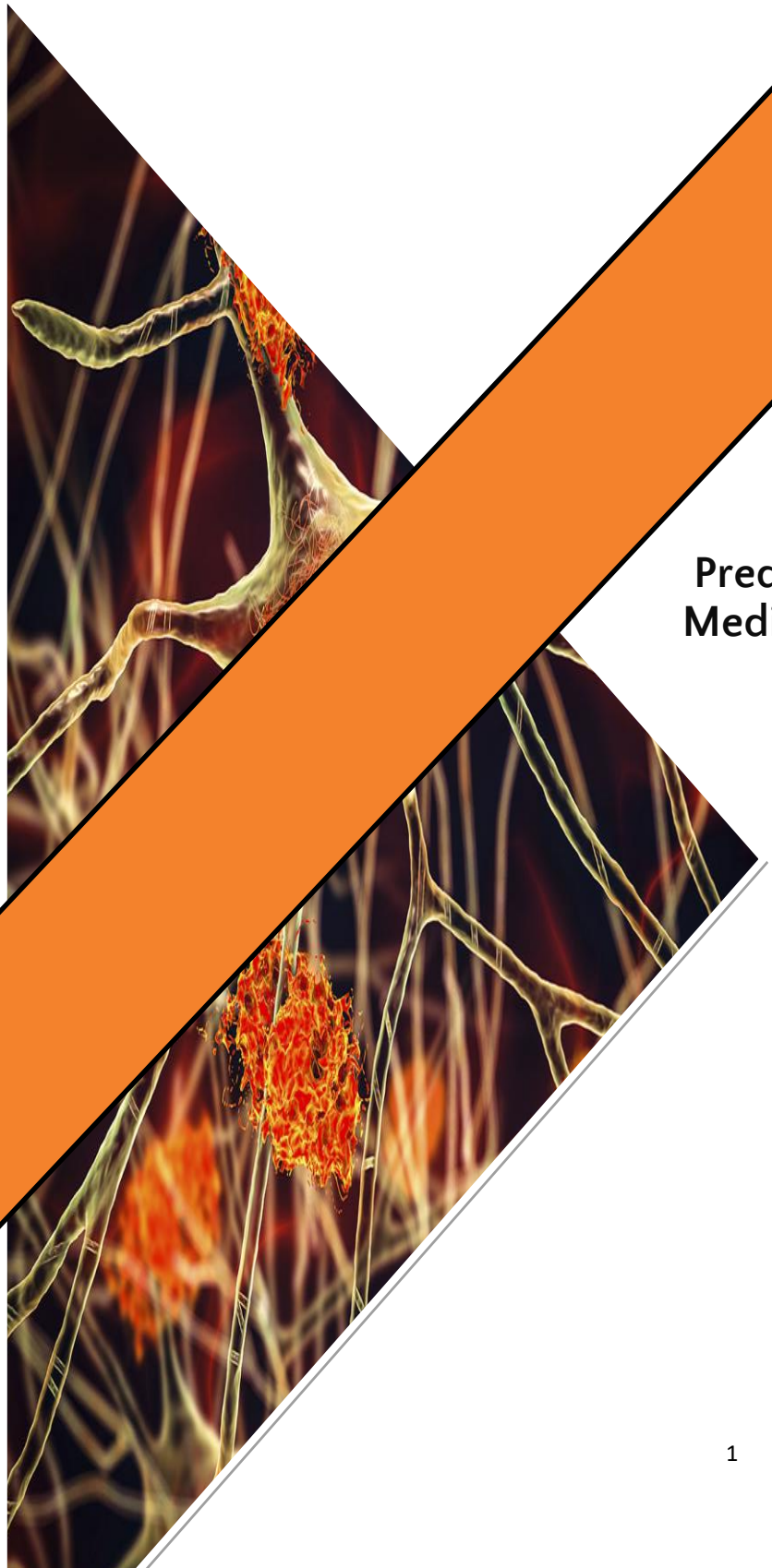
UNIVERSITY OF KENT

KENT BUSINESS SCHOOL

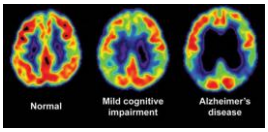
2022-2023



Predicting Alzheimer's Disease with Medical Imaging and Clinical Data: A Deep Learning Approach

NAME: SANOVAR

LOGIN: SS2648



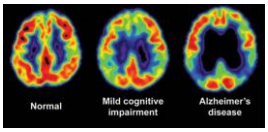
ACKNOWLEDGEMENT

I would like to sincerely thank and appreciate my supervisor, Dr. Sheema Noorain, for her tremendous advice, assistance, and encouragement throughout the completion of my dissertation. Her knowledge and perceptive criticism were extremely helpful in determining the course of our study, and they motivated me to learn more about Alzheimer's disease prognosis. Her unfailing confidence in my abilities has served as a constant source of inspiration, and I am incredibly appreciative of the chance to study under her tutelage.

I also want to express my sincere gratitude to my director, Zhen Zhu, for his unwavering support and guidance. His knowledge of deep neural networks and machine learning has been essential to the development and improvement of my research methods. His openness to impart his expertise and offer insightful commentary has enhanced our study and made it a satisfying experience.

I also appreciate the University of Kent for providing the tools and space needed to carry out this study. I am grateful for the opportunity to be exposed to a diverse and engaging learning community at the institution, where the academic environment has been supportive of learning and the exploration of novel concepts.

