adam optimizer for 100 epochs. The accuracy of the CNN model was found to be 99.68% for the three-way classification.

Dar *et al.* [20] conducted a 5-way classification consisting of the labels CN, MCI, EMCI, LMCI and AD with a total of 2900 images taken from the ADNI dataset. Data normalization and unit vector normalization was carried out as part of data pre-processing. MobileNet was used as a classifier with RMSprop as optimizer and learning rate of 0.00001. An accuracy of 96.22% was achieved. ADNI dataset was used by Raza *et al.* [21] for conducting a 5-way classification consisting of labels Alzheimer's Disease (AD), Non-Cognitive (NC), Late Mild Cognitive Impairment (LMCI), and Mild Cognitive Impairment (MCI). A total of 5016 brain MRI images were considered from the ADNI dataset. DenseNet 169 was used as the base model. A series of data pre-processing was carried out including, skull stripping, Gray Matter (GM) segmentation, Montreal Neurological Institute (MNI) space normalization and smoothing. An accuracy of 93.11, 96.82 and 97.84 was achieved at 10, 25 and 50 epochs respectively.Kaggle's Alzheimer's dataset was used both for binary and 4-way classification using a lightweight CNN by Latif *et al.* [22]. For the binary classification an accuracy of 99.2% was achieved at 80 epochs and for the 4-way classification 95.93% was achieved at 90 epochs. Balasundaram *et al.* [23] used two datasets Kaggle's Alzheimer dataset and OASIS 2 MRI for Alzheimer's disease classification. Hippocampus Segmentation was carried out as part of data pre-processing. Three models were experimented on the dataset namely, the simple multilayer model, CNN and ResNet50. It was observed that the performance of the models improved while using segmented images rather than whole brain images. CNN outperformed the other two models by achieving 94% accuracy. Ullah *et al.* [24] used Kaggle's Alzheimer's dataset of 6400 images for

Alzheimer's disease stage detection. The dataset was increased to a size of 10,074 images through the technique of data augmentation. A CNN model was used for 100 epochs with a batch size of 64. An accuracy of 99.38% was achieved for the four labels used.

Alzheimer's disease classification was carried out using the ADNI dataset considering 4482 images belonging to three classes AD, CN and MCI by Awarayi *et al.* [25]. Using data augmentation, the size of the dataset was increased to 26,892. They have used the Neural Architecture Search (NAS) framework for CNN architecture as it saves time required for model development as well as for parameter hypertuning. The model achieved an accuracy of 97.17%. 10-fold cross validation was used to improve the classifier's performance. Assaduzzaman *et al.* [26] conducted Alzheimer's disease classification on the Kaggle's Alzheimer's dataset. Seven data pre-processing techniques like Contrast Limited Adaptive Histogram Equalization (CLAHE), bilateral filter, and green fire blue filter, are applied to enhance image quality and eliminate artefacts. A customized CNN ALSA-3 was developed for the classification. They have analysed that batch size affects accuracy and so experimented with three batch sizes: 32, 64 and 128. The 7-fold cross validation technique was applied. The best accuracy was 99.50% when batch size 64 was considered with Adam as optimizer and a learning rate of 0.001. A Siamese 4D AlzNet comprising of four parallel convolutional neural networks and customized transfer learning models namely: Frozen VGG16, Frozen VGG19 and customized AlexNet was used for Alzheimer's disease classification by Mehmood *et al.* [27]. Skull striping, image registration, image normalization and segmentation are the data processing techniques applied to the dataset. Although four labels NC, MCI, LMCI, AD and were considered but binary classification was performed between the labels. The Siamese 4D AlzNet outperformed the transfer learning models by achieving an accuracy of 95.07%, 96.75%, 96.82%, 95.43 % in NC vs AD, NC vs LMCI, NC vs MCI and MCI vs AD respectively. Frozen VGG 19 performed well in LMCI vs AD by acquiring 80.70% accuracy. Shastri [28] used three datasets namely, Kaggle comprising 6400 images across 3 classes, ADNI comprising 1296 images across 5 classes and a dataset containing 5154 images across three classes: AD, Confidence Interval (CI), and Cognitive Normal (CN). A customized CNN was used for classification. The model acquired accuracy of 96.02% in Kaggle, 71.03% in ADNI and 98.84% in the third dataset. The ADNI dataset with 6 labels SMC, NC, LMCI, MCI, AD, and EMCI with a total of 1598 images were considered by Singh and Kumar in [29] for Alzheimer's Disease classification. Data preprocessing techniques like reorientation, registration, brain extraction, shading correction, and segmentation were carried out for enhancing the quality and consistency of the images. Many CNN models like EfficientNet, MobileNet, DenseNet, Resnet, AlexNet, InceptionV2, and NASNet were used to classify the images. EfficientNet outperformed all the other models by achieving 99.8% accuracy. Heurta *et al.* [30] used oversampling to balance the number of images in the class that contains fewer images thus making 3200 images in all the classes of the Kaggle's Alzheimer's dataset. Adam and SGD optimizers were used with learning rate value 0.001,0.002 and 0.0005. Adam with learning rate 0.0005 performed best with99% accuracy. 26 Keras pretrained models were used by Srividhya *et al.* [31] for Alzheimer's disease classification with four labels. A total of 1296 images were considered from the ADNI dataset. The SMOTE technique was used for handling class imbalance and the size of the dataset increased to 2900 images. ResNet-50v2 performed best achieving 91.84% accuracy. Hussain *et al.* [32] conducted Alzheimer's disease classification using different CNN architectures like AlexNet, GoogleNet and MobileNetV2 on two datasets Kaggle and OASIS. Data augmentation was applied on the dataset. The models were tested on three optimizers namely SGDM, Adam and RMSProp. AlexNet and GoogleNet performed well with Adam optimizer by acquiring an accuracy of 99.4% and 98.0% respectively at 25 epochs. MobileNetV2 attained an accuracy of 96.5% using SGDM as an optimizer at 25 epochs. Gondalia and Popat [33] conducted Alzheimer's classification using Kaggle's Alzheimer's dataset. Data augmentation was conducted on the 6400 images to expose the classifier to variations of images so that it does not memorize anything and gives better accuracy results in the testing dataset. An accuracy of 93.82% was found. A summary of the previous works that have used MRI images and conducted multiclass classification is depicted in Table II.

TABLE II. PREVIOUS WORKS

Author	Model	Dataset	Class Labels	Number of Samples	Accuracy
R. Jain *et al.* [10]	VGG16	ADNI	AD, CN, MCI	4800	95.7%
Muhammed Yildirim and Ahmet Cinar [11]	ResNet50	Kaggle	MD, MOD, VMD, ND	6400	90%
Y.N. Fuadah *et al.* [12]	AlexNet	Kaggle	MD, MOD, VMD, ND	664	95%
P.R. Buvaneswari*et al.* [13]	ResNet101	ADNI	AD, CN, MCI	240	96%
S.A. Ajagbe *et al.* [14]	CNN VGG16 VGG19	Kaggle	MD, MOD, VMD, ND	6400	71 % for CNN 77% for VGG16 77.66% for VGG19
S. Murugan *et al.* [15]	CNN	Kaggle	MD, MOD, VMD, ND	6400	94%

S. Sharma *et al.* [16]	VGG16	Kaggle	MD, MOD, VMD, ND	6391 for Dataset1 6330 for Dataset2	90% for Dataset1 71% for Dataset2
J.V. Shanmugam *et al.* [17]	AlexNet, GoogleNet ResNet-18.	ADNI	AD, CN, MCI, EMCI, LMCI	7800	96% in AlexNet 94% in GoogleNet 97.51% in ResNet-18
R.A. Hazarika *et al.* [18]	LeNet, AlexNet, VGG-16, VGG-19, Inception-V1 (Googlenet), Inception-V2, Inception-v3, ResNet-50, ResNet-101, ResNet50-V2 ResNet152-V2, InceptionResNet, MobileNet, MobileNet-V2, EfficientNet-B0, EfficientNet-B7, Xception, NasNet-A, NasNet-C DenseNet-121.	ADNI	AD, CN, MCI,	15,120	DenseNet-121 outperformed other models achieving 88% accuracy
Marwa El-Geneedy *et al.* [19]	CNN	OASIS	ND, MOD, MD, VMD	6400	99.68%
Mohiud din Dar *et al.* [20]	MobileNet	ADNI	CN, MCI, EMCI, LMCI, AD	2900	96%
Noman Raza *et al.* [21]	Dense-Net169	ADNI	AD, LMCI, MCI, NC	5016	97.84%
AAA. El-Latif [22]	DNN	Kaggle	AD, MOD, ND, VMD	6400	95.93%
A. Balasundaram *et al.* [23]	CNN	Kaggle	AD, MOD, ND, VMD	6400	94%
Ullah and Jamjoom [24]	CNN	Kaggle	MD, MOD, ND, VMD	10,074	99%
N.S. Awarayi *et al.* [25]	CNN	ADNI	AD, CN, MCI	26,892	97%
Md Assaduzzaman *et al.* [26]	CNN	Kaggle	AD, MOD, ND, VMD	6400	99%
A. Mehmood *et al.* [27]	CNN	ADNI	NC, MCI, LMCI, AD	11,465	NC vs AD: 95.07% NC vs LMCI: 96.75% NC vs MCI: 96.02 MCI vs AD: 95.43 LMCI vs AD: 79.16
K. Aditya Shastri [28]	CNN	Kaggle ADNI	For Kaggle: AD, MOD, ND, VMD For ADNI: AD, CN, EMCI, LMCI,	Kaggle:6400 ADNI: 1296	Kaggle: 96.02% ADNI: 71.02%
Singh and Kumar [29]	EfficientNet	ADNI	SMC, NC, LMCI, MCI, AD, and EMCI	1548	EfficientNet: 99.8%
Heurta *et al.* [30]	CNN	Kaggle	MD, MOD, VMD, ND	12,800	99%
Srividhya *et al.* [31]	ResNet50V2	ADNI	NM, EMCI, MCI, LMCI, AD	1296	91.84%
M Z Hussain *et al.* [32]	CNN	Kaggle OASIS	MD, MOD, VMD, ND	Kaggle: 10,254 OASIS: 2744	Kaggle: 99% OASIS: 98%
Gondalia *et al.* [33]	CNN	Kaggle	MD, MOD, VMD, ND	6400	94%

III. METHODOLOGY

A. Dataset

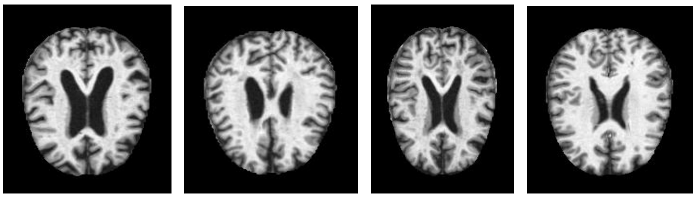
In the present study, both the ADNI [34] and the Kaggle Alzheimer's disease dataset [35] have been considered for classification.

- **Kaggle Dataset:**The Kaggle dataset consists of 6400 images with a size of 176×208.The images demonstrated the axial view of the brain. The dataset has four classes of images: mild demented, moderately demented, non-demented and very mild demented. The images in the Kaggle dataset are in JPEG format. The detailed data statistics with class names and number of images for the Kaggle dataset are described in Table III.

TABLE III. DATA STATISTICS OF KAGGLE DATASET

Class Name	Number of images
Mild Dementia	896
Moderate Dementia	64
Non-Dementia	3200
Very Mild Dementia	2240

The sample images of all the classes present in the Kaggle dataset is shown in Fig. 1:



Mild-demented Moderate-demented Non-demented Very mild demented

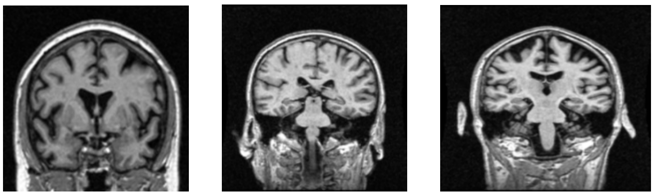
Fig. 1. MRI samples of Kaggle dataset.

- **ADNI Dataset:** ADNI is a huge dataset providing MRI images of the brain. We have downloaded a subset of the dataset consisting of 1712 images. As the dataset provides different views of the brain, we have selected the coronal view of the brain. The images are typically of size 256×256 matrices with a voxel size of approximately 1.33 mm×1 mm×1 mm. Initially the images were in Neuroimaging Informatics Technology Initiative (NIfTI) format but after data preprocessing they were converted to BMP format. The dataset consists of three classes of images: Alzheimer Disease (AD), Cognitive Normal (NC) and Mild Cognitive Impairment (MCI). The detailed data statistics with class names and number of images of the ADNI dataset are described in Table IV.

TABLE IV. DATA STATISTICS OF ADNI DATASET

Class Name	Number of images
AD	352
CN	494
MCI	866

The sample images of all the classes of images present in the ADNI dataset is shown in Fig. 2:



Alzheimer'S Disease (AD) Cognitive Normal (CN) Mild Cognitive Normal (MCI)

Fig. 2. MRI samples of ADNI dataset.

From Tables III and IV it is obvious that both datasets have class imbalance issues. In the Kaggle dataset the biased class is Non-Dementia with 3200 images and in the ADNI dataset the biased class is the Mild Cognitive Impairment with 866 images. While training, the model might lead to overfitting. To handle the problem of overfitting, we have applied the stratified k-fold cross validation technique, which is discussed in the later section.

B. Preprocessing Brain Images

Preprocessing is an essential step as MRI images may contain noise due to the imaging procedure or the way in which the MRI images were acquired from different sources. Images should be clean enough for better classification results. The different image pre-processing techniques applied to both the Kaggle and the ADNI datasets are described below:

1) Image resizing

The image sizes in both datasets were resized to 224×224, as the original sizes of both datasets, Kaggle and ADNI were different. This is the image size mostly accepted as input size in deep learning models, as a big image size requires more memory, as well as more time is consumed while training and testing.

2) Normalization

Normalization is very much essential because if we pass an image as it is to the classification model, the computation might become very complex due to the high numeric values. Normalization is the process of making the features of the image generated of uniform scale, which makes the classification model easy and suitable for further processing. As the pixel values of an image can range from 0 to 255, normalization scales it down in the range [0,1] or [−1,1]. Normalization is performed with the use of the following equation:

$$\text{Img} = 1/255.0$$

3) Zooming

The zoom augmentation technique is magnifying the original image which may lead to adding new pixels around the image or interpolating the image. It usually accepts a float value and here in this study we have used a zoom range of 0.20.

Besides this, the images in ADNI had to undergo another set of pre-processing before applying the above techniques. The MRI images in the ADNI dataset were initially in NIFTI format. Every human brain differs in volume, shape and size depending upon the different populations residing in different environments and having different genetic developments [36]. To provide finer details of the anatomy of the brain, it is necessary to have population specific brain templates that capture, quantify, and visualize the brain anatomy that can be used later in many structural, functional, and physiological studies for better interpretation [37]. So, it is necessary to convert the images into a standard brain template. We have used the MNI-152 (Montreal Neurological Institute) template. The advantage of the MNI-152 brain template is that it provides full head coverage and also provides more detailed information from the top portion of the brain to the bottom portion of the cerebellum [38, 39]. The pre-processing techniques that were applied only to the images in the ADNI dataset are described below.

4) Image registration

We have conducted the image registration using the FLIRT algorithm [40].

Step 1: Given a reference image i and a moving image J, the algorithm uses a multistart, multiresolution global optimization method to find the affine transformation that minimizes the disparity between the reference image and the moving image.

Step 2: A standard way of formulating the above mathematical problem is to construct a cost function thatquantifies the dissimilarity between two images and then search for the transformation (T*) that gives the minimum cost. Mathematically, it can be written as:

$$T^* = \underset{T \in S_T}{\arg\min}\ C(Y, T(X)),$$

where S_T is the space of allowable transformations, C (I1, I2) is the cost function, and $T(X)$ represents the image X after it has been transformed by the transformation T. Here, only linear transformation is considered. The most commonly used intensity-based cost functions are Least Squares (LS); Normalized Correlation (NC); Woods (W); Correlation Ratio (CR); Mutual Information (MI); and Normalized Mutual Information (NMI).

Step 3: It divides the process of searching for the best transformation into four different resolution scales: 8, 4, 2 and 1mm.

Step 4: At each scale, the two images are resampled after initial pre-blurring so that they have isotropic voxels of size equal to the scale size.

Step 5: Choose Powell's method as the local optimization method.

Step 6: To estimate the final transformation sufficiently accurately, a coarse search of the cost function at this resolution is used, as it avoids misregistration. The search can be divided into three stages.

a) A coarse search over the rotation parameters with a full local optimization of translation and global scale for each rotation was tried.
b) A finer search over rotation parameters, but with only a single cost function evaluation at each rotation.
c) A full local optimization for each local minimum was detected in the previous stage.

Step 7: It is unlikely that the first stage in this process will get very close to the correct rotation, but the second stage should get close enough for the local optimization in the last stage to give a good estimate.

Step 8: Following the previous search stage (at 8 mm scale), there are usually several local minima selected as candidates for initialising more detailed searches for the global minimum.