I would want to express my sincere gratitude to everyone who helped finish this dissertation. Your help has been tremendous, and I am very appreciative of the chance to conduct this study and advance the field of Alzheimer's disease prediction.



ABSTRACT

Alzheimer's disease (AD) is a prevalent and debilitating neurological disorder that must be appropriately detected in early stages. Neuroimaging procedures such as positron emission tomography (PET) and magnetic resonance imaging (MRI) scans can be difficult to interpret, despite their importance in identifying the physical changes in the brain associated with Alzheimer's disease. Convolutional Neural Networks (CNNs) in particular have shown promising results in automating and enhancing the processing of such medical images, hence assisting in the early diagnosis of disease.

In this dissertation, an entirely novel CNN model for detecting AD from MRI images is presented. The model was trained using data augmentation to solve the inherent problem of unequal and insufficient data that is common in medical imaging. To further enhance model interpretability, Layer-wise Relevance Propagation (LRP) was employed. LRP reveals details on the regions of the brain that the model prioritises for making predictions.

With a classification accuracy of 96% on the ADNI dataset, the suggested model greatly surpassed previously reported strategies. The CNN model created in conjunction with LRP for interpretability provides a robust and dependable strategy for aiding in the early identification of Alzheimer's disease, which has the potential to have a significant impact on clinical practice.

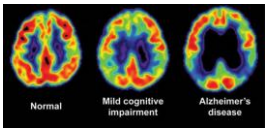
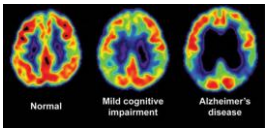
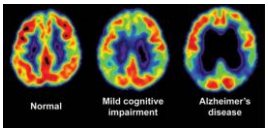
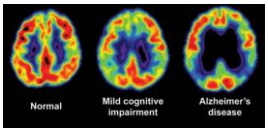


TABLE OF CONTENTS

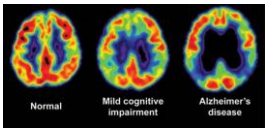




LIST OF FIGURES

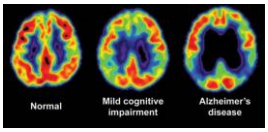


LIST OF TABLES



LIST OF ABBREVIATIONS

ML	MACHINE LEARNING
AD	ALZHIMER'S DISEASE
EMCI	EARLY MILD COGNITIVE IMPAIRMENT
MCI	MILD COGNITIVE IMPAIRMENT
CNN	CONVOLUTIONAL NEURAL NETWORK
AUC	AREA UNDER CURVE
ADNI	ALZHIEMER'S DISEASE NUEROIMAGING INITIATIVE
MRI	MAGNETIC RESONANCE IMAGING
ROC	RECIEVER OPERATING CHARACTERISTICS
SVM	SUPPORT VECTOR MACHINE



1.INTRODUCTION

Alzheimer's disease (AD) is a crippling neurological disease that affects millions of people worldwide. Its development may be gradual, but the subsequent brain cell destruction and cognitive deterioration can be severe. Given the estimated global prevalence of dementia occurring every three seconds, there is an urgent need to create breakthrough technology to combat Alzheimer's disease and other related disorders. Alzheimer's disease is predicted to affect 1.2% of the world's population by 2046, which is a large and alarming percentage (Liu, et al., 2014).

The use of machine learning and deep neural networks, which can analyse massive datasets and identify patterns in a fraction of the time previously necessary, is one promising area for research. In particular, the creation of deep learning models for radiological image analysis has established itself as a standard practise. Researchers and medical practitioners can more correctly detect and forecast the development of AD by utilising these cutting-edge methodologies, which may result in early interventions and better patient outcomes (Jadhav, 2019). The suggested method generates high-resolution disease probability maps from local brain anatomy and feeds them into a multilayer perceptron. To forecast Alzheimer's, we are employing convolutional neural networks. The model takes into account the four phases of dementia and offers a focused diagnostic to deliver precise and understandable visualisations of a person's risk for Alzheimer's disease (Murugan, et al., 2021).

1.1 OBJECTIVES

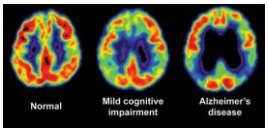
The major objective of this dissertation is to develop and evaluate a convolutional neural network (CNN) model for predicting the onset of Alzheimer's disease using data from medical imaging. The proposed model aims to achieve high diagnostic accuracy for AD, enabling early detection and intervention. By giving crucial information about the brain regions that greatly affect the CNN model's predictions, this study also intends to make the CNN model easier to understand. The ultimate goal is to assist in the creation of AD prediction tools, which could improve patient outcomes and aid in our understanding of the disease's underlying causes.

1.2 RESEARCH METHODOLOGY

The suggested method represents a significant advancement towards a more accurate, discernible, and intelligible diagnosis of Alzheimer's disease. The technique used is deep learning, specifically convolutional neural network. We will be considering how different architecture of CNN layers change the outcome. We will be doing various data preprocessing steps to make MRI scans suitable for image recognition. This study contributes to the growing corpus of research on the use of machine learning to address the challenges of AD diagnosis through rigorous development, testing, and performance assessment. By using the strength of deep learning and interpretability techniques, this research has the potential to change the early diagnosis and management of Alzheimer's Disease, ultimately improving the lives of millions of individuals affected by this life-altering condition.

1.3 READERS MAP

This thorough report covers many important aspects of diagnosing Alzheimer's disease (AD), beginning with an insightful introduction, followed by a thorough literature review, clarifying the aims and objectives, outlining the methodology, showcasing the noteworthy findings, engaging in thoughtful discussions, and closing with a strong conclusion. We aim to shed light on the potential of cutting-edge deep learning algorithms in enhancing the accuracy and early identification of AD utilising medical imaging data through this methodical exploration.



2.THEMATIC LITERATURE REVIEW

The identification and prediction of Alzheimer's Disease (AD) has showed tremendous promise when using cutting-edge imaging and deep learning approaches. These techniques use information from medical imaging tests, such MRI and PET scans, to create precise diagnostic tools. We examine numerous studies that investigate the possibility of deep learning models for early AD diagnosis in this overview of the literature. Convolutional neural networks (CNNs) and recurrent models are used in the studies to extract useful characteristics from medical images, improving classification accuracy. To fully realise the potential of these methodologies, however, issues including restricted dataset availability, class imbalance, and model interpretability must be resolved.

The structure and scope of this literature review are designed to help readers navigate through the various themes and findings. The thematic analysis will cover the following themes: (1) imaging modalities for AD detection, (2) deep learning techniques for AD prediction, (3) Comparison of deep learning models with traditional machine learning method, (4) Interpretability of deep learning models for AD prediction, and (5) challenges and limitations of deep learning techniques for AD prediction. The tabular taxonomy will provide a summary of the different studies, including the authors, year of publication, dataset used, deep learning architecture, performance metrics, and other relevant details.

2.1 IMAGING MODALITIES FOR AD DETECTION

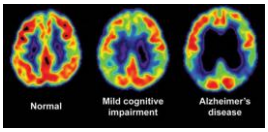
The study (Liu, et al., 2014) used Magnetic Resonance Imaging (MRI) images from 311 participants taken from the Alzheimer's Disease Neuroimaging Initiative (ADNI) database. These images were segmented into 83 functional regions, and the Cerebral Metabolic Rate of Glucose (CMRGlc) patterns from Positron Emission Tomography (PET) scans were also extracted. After selecting features using Elastic Net, the data was classified. The study (Francisco J. Martinez-Murcia, 2020) used Normalized (Norm) MRI images that were spatially normalized to the standard TPM template using SPM12 software. Furthermore, in this paper (Basher et al., 2021), pre-processing entailed down sampling the input pictures and performing intensity normalization using the integral normalization method.

The authors have suggested an automated approach for diagnosing Alzheimer's Disease by using MRI data from the Gwangju Alzheimer's and Related Dementia (GARD) dataset, based on deep learning techniques. The authors (Li, et al., 2019) employed ResNet-50, a convolutional neural network (CNN) model that has already been trained, to extract features from brain MRI data. I'll make use of a set of data from the LONI Image and Data Archive (IDA), which consists of 15170 T2-weighted axial MRI scans. To avoid overfitting, the photographs will be transformed to.jpeg and enhanced utilizing tricks like rotation, flipping, and scaling. Following the literature reviews, the dataset will be balanced using SMOTE (Synthetic Minority Oversampling Technique) to address class imbalance.

2.2 DEEP LEARNING TECHNIQUES FOR AD PREDICTION

The proposed method in the study (Chima S Eke, 2021) identifies AD at an early stage with high accuracy by using support vector machines (SVMs) for classification. The study (Liu, et al., 2014) used Stacking sparse auto-encoders and a SoftMax regression layer are used in a deep learning architecture to suggest a novel approach for the early diagnosis of Alzheimer's disease. This strategy, which is semi-supervised, lessens the reliance on past data knowledge. Different deep learning techniques (Francisco J. Martinez-Murcia, 2020), the data-driven features that were derived using this CAE architecture were highly correlated with other clinical and NST variables.

The paper (Shui-Hua Wang, 2018) suggested an eight-layer Convolutional Neural Network (CNN) with Max Pooling, Leaky Rectified Linear Unit (LeakyReLU), and AD classification. This paper concluded that the use of L-Relu with CNN gives the best performance when compared with different techniques used. The study by (Qian & Jun Zhou, 2018) examines the effects of developing widely-used activation functions and optimising the performance of neural networks with deep architecture. This study uses a variety of neural network activation functions, including, Sigmoid function 1, Second Hyperbolic Tangent Relu layer three and its



development. It was concluded that Relu layer outperforms CNN because it fixes the vanishing gradient issue.

The paper (Afzal, et al., 2019) compared input layers, convolutional layers, pooling layers, and fully connected layers using a pre-trained Alex-Net model. They discovered that comprehensive augmentation techniques can stop overfitting problems, which are a significant problem in a class-balanced dataset. This paper (Basher, et al., 2021) combines a deep neural network (DNN) model with a convolutional neural network (CNN) model to suggest a unique way for diagnosing Alzheimer's disease. Using a two-stage ensemble Hough-CNN, the left and right hippocampi are automatically located in the method. Next, 3-D patches are extracted and 2-D slices are divided alongaxial, sagittal, and coronal views.

The classification network is then trained and tested using the volumetric features that were extracted from each slice using a discrete volume estimation CNN model (DVE-CNN). Various CNN-based topologies exist from AD prediction, including ResNet, RCNN, GoogLeNet, and U-Net. The common element in these topologies is the encoder unit, which generates features using convolution, normalization, activation, and pooling (Bijen Khagi, 2020). For the proposed AD prediction job, I will employ a deep convolutional neural network (CNN) architecture.

CNNs are renowned for their accomplishments in picture segmentation and classification tasks. Multiple convolutional layers, max-pooling layers, batch normalisation layers, dropout layers, a flatten layer, and dense layers will all be included in the model. The Rectified Linear Unit (ReLU) activation function will be employed to optimise the training process. The findings from the literature review will be used to analyse and classify CNN's architecture.