Step 9: This stage (at 4 mm) performs a local optimization for the best of these candidate transformations.

Step 10: It takes several perturbations of the candidate transformations and performs local optimization of these perturbations.

Step 11: The single best solution is selected from these optimization results.

Step 12: Since the cost function evaluations take 8 times longer at the 1mm scale than at the 2 mm scale and 512 times longer than at the 8 mm scale, only a single pass of the local optimization is done at the 1mm scale.

Step 13: The registration solution represents the outcome of this single pass.

5) Image segmentation

After brain image registration, the actual part of the brain is extracted using the Brain Extraction Tool (BET) algorithm [41]. The algorithm works as follows:

Step 1: The robust image intensity minimum t_2 and intensity maximum t_{98} are estimated.

Step 2: A brain/background threshold 't' is estimated, which lies 10% on the way between t_2 and t_{98}.

Step 3: This threshold 't' is used to estimate the position of the Centre of Gravity (COG) of the brain.

Step 4: The mean radius of the brain or head is estimated by counting all voxels with intensity greater than t, considering the voxel volume.

Step 5: The median intensity t_m of all points within a sphere of the estimated radius and centred on the estimated COG is found to initialize the brain surface model.

Step 6: The brain surface is modelled by a surface tessellation using connected triangles. The initial model is a tessellated surface generated by starting with an icosahedron and iteratively subdividing each triangle into 4 smaller triangles while adjusting each vertex's distance from the centre to form as spherical a surface as possible.

Step 7: Each vertex in the tessellated surface is updated by estimating where that vertex should move to improve the surface.

Step 8: Repeat step 7 to get an optimum surface.

Step 9: On the final tessellated surface, the brain gets extracted.

The flowchart of the BET algorithm [41] is demonstrated in Fig. 3.

The final brain image obtained after implementing both the FLIRT and the BET algorithm is depicted in Fig. 4.

C. Proposed Framework

The proposed CNN model consists of five major steps: (i) Pre-processing, (ii) Data split, (iii) K-fold technique for handling data imbalance, (iv) Feature extraction and (v) Image classification. The framework of the CNN model is depicted in Fig. 5.

D. Network Structure of the Proposed Architecture

The proposed model consists of five Conv2D layers of kernel size 3×3 each, followed by five MaxPooling layers of kernel size 2×2, and two FC layers with some other components like Batch Normalization (BN) and dropout layers, as shown in Fig. 6. Each of the CN layers follows a BN layer to make the training process of the model more efficient. The number of kernels used in the layers is 16, 32, 32, 64 and 128 respectively. This is followed by a flattening layer and two dense layers, including an output layer. The diagrammatic representation of the CNN model is demonstrated in Fig. 6. Here we have shown the proposed architecture for the ADNI dataset. For the Kaggle dataset the setup of the model will be the same except for the output layer which will consist of 4 classes instead of 3.

We have considered K = 5 in our experiment. The summary of the model has been depicted in Table V.

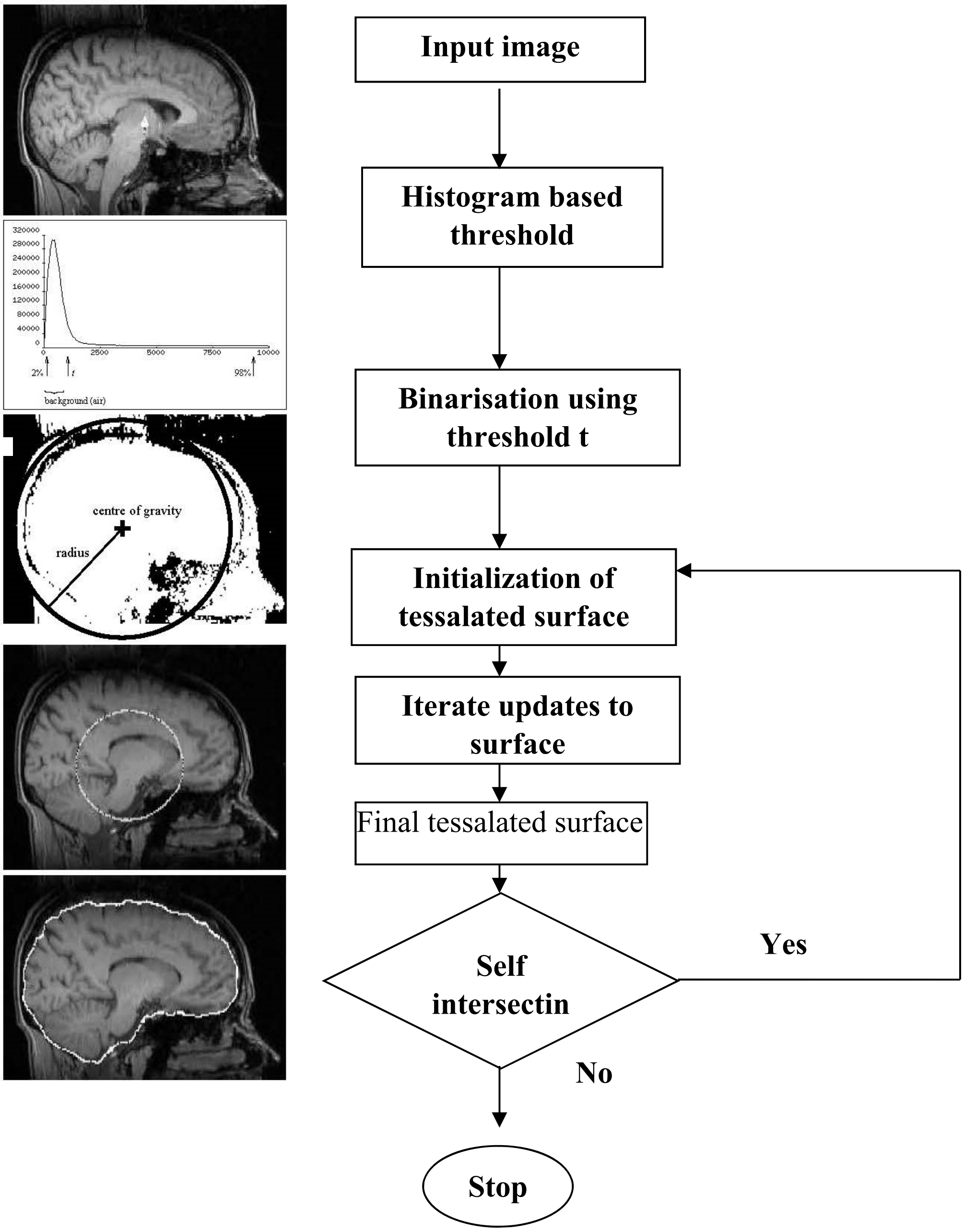


Fig. 3. Flowchart of BET algorithm.

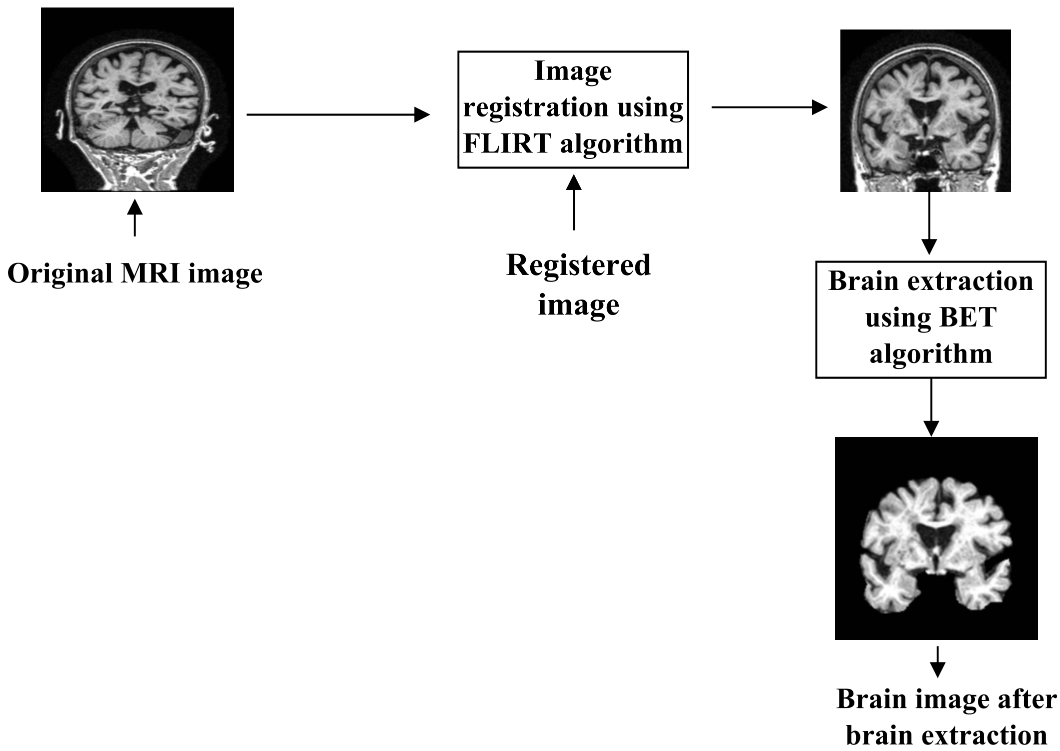


Fig. 4. Image pre-processing was done using FLIRT and BET algorithms.

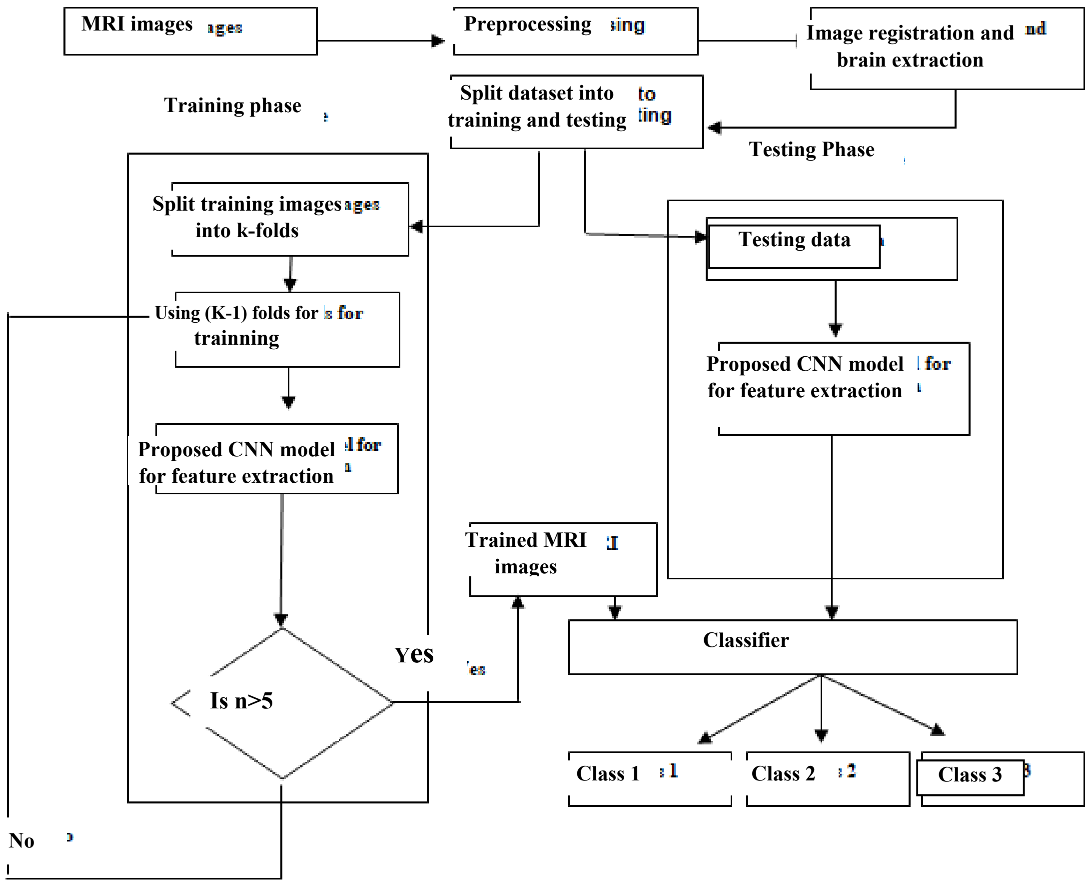


Fig. 5. Proposed CNN framework.

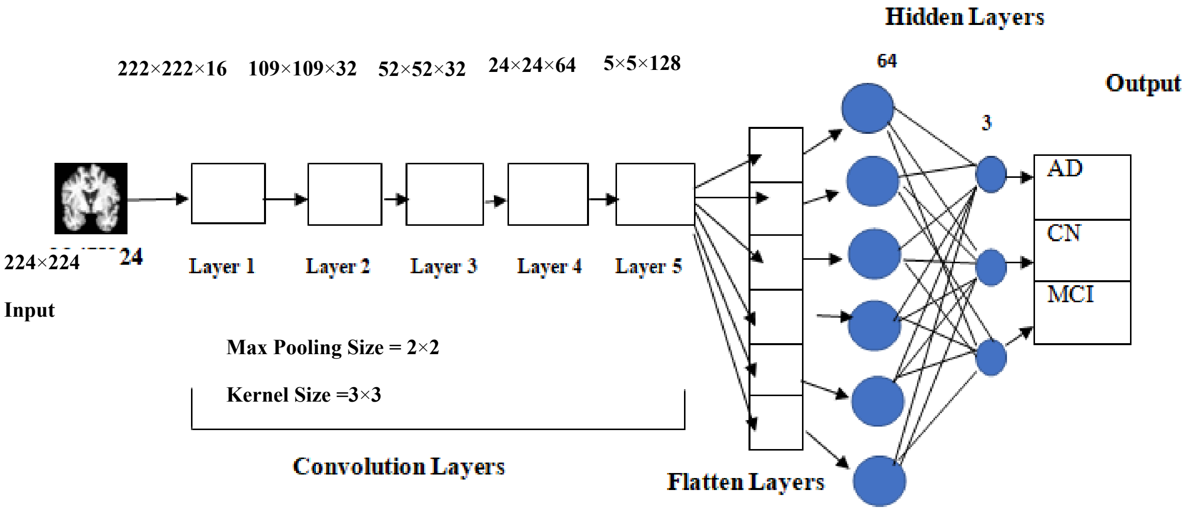


Fig. 6. Diagrammatic representation of the proposed CNN model.

TABLE V. NETWORK ARCHITECTURE OF THE PROPOSED CNN MODEL

Layer (type)	Output Shape	Param #
conv2d	(None, 222, 222, 10)	448
max_pooling2d_4 (MaxPooling2D)	(None, 111, 111, 10)	0
conv2d_1 (Conv2D)	(None, 54, 54, 32)	4640
max_pooling2d_1 (MaxPooling2D)	(None, 54, 54, 32)	0
conv2d_2 (Conv2D)	(None, 52, 52, 32)	9248
max_pooling2d_2 (MaxPooling2D)	(None, 26, 26, 32)	0
conv2d_3 (Conv2D)	(None, 24, 24, 64)	18496
max_pooling2d_3 (MaxPooling2D)	(None, 12, 12, 64)	0
conv2d_4 (Conv2D)	(None, 10, 10, 128)	73856
max_pooling2d_4 (MaxPooling2D)	(None, 5, 5, 128)	0
flatten (Flatten)	(None, 3200)	0
dense (Dense)	(None, 64)	204864
dropout (Dropout)	(None, 64)	0
batch_normalization (BatchNormalization)	(None, 64)	256
dense_1 (Dense)	(None, 3)	195

Notes: Total params:312003 (1.19 MB); Trainable params:311875 (1.19 MB); Non-trainable params:128 (512.00 Byte)

E. Parameter Setting for the Proposed Model

The training process of the proposed model has been computed iteratively to update the different parameters. There are some important parameters, such as batch size, regularization parameter, number of epochs, and learning rate, that control the performance of the classification. In deep learning, batch size is a very important parameter that influences the system update phase. In the present study, different batch sizes have been considered to train the model, and finally, the optimum result is found when it is set to 32. A deep neural network tends to overfit during training when the number of tunable parameters is high compared to the number of samples in the training set. In such a situation, both Batch Normalization (BN) and dropout have been used for regularization during training [42]. In the present study, BN, along with a 20% dropout rate, has been used for regularizing the training process. Another hyperparameter, 'epoch' refers to the number of iterations the model passes through the training samples. In each iteration, every training sample in the training dataset gets a chance to update the model parameter. It permits the learning algorithm to run until the optimal performance of the model is achieved. The number of epochs changes with different learning algorithms. A literature review reveals that there is no standard algorithm or mathematical model to set the value of the epoch. The best way to set the value of epochs is to gradually increase the value until the validation accuracy starts decreasing, even when the training accuracy increases. In the present study, we have tested both datasets using different values for epochs such as 5, 10, 20, 35, 50, 80, and 100. The learning rate is another important hyperparameter that determines the value where the weights of the model are adjusted concerning the loss gradient. The smaller the learning rate, the slower the convergence, a higher learning rate may overshoot the solution region. Therefore, the selection of the optimum learning rate plays an important role in achieving a good result. In our study, we have not set any learning rate, and the default value of 0.001 is considered for all the optimizers.