2.3 COMPARISON OF DEEP LEARNING MODELS WITH TRADITIONAL MACHINE LEARNING METHOD

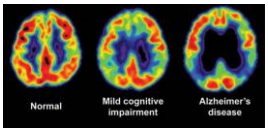
Multiple levels of abstraction and representation are used in deep learning (DL) to analyse complicated data, including text, voice, and visual data. DL can be roughly divided into generative and discriminative architectures. Recurrent neural networks (RNN), deep auto-encoders (DAE), deep boltzmann machines (DBM), and deep belief networks (DBN) are all examples of generative architecture.

Convolutional Neural Networks (CNN) and RNN are examples of discriminative architecture (Al-Shoukry, et al., 2020). In comparison to SVMs with deep learning mode, a performance improvement has been made in terms of overall accuracy and overall specificity (47.42% and 83.75%) (Liu, et al., 2014). The authors in journal (Francisco J. Martinez-Murcia, 2020) said when Support Vector Machines (SVMs) and Multi-Layer Perceptron (MLP) models are compared for classification accuracy, the Convolutional Autoencoder (CAE) model has an accuracy of roughly 80%.

The authors (Hong, et al., 2019) evaluated the accuracy of their LSTM model to more conventional machine learning techniques, attaining a score of 88.75%. Another author says (Shui-Hua Wang, 2018) Three activation functions (AFs): sigmoid, rectified linear unit (ReLU), and leaky ReLU. Additionally, the average pooling, maximum pooling, and stochastic pooling were evaluated. Leaky ReLU with max pooling produced the best performance results, according to the numerical experiments. It achieved 97.96% sensitivity, 97.35% specificity, and 97.65% accuracy.

In the next paper (Jadhav, 2019) Three different categorization methods are compared in terms of performance: creating a capsule network from scratch; learning from transfer examples using the VGG16 and InceptionV3; and using an SVM classifier with oriented fast and rotated binary robust independent elementary features (ORB). The accuracy of the proposed model in (Afzal, et al., 2019) using a single brain MRI view has a success rate of 98.41%, whereas 3D views have a 95.11% success rate. For phases of Alzheimer's disease, the suggested system fared better than the methods now in use.

The proposed method of two hough CNN model (Basher, et al., 2021) achieved excellent area under the curve (AUC) values of 92.54% and 90.62%, as well as high classification accuracies of 94.82% and 94.02% for the left and right hippocampi, respectively. On the same dataset, the approach performed better than other methods by a specified margin. In the research (Li, et al., 2019), manually created features were extracted from the MRI data using gray-level co-occurrence matrix (GLCM) and gray-level run-length matrix (GLRLM) approaches. The performance of the proposed knowledge transfer strategy was compared with a conventional machine



learning method. An SVM classifier was then used to diagnose using these features. In terms of accuracy, sensitivity, specificity, and area under the curve (AUC) metrics, the knowledge transfer strategy fared better than the conventional machine learning method.

2.4 INTERPRETABILITY OF DEEP LEARNING MODELS FOR AD PREDICTION

The authors- in journal (Francisco J. Martinez-Murcia, 2020) used t-SNE visualization technique to show that the representations learned by their CAE model could separate the different stages of AD progression, giving insight into the disease's underlying manifold structure. The authors (Al-Shoukry, et al., 2020) mention the use of visualization techniques, such as saliency maps and activation maps, to understand the important features learned by deep learning models for AD prediction. The paper (Al-Shoukry, et al., 2020) proposes several techniques such as Layer-wise Relevance Propagation (LRP) and Integrated Gradients have been proposed to interpret deep learning models for medical diagnosis.

The authors in (Shui-Hua Wang, 2018) used a visualization technique called Class Activation Mapping (CAM) to highlight the important regions in the MRI images that contributed to the classification results. The authors (Afzal, et al., 2019) used a technique called Grad-CAM (Gradient- weighted Class Activation Mapping) to interpret the deep learning model's predictions. Grad- CAM highlights the important regions in the input image that contributed to the model's decision, thus providing interpretability. The evaluation of the proposed model's performance will be done using a heatmap and confusion matrix. These tools are essential for comparing variables and displaying accuracy, precision, and recall. The heatmap and confusion matrix will allow for a comprehensive assessment of the model's predictive power, aiding in the identification of potential areas of improvement.

2.5 CHALLENGES AND LIMITATIONS OF DEEP LEARNING TECHNIQUES FOR AD PREDICTION

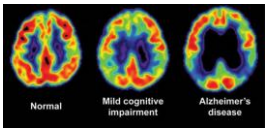
The machine learning algorithms like SVM is computationally complex for multiple classification of images (Francisco J. Martinez-Murcia, 2020). Overfitting is the biggest risk while using LSTM as there is no dimensional reduction of images. LSTM performs accurately for NLP rather than Image Processing models (Al-Shoukry, et al., 2020). In order for a CNN-based system to be used in actual clinical applications, visualisation must be incorporated to help with the interpretation and understanding of the results of the system (Jadhav , 2019). ‘Overfitting’ and ‘generalization’ problems are the biggest challenges for deep learning models (Bijen Khagi, 2020). The paper (Li, et al., 2019) identified several challenges and limitations of using deep learning techniques for AD prediction, especially on small datasets. These include overfitting, limited data availability, class imbalance, and difficulty in interpreting the learned features.