F. Optimizer Selection

Optimization algorithms try to minimize the error function by updating the weight vectors in a deep neural network. The Gradient Descent algorithm faces some challenges, such as the vanishing gradient problem, slow learning rates, etc. To solve these challenges, different optimization techniques have been proposed. These optimization algorithms aim to make gradient descent more efficient and faster. The proposed CNN model has been compared with different optimization algorithms to minimize the error function by updating the weight vectors. The proposed model has been computed with Stochastic Gradient Descent (SGD), SGD with momentum, AdaptiveGradient (AdaGrad), Root Mean Square Propagation (RMSprop), AdaptiveDelta (Adadelta) and Adaptive Moment Estimation (Adam) [43–45].

1)Stochastic Gradient Descent (SGD)

SGD is a type of gradient descent that updates the parameters one by one. This can make SGD faster than batch gradient descent because, in batch gradient descent, we need to have access to all training samples at once.

Algorithm: Stochastic Gradient Descent XX

Step 1: Randomly shuffle the data set of size *m*

Step 2: Select a learning rate α.

Step 3: Select initial parameter values Was the starting point.

Step 4: Update all parameters from the gradient of a single training example *X* input vector i.e. compute

$$W_{t+1} = W_t - \alpha \times \nabla_W L(W_t, X)$$

where α is the learning rate.

Step 5: Repeat Step 4 until a local minimum is reached.

The advantage of SGD is the frequent updates immediately give an insight into the performance of the model and the rate of improvement.

2) SGD with momentum optimizer

One of the approaches that can be used to make the gradient descent learning algorithm very efficient is the momentum optimizer. The number of epochs can be reduced, considering the concept of momentum. If we consider both the gradient force and the momentum force at a particular point, the net force will increase. As a result, the solution moves faster towards the minimum location.

The gradient descent algorithm can be written as:

$$SGD \rightarrow W_{t+1} = W_t - \nabla_w L(W_t)$$

By adding the momentum term, the weight updating rule will be:

$$W_{t+1} = W_t + \gamma v_{t-1} - \nabla_w L(W_t)$$

Here, γv_{t-1} is the momentum term and $\nabla_w L(W_t)$ is gradient term. The gradient descent with momentum improves the rate of convergence. If we assume

$$v_t = -\gamma v_{t-1} + \nabla_w L(W_t)$$

then

$$W_{t+1} = W_t - v_t$$

The main disadvantage of this algorithm is that it requires the hyperparameters to be set manually, which determines the learning rate. Moreover, the algorithm uses the same learning rate for all dimensions. It may require different learning rates in different dimensions.

Algorithm: Adadelta

Step1: Compute the gradient of the loss function at location *t*, taking *X* as the input vector w.r.t. weight vector

$$\mathrm{g_t} = \frac{1}{\mathrm{n}} \sum_{\forall \mathrm{X} \in \mathrm{Minibatch}} \nabla_{\mathrm{w}} \mathrm{L}(\mathrm{W_t}, \mathrm{X})$$

Here, *n* is the number of samples used for training the network model. g_t is the gradient at instant *t*, W_t is a weight vector.

Step2: Accumulate the Gradient over a window of size *w*:

$$\mathrm{r_t} = \beta \mathrm{r_{k-1}} + (1 - \beta)\mathrm{g_t} \circ \mathrm{g_t}$$

Step3: Compute update:

$$\Delta \mathrm{W_t} = -\frac{RMS[\Delta r]_{t-1}}{RMS[g]_t} + \mathrm{g_t}$$

Where,

$$RMS[g]_t = \sqrt{\in \mathrm{I} + \mathrm{r_t}}$$

Here *I* is the vector where all the components are 1 and r_t is also a vector and is a very small value.

Step 4: Apply update:

$$W_{t+1} = W_t + \Delta W_t$$

where

$$\Delta \mathrm{W_t} = -\frac{\eta}{RMS[g]_t}$$

3) Adadelta

AdaDelta is an improvement over Adagrad just like RMSProp. AdaDelta is a closely related algorithm to RMSProp. In the case of RMSProp, we take the exponentially decaying average of the squared gradient and discard history from the extreme past. In AdaDelta, instead of taking the exponentially decaying average of the squared gradient, it computes the moving window average of the gradient. The window is fixed in size, and it moves forward in every iteration. It computes the average over the samples in a window in each iteration. Both AdaDelta and RMSProp give almost similar performance.

Adadelta's main advantages over Adagrad are that it doesn't need a default learning rate. Moreover, it doesn't decrease the learning rate as aggressively and monotonically as Adagrad.

4) Adaptive Gradients (Adagrad)

The Adadelta algorithm tries to adaptively scale the learning rate in different dimensions. This problem has been solved in Adagrad by tuning the learning rate in all the dimensions of the weight vector. Moreover, the scale parameters of the scale factor to scale the learning rate in different dimensions are inversely proportional to the square root of the sum of the historical squared values of the gradient. As a result, the parameters that have the largest partial derivative of the loss will have a rapid decrease in their learning rate. Parameters with small partial derivatives will have a relatively small decrease in learning rate.

Algorithm: Adaptive Gradients (Adagrad)

Step 1: Compute the gradient of the loss function at location *t*, taking *X* as the input vector w.r.t. weight vector.

$$g_t = \frac{1}{n} \sum_{\forall X \in Minibatch} \nabla_w L(W_t, X)$$

Here *n* is a number of samples used for training the network model. g_t is gradient at instant *t*, W_t is weight vector.

Step 2: Compute the square of the different components of the gradient and then sum them up from 1 to *t*.

$$r_t = \sum_{\tau=1}^{t} g_\tau \circ g_\tau$$

Here o is element wise product

Step 3: Update W_t

$$W_{t+1} = W_t - \frac{n}{\sqrt{\in I + r_t}} \circ g_t$$

Here *I* is the vector where all the components are 1 and r_t is also a vector and ∊ is a very small value.

The general form of the equation will be

$$W_{t+1}^{(i)} = W_t^{(i)} - \frac{n}{\sqrt{\in + r_t^{(i)} \circ g_t^{(i)}}}$$

Algorithm: Adaptive Moments (Adam)

Step 1: Compute the gradient of the loss function at location *t*, taking *X* as the input vector w.r.t. weight vector.

$$g_t = \frac{1}{n} \sum_{\forall X \in Minibatch} \nabla_w L(W_t, X)$$

Here *n* is the number of samples used for training the network model. g_t is gradient at instant *t*, W_t is weight vector.

Step 2: Compute bias-corrected first and second moments

$$\widehat{s_t} = \frac{s_t}{1-\beta_1} \widehat{r_t} = \frac{r_t}{1-\beta_2}$$

Here $\widehat{s_t}$ is bias-corrected first moment and $\widehat{r_t}$ is the bias-corrected second moment.

Step 3: Update W_t

$$W_{t+1} = W_t - \eta \frac{\widehat{s_t}}{\sqrt{\in I + \widehat{r_t}}}$$

Here *I* is the vector where all the components are 1 and r_t is also a vector and ∊ is a very small value.

The advantage of Adagrad is that it adaptively scales the learning rate for different dimensions by normalizing with respect to the gradient magnitude in the corresponding dimension. Moreover, it converges rapidly when applied to convex functions. But the limitation is that if the function is non-convex, then it may find a locally convex region, and in such a case, the algorithm quickly converges at that minimum convex region.

5) Adaptive Moments (Adam)

It is a variant of the combination of RMSProp and momentum. Here, both first-order and second-order momentum have been considered. In addition, Adam incorporates one more term that tries to correct the bias by initializing to zero at time t = 0.

6) RMSProp

It tries to overcome the problem of the Adagrad algorithm, i.e., the vanishing learning rate. In the RMSProp algorithm, instead of taking the cumulative sum of squares of the gradients, it takes the exponentially decaying average of the squared gradient and discards history from the extreme past. As a result of this, the algorithm converges rapidly. Once it reaches the locally convex surface, the algorithm initializes at that point. So, RMSProp does not consider the accumulated sum of squares of the gradient from the beginning.

Algorithm: RMSProp

Step 1: Compute the gradient of the loss function at location *t* taking *X* as the input vector w.r.t. weight vector.

$$g_t = \frac{1}{n} \sum_{\forall X \in Minibatch} \nabla_w L(W_t, X)$$

Here n is a number of samples used for training the network model. g_t is gradient at instant *t*, W_t is weight vector.