2.6 SUMMARY AND RECOMMENDATIONS

The literature review highlights the possibility of applying deep learning algorithms for the identification and prognosis of Alzheimer's disease (AD) using medical imaging data. The need for more research to improve the performance of deep learning models in AD detection and prediction is highlighted by challenges such as restricted access to diverse and vast datasets, as well as problems such class imbalance and overfitting.

Future research should give special consideration to the creation of multi-modal models that incorporate several categories of medical imaging data in order to improve the accuracy of AD diagnosis and prognosis. Collaborations between research institutes and healthcare facilities can be crucial for increasing the variety and size of datasets that are accessible.

My own research will use a deep convolutional neural network (CNN) architecture to try to increase the precision and effectiveness of AD diagnosis and prediction. This architecture will include several layers designed specifically for the goal of forecasting various phases of Alzheimer's. We can aim to improve the precision of AD diagnosis and prediction by utilising the potential of deep learning techniques, such as CNNs. The optimisation of these approaches and facilitation of their smooth integration into clinical practise will depend heavily on ongoing research efforts and beneficial collaborations.



3.AIMS, OBJECTIVES AND HYPOTHESIS

3.1 AIM:

The purpose of this study is to construct and assess CNN models for accurate Alzheimer's Disease (AD) onset prediction utilising medical imaging data. In order to improve early detection and diagnosis, the study compares the effectiveness of CNN models with conventional machine learning techniques and human experts in predicting AD.

3.2 OBJECTIVES:

CNNs, a type of deep learning model, can accurately predict the onset of AD by analysing medical imaging and clinical data, outperforming traditional machine learning methods and human experts.

1) How accurately can a CNN model predict the onset of AD using medical imaging data?
2) How does the architecture of a CNN model impact its performance in predicting Alzheimer's disease and how do different hyperparameters affect the model's performance?

3.3 HYPOTHESIS:

By developing specific hypotheses that direct the route of exploration, this study lays the framework for revolutionising Alzheimer's Disease diagnosis and intervention. The research seeks to shed light on the potential of cutting-edge computational techniques, particularly Convolutional Neural Networks (CNNs), in changing our comprehension of the disease's early phases. These assumptions capture the predicted effect of utilising cutting-edge technology, encouraging accurate prediction and meaningful interpretation, thus starting the path towards more efficient diagnostic methods.

HYPOTHESIS:

HYPOTHESIS 1: Compared to traditional machine learning methods and human experts, the CNN model will demonstrate a considerably higher accuracy in predicting the onset of Alzheimer's disease using medical imaging data.

HYPOTHESIS 2: The CNN model's architecture will affect how well it performs in making predictions about Alzheimer's disease, with more complex structures showing increased accuracy since they can better recognise subtle patterns in medical imaging data.

HYPOTHESIS 3: The CNN model's predictive performance will be impacted by several hyperparameters, with optimised hyperparameter settings producing a higher accuracy in forecasting the onset of Alzheimer's disease.

4.TABULAR TAXONOMY

Authors	Study Title	Year of publication	Data Sources	Methods	Preprocessing techniques	Performance	Interpretability	Comparison	Limitations
Siqi Liu; Sidong Liu; WeidongCai; Sonia Pujol; Ron Kikinis; Dagan Feng	Early Diagnosis Of Alzheimer's Disease With Deep Learning	2014	MRI (ADNI)	Stacked Auto-Encoders and a Softmax Regression Layer.	Elastic Net	accuracy 87.76%	variance map	Single-Kernel and Multi-Kernel SVM model.	t multi-classification.
Francisco J. Martinez-Murcia , Andres Ortiz , Juan-Manuel Gorriz , Javier Ramirez , and Diego Castillo-Barnes	Studying the Manifold Structure of Alzheimer's Disease: A Deep Learning Approach Using Convolution Autoencoders	2020	MRI	Convolutional Autoencoders (CAEs)	standard TPM template using SPM12 software	accuracy 80%	T-SNE maps	VSM, PCA,MLP	Need for large amounts of data
Suhad Al-Shoukry; Taha H. Rassem; Nasrin M. Makbol	Alzheimer's Diseases Detection by Using Deep Learning Algorithms: A Mini-Review	2020	MRI (ADNI)	RNN, CNN,Caps-Net	image re-orientation, cropping, skull stripping, image normalisation	accuracy 75%	saliency maps and activation maps	SVMs and Random Forests	A lot of inconsistencies like time required for routing.
Xin Hong; Rongjie Lin; ChenhuiYang; Nianyin Zeng; Chunting Cai; Jin Gou	Predicting Alzheimer's Disease Using LSTM	2019	MRI, PET,DTI	LTMS	skull strip, registration, segmentation, normalization, and smoothing	accuracy 88.75%.	ayer-wise Relevance Propagation (LRP) and Integrated Gradients	SVM vs RF-DNN vs CNN PCA	Overfitting is the biggest risk while using LSTM
Shui-Hua Wang, Preetha Phillips, Yuxiu Sui, Bin Liu, MingYang & Hong Cheng	Classification of Alzheimer's Disease Based on Eight-Layer Convolutional Neural Network with Leaky Rectified Linear Unit and Max Pooling	2018	MRI(OASIS)	CNN with RELU layer and Maxpooling layer	BET(Brain Extraction Tool), FLIRT and FNIRT were used for spatial normalization	sensitivity 97.96%, specificity 97.35%, accuracy 97.65%.	Class Activation Mapping	Random Forest, SVM Logistic Regression	backword propagation due to Vanishing Gradient Problem.
Samir S. Yadav & Shivajirao M. Jadhav	Deep convolutional neural network based medical image classification fordisease diagnosis	2019	OCT	VGG16 and InceptionV3, anda capsule network training	Data augmentation	accuracy 92.75%.	Feature map	SVM	Overfitting problem
Bin Ding, Nanjing, HuiminQian; Jun Zhou	Activation functions and their characteristics in deep neural networks	2018	NONE	ReLU and sigmoid activation functions, and the exponential linear unit (ELU)	NONE	NONE	NONE	CNN	Dying ReLU problem
Sitara Afzal; Muazzam Maqsood; Faria Nazir; UmairKhan; Farhan Aadil; Khalid	A Data Augmentation-Based Framework to Handle Class Imbalance Problem for Alzheimer's Stage Detection	2019	MRI	CNN	Data augmentation	accuracy 98.41%	Epochs	CNN	Negative transfer.
Abol Basher; Byeong C. Kim; Kun Ho Lee; Ho Yub Jung	Volumetric Feature-Based Alzheimer's Disease Diagnosis From sMRI Data Using a Convolutional Neural Network and a Deep Neural Network	2021	MRI(GARD)	Hough CNN	Patch generation	accuracy 94.82% , 94.02%, AUC 92.54% ,90.62%	Hough	ML algorithms	Relies on the accuracy of the automatic localization
Bijen Khagi,Goo-Rak Kwon	3D CNN Design for the Classification of Alzheimer's Disease Using Brain MRI and PET	2020	MRI, PET	CNN(DIVNET)	Data augmentation	accuracy 94.5	T-SNE maps	CNN	This trained CNN is not deep enough to prototype a human brain structure.
Suriya Murugan; Chandran Venkatesan; M. G.(base paper) Sumithra; Xiao-Zhi Gao; B. Elak	DEMNET: A Deep Learning Model for Early Diagnosis of Alzheimer Diseases and Dementia From MR Images	2021	MRI,PET(KAGGLE)	CNN	SMOTE	Accuracy 95.23%, Area Under Curve 97% Cohen's Kappa 0.93	AUC, Confusion matrix	NONE	Class imbalance
Wei Li; Yifei Zhao; Xi Chen; YangXiao; Yuanyuan Qin	Detecting Alzheimer's Disease on Small Dataset: A Knowledge Transfer Perspective	2019	MRI	CNN	Normalization	sensitivity 92%, specificity 79%, accuracy 84.6%.	Matlab	NONE	generalizability due to small dataset

TABLE NO. 1: TABULAR TAXONOMY

5.METHODOLOGY

This section describes the methodology and research plan used to examine the use of deep learning to forecast Alzheimer's disease.

5.1 RESEARCH ONION

We use the Research Onion model, which consists of numerous layers that direct the research process, each building upon the preceding one, to give an organised framework as shown below:

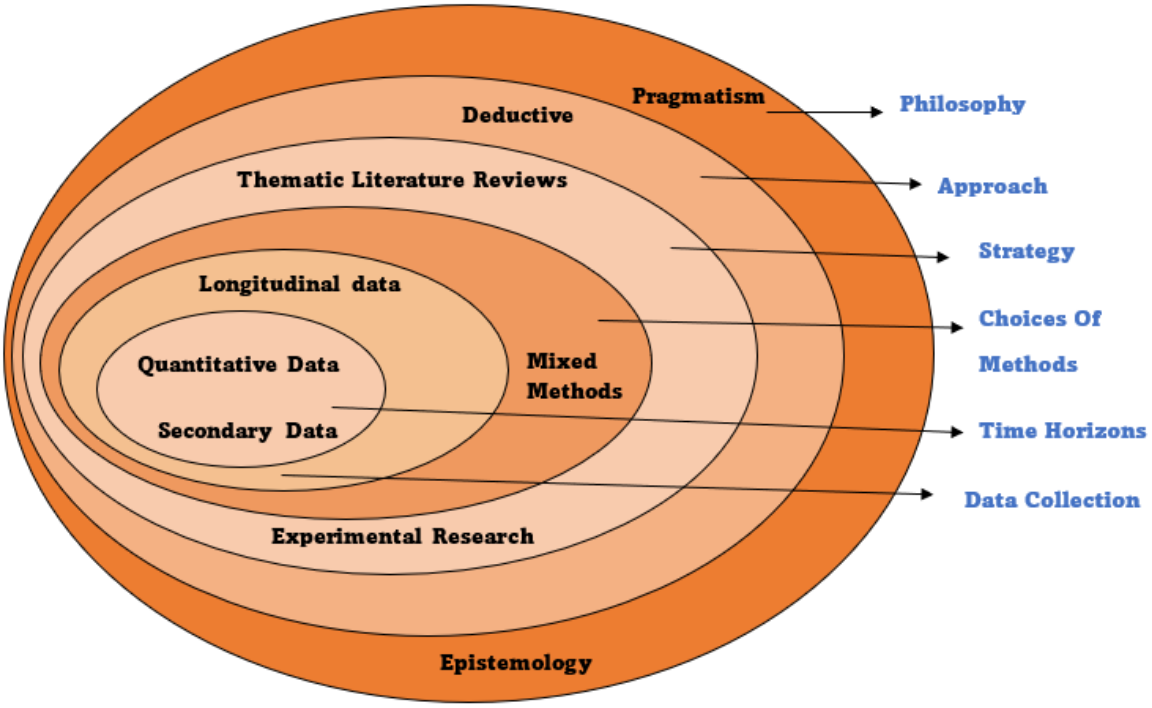


Figure 1: Research Onion (Mitchell, October 2018)

5.1.1 PHILOSOPHICAL ASSUMPTIONS

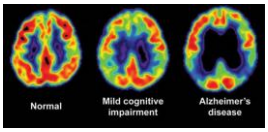
We start with the base layer of our research—the philosophical presumptions—which form the foundation of our discussion. We have opted for pragmatism, which acknowledges the importance of combining quantitative and qualitative research methodologies to develop a thorough grasp of Alzheimer's disease prediction. With pragmatism, we may concentrate on workable solutions and choose from a variety of approaches that support our study objectives.