Step 2: Compute the exponentially decaying $average$ square gradient.

$$r_t = \beta r_{k-1} + (1-\beta) g_t \circ g_t$$

where β is very small quantity

Step 3: Update W_t

$$W_{t+1} = W_t - \frac{n}{\sqrt{\in I + r_t}} \circ g_t$$

Here *I* is the vector where all the components are 1 and r_t also a vector and ∈ is a very small value.

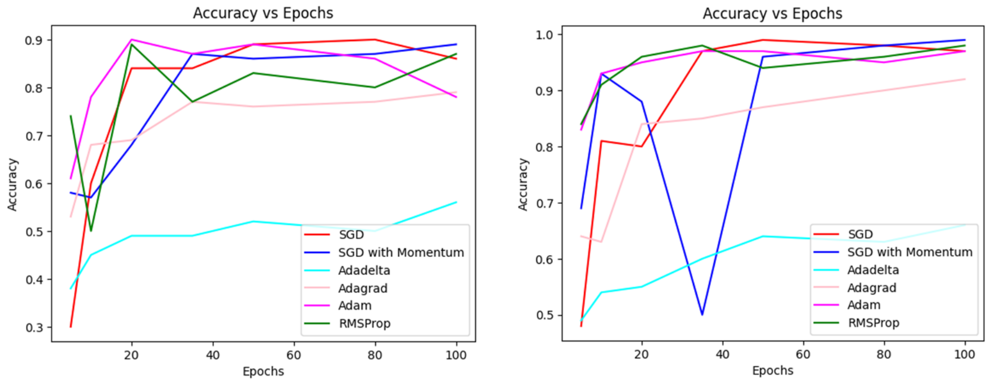
IV. Results and Discussion

The efficiency of the proposed model has been evaluated for six different optimizers, considering both the dataset ADNI and Kaggle, as depicted in Table VI. From Table VI, it is seen that in the ADNI dataset, the Adam optimizer gives a maximum accuracy of 90% at 20 epochs, whereas SGD gives a maximum accuracy of 90% at 80 epochs. In the case of the Kaggle dataset, SGD with a momentum optimizer gives a maximum accuracy of 99% at 100 epochs.

The diagrammatic representation of the accuracy results of both datasets, ADNI and Kaggle, is depicted in Fig. 7(a) and (b) respectively.

TABLE VI. Test Accuracies of Different Optimizers at Different Epochs

Dataset	Epochs	SGD Train	SGD Test	SGD with momentum 0.9 Train	SGD with momentum 0.9 Test	Adadelta Train	Adadelta Test	Adagrad Train	Adagrad Test	Adam Train	Adam Test	RMSprop Train	RMSprop Test
ADNI	5	0.32	0.30	0.63	0.58	0.40	0.38	0.59	0.53	0.69	0.61	0.78	0.74
	10	0.68	0.60	0.59	0.57	0.51	0.45	0.73	0.68	0.85	0.78	0.61	0.50
	20	0.84	0.84	0.74	0.68	0.52	0.49	0.81	0.69	0.97	0.90	0.98	0.89
	35	0.97	0.84	0.98	0.87	0.51	0.49	0.86	0.77	0.95	0.87	0.93	0.77
	50	0.97	0.89	0.95	0.86	0.60	0.52	0.85	0.76	0.97	0.89	0.97	0.83
	80	1.00	0.90	0.99	0.87	0.60	0.50	0.95	0.77	0.99	0.86	0.98	0.80
	100	0.99	0.86	0.97	0.89	0.61	0.56	0.89	0.79	0.99	0.78	0.97	0.87
KAGGLE	5	0.51	0.48	0.72	0.69	0.49	0.49	0.68	0.64	0.86	0.83	0.88	0.84
	10	0.84	0.81	0.95	0.93	0.54	0.54	0.64	0.63	0.95	0.93	0.93	0.91
	20	0.96	0.80	0.93	0.88	0.55	0.55	0.87	0.84	0.98	0.95	0.98	0.96
	35	0.99	0.97	0.50	0.50	0.60	0.60	0.87	0.85	0.99	0.97	0.99	0.98
	50	1.00	0.99	0.99	0.96	0.68	0.64	0.88	0.87	0.98	0.97	0.98	0.94
	80	0.99	0.98	1.00	0.98	0.64	0.63	0.91	0.90	0.96	0.95	0.97	0.96
	100	1.00	0.97	1.00	0.99	0.68	0.66	0.94	0.92	0.98	0.97	0.99	0.98



(a) ADNI dataset b) Kaggle Dataset

Fig. 7. Test Accuracy Vs Epoch using the various optimizers.

The performance of the proposed model has been evaluated by computing the rate of accuracy, precision, and F1 score considering both the ADNI and Kaggle datasets using the formula stated below:

$$\text{Accuracy} = \frac{TP+TN}{TP+FP+FN+TN}$$

$$\text{Precision} = \frac{TP}{TP+FP}$$

$$\text{Recall} = \frac{TP}{TP+FN}$$

$$\text{F1 Score} = \frac{2 \times Precision \times Recall}{Precision+Recall}$$

The four entries in the confusion matrix are: TP = number of true positives, TN = number of true negatives, FP = number of false positives, and FN = number of false negatives. The confusion matrix for three and four classes is depicted in Fig. 8 (a) and (b) respectively:

Out of all optimizers, SGD performed the best. The classification results of both datasets using SGD as an optimizer are depicted in Table VII.

	Predicted Classes			
Actual Classes	Classes	A	B	C
	A	TN	FP	TN
	B	FN	TP	FN
	C	TN	FP	TN

a) Confusion Matrix for 3 classes

	Predicted Classes				
Actual Classes	Classes	A	B	C	D
	A	TN	FP	TN	TN
	B	FN	TP	FN	FN
	C	TN	FP	TN	TN
	D	TN	FP	TN	TN

b) Confusion Matrix for 4 classes

Fig. 8. Confusion Matrix for 3 and 4 classes.

TABLE VII. CLASSIFICATION RESULTS OF ADNI AND KAGGLE DATASET USING SGD AS OPTIMIZER

Dataset	Classes	TP	FN	FP	TN	ACCURACY	PRECISION	RECALL	F1 SCORE
ADNI	AD	53	7	2	232	0.97	0.96	0.88	0.92
	CN	78	7	17	192	0.92	0.82	0.92	0.86
	MCI	131	18	13	132	0.89	0.91	0.88	0.89
	Average					0.93	0.90	0.89	0.89
KAGGLE	MILD	153	0	1	0	1.00	0.99	1.00	0.99
	MODERATE	0	11	0	0	1.00	1.00	1.00	1.00
	NON-DEMENTED	0	0	550	1	0.99	1.00	0.99	0.99
	VERY MILD DEMENTED	0	0	3	383	0.99	0.99	1.00	0.99
	Average					0.99	0.99	1.00	0.99

We have also computed the execution time for training and testing for both datasets ADNI and Kaggle. From Table VIII, we can observe that the computation time of training for the ADNI dataset is highest for SGD with momentum optimizer and lowest for Adadelta. In case of computation time of the testing set, RMSprop took only 60 seconds whereas SGD took 1200 seconds. On the other hand, for the Kaggle dataset, the computation time of the training set is highest when the SGD with momentum optimizer was considered and lowest for the Adadelta optimizer. Similarly, for the testing set, Adagrad took only 14 seconds to execute whereas Adam and RMSprop took 120 seconds to execute, which is highest of all the optimizers. The experiments have been performed in Google Colab with a T4 graphical processing unit. All the optimizers used around 1180 MB of GPU memory from a total of 15,360 MB.

TABLE VIII TRAINING AND TESTING TIME FOR THE DIFFERENT OPTIMIZERS FOR THE TWO DATASETS AT 100 EPOCHS

Optimizers	Time	ADNI	Kaggle
SGD	Training time (sec)	10,800	34,800
	Testing time(sec)	1200	60
SGD with momentum 0.9	Training time (sec)	25,200	62,400
	Testing time(sec)	1980	40
Adagrad	Training time(sec)	10,300	37,200
	Testing time(sec)	60	14
Adadelta	Training time(sec)	9720	24,000
	Testing time(sec)	1440	80
Adam	Training time(sec)	10,060	36,000
	Testing time(sec)	65	120
RMSprop	Training time (sec)	9980	34,800
	Testing time (sec)	49	120

V. COMPARISON WITH OTHER MODELS

We have compared our proposed model with other state-of-the- art models. In Table IX comparisons have been made on the ADNI dataset and in Table IX comparisons have been made on the Kaggle dataset.

TABLE IX. COMPARISON OF PROPOSED MODEL WITH EARLIER MODELS

Dataset	Reference	Year	Model/Classifier	Sample size	Accuracy	Merits	Demerits
ADNI	R. A. Hazarika *et al.* [18]	2022	Improved DenseNet121	15,120	88%	Use of a set of different DL models with a very large dataset.	None of the DL model could achieve a good result
	K. AdityaShastry [28]	2024	CNN	1296	71.6%	Used the concept of multi-dataset as well as also multi classification	Have the potential of overfitting as well as also cannot be applied in real life scenario
	Proposed Model	2025	CNN	1712	90%	Used customized CNN taking into account the issues of overfitting and class imbalance.	Experiment done on public dataset and the cost of computational resources may be expensive
Kaggle	S. Murugan *et al.* [15]	2021	CNN	12,800	94%	SMOTE technique used to tackle the class imbalance problem	Considered uniform number of images in each of the directories
	S. Sharma *et al.* [16]	2022	VGG16	6400	90.4%	Applied feature extraction through the use of VGG16 model.	The model may tend to have the problem of overfitting.
	A. Balasundaram *et al.* [23]	2023	CNN	6400	94.45%	Image segmentation performed to isolate hippocampus region.	Image segmentation might lose some important information regarding the disease which might affect the accuracy of the model.
	K. AdityaShastry [28]	2024	CNN	6400	96.02%	Used the concept of multi-dataset as well as also multi classification	Have the potential of overfitting as well as also cannot be applied in real life scenario
	Proposed Model	2025	CNN	6400	99%	99% accuracy achieved without using any data pre-processing techniques. Class imbalance was tackled by using k-fold cross validation.	The model may be expensive in case of computational resources.

Merits: The merit of the following proposed model is that it can secure 90% as well as 99% in the datasets Kaggle and ADNI respectively without much data pre-processing. For the Kaggle dataset, the proposed model worked really well as we can see from the previous works that even using pre-trained model and applying different data pre-processing techniques, the model fails to achieve 99% accuracy. The proposed model doesn't use any pretrained model. Most of the earlier works have conducted binary classification which is less challenging. We have not considered a uniform number of images in each of the classes of the brain images rather we have used the concept of K-fold cross-validation to tackle the issue of class imbalance as well as overfitting. Another merit of this work is that it analyses all the state-of-the-art optimizers and it is being discussed in details with mathematical formulas. The performance of the ADNI dataset is 90%. It might not achieve accuracy above 90% like some previous works but one thing needs to be considered that every earlier work has used a subset of the ADNI dataset. As ADNI is a huge repository of brain MRI images, every researcher's dataset might differ from the other leading to different accuracies. We can also see that the number of images in each of the earlier works is different and each work has performed different sets of data pre-processing techniques. So, to justify which model is best for the classification of AD in case of the ADNI dataset it needs to be analysed further. This study is unique as we have not only analysed the optimizers but also examined the classsifier's performance at different epochs.

Demerits: Both the datasets Kaggle and ADNI datasets are public repository. So, the need of local dataset is utmost necessary for this kind of research work. But this needs collaboration of researchers, medical professionals and radiologist. The collaboration with medical institutions would let the researchers access the medical images, which would help them to create a real life dataset of a particular region. Because the ADNI, OASIS and Kaggle dataset consists of the brain images belonging to people from different countries. According to studies Alzheimer's disease is influenced by factors like lifestyle, food habits and genetics. So, these factors differ very much from region to region. Creating a real life dataset of the local region might benefit society in dealing with this disease and also help medical experts to diagnose it in proper time. The proposed model may be expensive in terms of computational resources.

VI. Conclusion

The challenges to achieving high classification accuracy with state-of-the-art CNN models require a critical update of parameters. In this paper, we have compared two datasets (ADNI and Kaggle) with different image formats (jpeg, bmp) and different numbers of images (6400 in Kaggle and 1700 in ADNI) with the same model. We have addressed the query that usually comes to mind that whether pre-processing is necessary when using deep learning. We have evaluated the proposed model, considering the ADNI dataset with pre-processing. Without pre-processing, it is observed that the results are inconsistent. After pre-processing using the FLIRT and BET algorithms, the model shows consistent results with high accuracy. In the case of the Kaggle dataset, no pre-processing has been done. A literature review reveals that for high classification accuracy, an equal number of images are essential in each class. In the present study, the problem of class imbalance has been overcome using the stratified K-fold technique to minimize the overfitting problem. The proposed model has been implemented using six different optimizers to reduce the convergence time. Further analysis of these optimizers will allow to analyse the nature of the dataset. We have achieved high accuracy using SGD with momentum and Adam optimizers. Thus, SGD with momentum and the Adam optimizer following the proposed CNN model could be the best optimizer for Alzheimer disease classification in particular and for medical image processing in general. The present work may be extended using different advanced machine learning techniques such as transfer learning, vision transformers, LSTM, and GAN to reduce convergence time and improve accuracy. The concept of ensemble learning can also be used in multiclass classification. As research on Alzheimer's disease heavily depends on the three public datasets ADNI, Kaggle, and OASIS, building a new real-life dataset can be challenging but would definitely benefit the community of researchers working on this domain. Super resolution techniques like SRCNN (Super Resolution Convolutional Neural Network, EDSR (Enhanced Deep Super-Resolution) and RCAN (Residual Channel Attention Network) can be used in the future for the enhancement of the classifier's performance.

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

Pallavi Saikia conducted the research and also wrote the paper. Sanjib Kumar Kalita analyzed the data and supervised the paper. All authors had approved the final version.

References

[1] R. Rajmohan and P. H. Reddy, "Amyloid-beta and phosphorylated tau accumulations cause abnormalities at the synapses of Alzheimer's disease neurons," *Journal of Alzheimer's Disease*, vol. 57, no. 4, pp. 975–999, 2017. https://doi.org/10.3233/JAD-160612

[2] J. Gaugler, B. James, T. Johnson *et al.*, "2022 Alzheimer's disease facts and figures," *Alzheimer's & Dementia*, vol. 18, no. 4, pp. 700–789, 2022. https://doi.org/10.1002/alz.12638

[3] K. B. Rajan, J. Weuve, L. L. Barnes *et al.*, "Population estimate of people with clinical AD and mild cognitive impairment in the United States (2020–2060)," *Alzheimer's & Dementia*, 2021. https://doi.org/10.1002/alz.12362

[4] K. S. Shaji, A. T. Jotheeswaran, and N. Girish, "The Dementia India Report—prevalence, impact, costs, and services for dementia: Executive summary," *Alzheimer's & Related Disorders Society of India*, 2010.

[5] P. Lodha, A. Talele, and K. Degaonkar, "Diagnosis of Alzheimer's disease using machine learning," in *Proc. 2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA)*, Aug. 2018, pp. 1–4.

[6] M. Shahbaz, S. Ali, A. Guergachi *et al.*, "Classification of Alzheimer's disease using machine learning techniques," *Data*, pp. 296–303, Jul. 2019.

[7] R. Alroobaea, S. Mechti, M. Haoues *et al.* (2021). Alzheimer's disease early detection using machine learning techniques. [Online]. Available: https://www.researchsquare.com/article/rs-624520/latest

[8] S. Kumar, I. Oh, S. Schindler *et al.*, "Machine learning for modeling the progression of Alzheimer disease dementia using clinical data: A systematic literature review," *JAMIA Open*, vol. 4, no. 3, p. ooab052, 2021.

[9] C. Kavitha, V. Mani, S. R. Srividhya *et al.*, "Early-stage Alzheimer's disease prediction using machine learning models," *Frontiers in Public Health*, vol. 10, 853294, 2022.

[10] R. Jain, N. Jain, A. Aggarwal *et al.*, "Convolutional neural network-based Alzheimer's disease classification from magnetic resonance brain images," *Cognitive Systems Research*, vol. 57, pp. 147–159, 2019. https://doi.org/10.1016/j.cogsys.2018.12.015

[11] M. Yildirim and A. Cinar, "Classification of Alzheimer's disease MRI images with CNN based hybrid method," *Ingénierie des Systèmesd'Information*, vol. 25, no. 4, pp. 413–418, 2020. https://doi.org/10.18280/isi.250402

[12] Y. N. Fu'adah, I. Wijayanto, N. K. C. Pratiwi *et al.*, "Automated classification of Alzheimer's disease based on MRI image processing using Convolutional Neural Network (CNN) with AlexNet architecture," *Journal of Physics: Conference Series*, vol. 1844, 012020, Mar. 2021.

[13] P. R. Buvaneswari and R. Gayathri, "Deep learning-based segmentation in classification of Alzheimer's disease," *Arabian Journal for Science and Engineering*, vol. 46, pp. 5373–5383, 2021.

[14] S. A. Ajagbe, K. A. Amuda, M. A. Oladipupo *et al.*, "Multi-classification of Alzheimer disease on Magnetic Resonance Images (MRI) using Deep Convolutional Neural Network (DCNN) approaches," *International Journal of Advanced Computer Research*, vol. 11, p. 53, 2021. http://dx.doi.org/10.19101/IJACR.2021.1152001

[15] S. Murugan, C. Venkatesan, M. G. Sumithra *et al.*, "DEMNET: A deep learning model for early diagnosis of Alzheimer diseases and Dementia from MR images," *IEEE Access*, vol. 9, 2021. https://doi.org/10.1109/ACCESS.2021.3090474

[16] S. Sharma, K. Guleria, S. Tiwari *et al.*, "A deep learning based convolutional neural network model with VGG16 feature extractor for the detection of Alzheimer Disease using MRI scans," *Measurement: Sensors*, vol. 24, 100506, 2022. https://doi.org/10.1016/j.measen.2022.100506

[17] J. V. Shanmugam, B. Duraisamy, B. C. Simon *et al.*, "Alzheimer's disease classification using pre-trained deep networks," *Biomedical Signal Processing and Control*, vol. 71, 103217, 2022. https://doi.org/10.1016/j.bspc.2021.103217

[18] R. A. Hazarika, D. Kandar, and A. K. Maji, "An experimental analysis of different deep learning-based models for Alzheimer's disease classification using brain magnetic resonance images," *Journal of King Saud University-Computer and Information Sciences*, vol. 34, no. 10, pp. 8576–8598, 2022. https://doi.org/10.1016/j.jksuci.2021.09.003

[19] E. G. Marwa, H. E. D. Moustafa, F. Khalifa *et al.*, "An MRI-based deep learning approach for accurate detection of Alzheimer's disease," *Alexandria Engineering Journal*, vol. 63, pp. 211–221, 2023. https://doi.org/10.1016/j.aej.2022.07.062

[20] G. Mohiud din dar, A. Bhagat, S. I. Ansarullah *et al.*, "A novel framework for classification of different Alzheimer's disease stages using CNN model," *Electronics*, vol. 12, no. 2, p. 469, 2023. https://doi.org/10.3390/electronics12020469

[21] N. Raza, A. Naseer, M. Tamoor *et al.*, "Alzheimer disease classification through transfer learning approach," *Diagnostics*, vol. 13, no. 4, p. 801, 2023. https://doi.org/10.3390/diagnostics13040801

[22] A. A. A. El-Latif, S. A. Chelloug, M. Alabdulhafith *et al.*, "Accurate detection of Alzheimer's disease using lightweight deep learning model on MRI data," *Diagnostics*, vol. 13, no. 7, p. 1216, 2023. https://doi.org/10.3390/diagnostics13071216

[23] A. Balasundaram, S. Srinivasan, A. Prasad *et al.*, "Hippocampus segmentation-based Alzheimer's disease diagnosis and classification of MRI images," *Arabian Journal for Science and Engineering*, vol. 48, pp. 10249–10265, 2023. https://doi.org/10.1007/s13369-022-07538-2

[24] Z. Ullah and M. Jamjoom, "A deep learning for Alzheimer's stages detection using brain images," *Computers, Materials & Continua*, vol. 74, no. 1, 2023. https://doi.org/10.32604/cmc.2023.032752

[25] N. S. Awarayi, F. Twum, J. B. Hayfron-Acquah *et al.*, "A neural architecture search CNN for Alzheimer's disease classification," *ECTI Transactions on Computer and Information Technology (ECTI-CIT)*, vol. 18, no. 4, pp. 543–554, 2024. https://doi.org/10.37936/ecti-cit.2024184.255728

[26] M. Assaduzzaman, M. Dutta, A. Saha *et al.*, "ALSA-3: Customized CNN model through ablation study for Alzheimer's disease classification," *Informatics in Medicine Unlocked*, vol. 50, 101584, 2024. https://doi.org/10.1016/j.imu.2024.101584

[27] A. Mehmood, F. Shahid, R. Khan *et al.*, "Utilizing siamese4D-AlzNet and transfer learning to identify stages of Alzheimer's disease," *Neuroscience*, vol. 545, pp. 69–85, 2024. https://doi.org/10.1016/j.neuroscience.2024.03.007

[28] K. A. Shastry, "Deep learning-based classification of Alzheimer's disease using MRI scans: A customized convolutional neural network approach," *SN Computer Science*, vol. 5, no. 7, p. 917, 2024. https://doi.org/10.1007/s42979-024-03284-4

[29] A. Singh and R. Kumar, "Brain MRI image analysis for Alzheimer's Disease (AD) prediction using deep learning approaches," *SN Computer Science*, vol. 5, no. 1, p. 160, 2024. https://doi.org/10.1007/s42979-023-02461-1

[30] H. Vega-Huerta, K. R. Pantoja-Pimentel, S. Y. Quintanilla-Jaimes *et al.*, "Classification of Alzheimer's disease based on deep learning using medical images," *International Journal of Online & Biomedical Engineering*, vol. 20, no. 9, 2024. https://doi.org/10.3991/ijoe.v20i10.49089

[31] L. Srividhya, V. Sowmya, R. Vinayakumar *et al.*, "Deep learning based for multi-stage diagnosis of AD," *Multimedia Tools and Applications*, vol. 83, no. 6, 2024. https://doi.org/10.1007/s11042-023-16026-0

[32] M. Z. Hussain, T. Shahzad, S. Mehmood *et al.*, "A fine-tuned convolutional neural network model for accurate Alzheimer's disease classification," *Scientific Reports*, vol. 15, 11616, 2025. https://doi.org/10.1038/s41598-025-86635-2

[33] V. Gondalia and K. Popat, "Advanced implementation of convolutional neural networks for Alzheimer's diseases diagnosis," *SN Computer Science*, vol. 6, no. 4, 2025. https://doi.org/10.1007/s42979-025-03851-3

[34] ADNI (Alzheimer's Disease Neuroimaging Initiative) (n.d.). [Online]. Available: https://adni.loni.usc.edu/about/

[35] Alzheimer MRI 4 classes dataset(n.d.). [Online]. Available: https://www.kaggle.com/datasets/marcopinamonti/alzheimer-mri-4-classes-dataset

[36] Y. Tang, C. Hojatkashani, I. D. Dinov *et al.*, "The construction of a Chinese MRI brain atlas: A morphometric comparison study between Chinese and Caucasian cohorts," *Neuroimage*, vol. 51, no. 1, pp. 33–41, 2010.

[37] J. Mazziotta, A. Toga, A. Evans *et al.*, "A probabilistic atlas and reference system for the human brain: International Consortium for Brain Mapping (ICBM)," *Philosophical Transactions of the Royal Society of London. Series B: Biological Sciences*, vol. 356, no. 1412, pp. 1293–1322, 2001.

[38] J. L. Lancaster, D. Tordesillas-Gutiérrez, M. Martinez *et al.*, "Bias between MNI and Talairach coordinates analyzed using the ICBM-152 brain template," *Human Brain Mapping*, vol. 28, no. 11, pp. 1194–1205, 2007.

[39] P. K. Mandal, R. Mahajan, and I. D. Dinov, "Structural brain atlases: Design, rationale, and applications in normal and pathological cohorts," *Journal of Alzheimer's Disease*, vol. 31, no. s3, pp. S169–S188, 2012.

[40] M. Jenkinson, P. Bannister, M. Brady, and S. Smith, "Improved optimization for the robust and accurate linear registration and motion correction of brain images," *Neuroimage*, vol. 17, no. 2, pp. 825–841, 2002. https://doi.org/10.1006/nimg.2002.1132

[41] S. M. Smith, "BET: Brain extraction tool," *Oxford Centre for Functional Magnetic Resonance Imaging of the Brain, Tech. Rep. FMRIB TR00SMS2b*, 2002.

[42] R. Khandelwal. (Feb. 2019). Overview of different optimizers for neural networks. [Online]. Available:https://medium.datadriveninvestor.com/overview-of-different-optimizers-for-neural-networks-e0ed119440c3

[43] D. A. Das. (Jul. 2020). Optimizers in deep learning. [Online]. Available:https://medium.com/analytics-vidhya/this-blog-post-aims-at-explaining-the-behavior-of-different-algorithms-for-optimizing-gradient-46159a97a8c1

[44] CampusX. (Jul. 2022). Optimizers in deep learning. [Online]. Available: https://www.youtube.com/watch?v=iCTTnQJn50E

[45] Z. Zhong, J. Li, Z. Luo *et al.*, "Spectral-spatial residual network for hyperspectral image classification: A 3D deep learning framework," *IEEE Transactions on Geoscience and Remote*

Sensing, vol. 56, no. 2, pp. 847–858, Feb. 2018. https://doi.org/10.1109/TGRS.2017.2755542