5.1.2 RESEARCH APPROACH

We take a deductive approach, building on our philosophical viewpoint. With this strategy, specific hypotheses drawn from theories or models in use are put to the test. It enables us to investigate how well our suggested deep learning model fits with empirical information and current understanding of Alzheimer's disease.

5.1.3 RESEARCH STRATEGY

We incorporate several research tactics into the Research Onion model:



5.1.3.1 THEMATIC LITERATURE REVIEW

We carry out a theme literature review as a component of our mixed methodologies strategy. In order to pinpoint important themes, trends, and gaps regarding alzheimer's disease prediction, this technique entails systematically analysing the body of existing literature. The lessons learned from this review guide our work and put our experimental results in their proper context.

5.1.3.2 EXPERIMENTAL RESEARCH

To examine the effectiveness of our deep learning model, we conduct experimental research. We may evaluate the precision and accuracy of the model in predicting the phases of Alzheimer's disease by adjusting variables and using controlled experiments.

5.1.4 CHOICES OF METHODS

We integrate quantitative and qualitative data gathering and analysis methodologies using mixed methods. We can triangulate results using this multi-layered technique, which gives us a thorough grasp of Alzheimer's disease prediction. Our approach makes use of both organised numerical data and detailed contextual knowledge.

5.1.5 TIME HORIZONS

We choose a longitudinal time horizon in order to account for the temporal component of our research. With this method, changes in the stages of Alzheimer's disease are seen and tracked over a lengthy period of time. We may evaluate the model's predicting skills over time by looking at the disease's course.

5.1.6 DATA COLLECTION

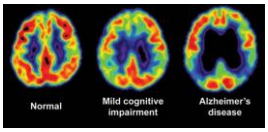
We use a number of techniques to collect our data:

5.1.6.1 QUANTATIVE DATA COLLECTION

In order to train and test the deep learning model, we collect quantitative data using medical imaging, including brain pictures. The numerical foundation for our prediction study is provided by these data.

5.1.6.2 SECONDARY DATA COLLECTION

In addition, we use secondary data sources, such as pre-existing datasets and academic publications, to deepen our comprehension of Alzheimer's disease and to supplement our primary data collection efforts.



5.2 MODEL ARCHITECTURE

In the proposed study, we employ the strength of convolutional neural networks (CNNs) as a reliable method to extract discriminant information for the categorization of Alzheimer's disease (AD). The methodology includes numerous crucial processes, including dataset collecting, pre-processing, data training and testing, as well as thorough evaluation of the outcomes.

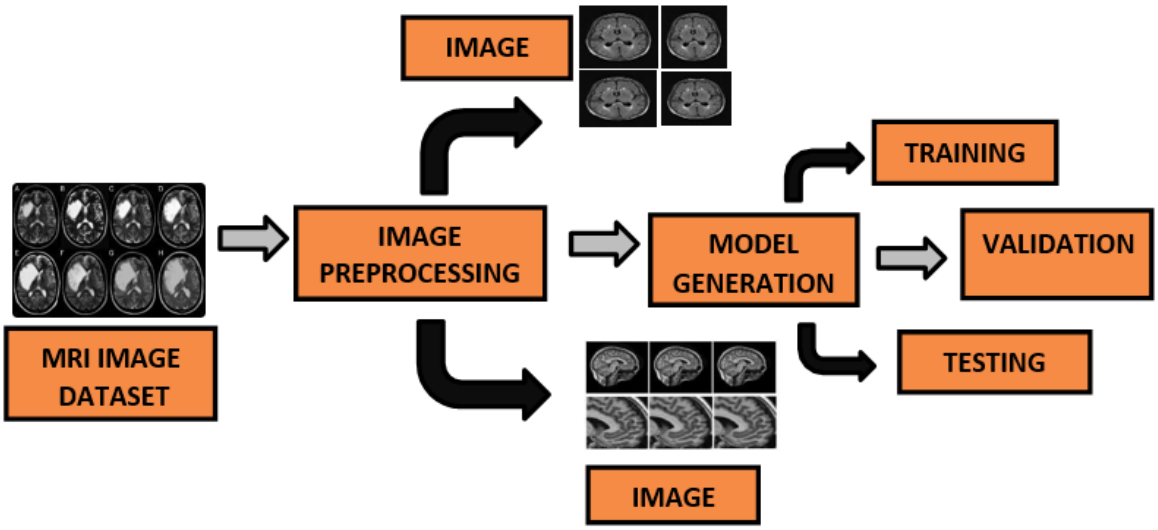


Figure 2:PROPOSED MODEL (Murugan, et al., 2021)

This is the enhanced model which we are using in our dissertation to detect AD. This has been modified using the above flowchart. The workflow of our model is described below:

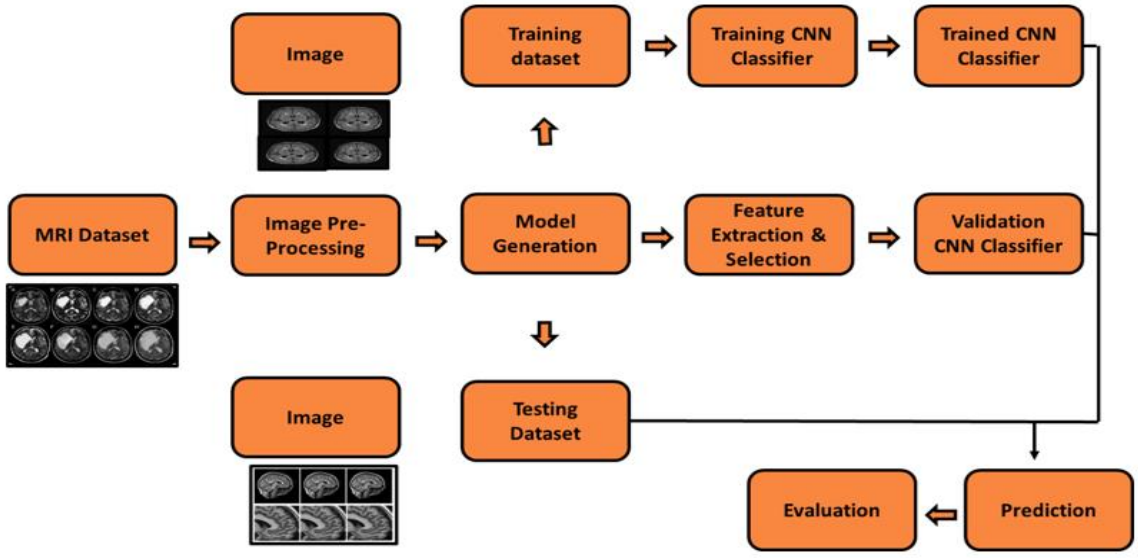
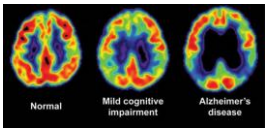


Figure 3: ENHANCED MODEL (Murugan, et al., 2021)



5.3 MRI DATASET

The dataset used in this study is the dataset obtained from the LONI Image and Data Archive (IDA). It comprises of 15170 MRI images categorized into four classes and is accessed through an authenticated user ID and password at https://ida.loni.usc.edu. The images are in T2-weighted axial MRI format and stored in. dcm format, which are then converted to .jpeg. The RAR file containing the MRI images for this investigation has been successfully unzipped in Google Colab using the right code. The images in the dataset have a size of 256x256.

The dataset utilised is the "Alzheimer's Dataset - 4 Class of Images," which includes images reflecting the Mild Demented, Moderate Demented, Non-Demented, and Very Mild Demented stages of Alzheimer's disease. This has been shown below.

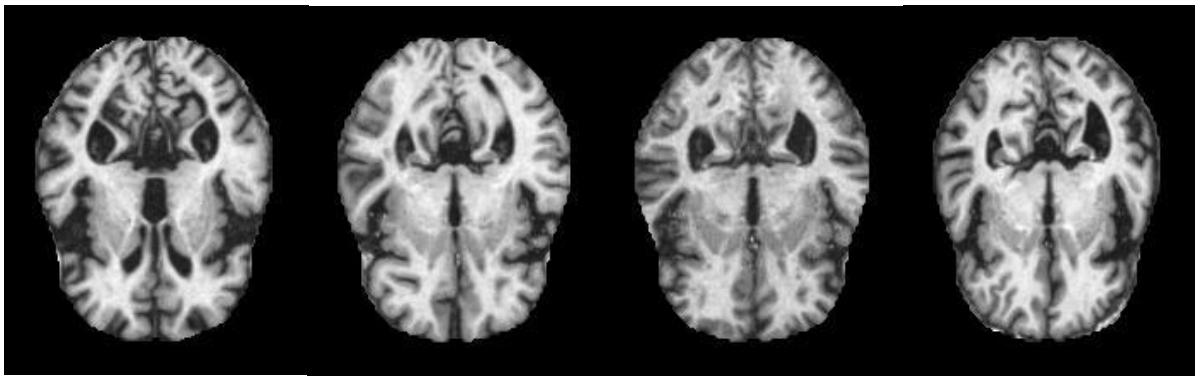


Figure 4:MRI DATASET (USC, n.d.)

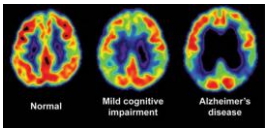
It is critical to address batch effects caused by variances in data collecting and processing across many sources if we are to assure the accuracy and generalizability of our deep learning-based model for MRI imaging. Information Harmonisation, which uses data normalisation and quality control procedures to reduce undesirable batch effects, and Space Transformation, which uses deep learning techniques built to handle batch effects for reliable and consistent outcomes, are the two key strategies that we will assess.

We seek to create an accurate and clinically applicable AI model for brain MRI imaging, advancing medical diagnosis and treatment for Alzheimer's disease by implementing these approaches and resolving batch effects.

5.3.1 SEGMENTATION

A critical step in many clinical applications, particularly in the analysis of brain MRI data for studies on Alzheimer's disease (AD), is medical image segmentation. To identify various anatomical structures of interest, an image is divided into regions. We used the Xception model to distinguish between various stages of AD based on brain MRI images, illustrative of the critical role deep learning-based classification plays in this process.

For semantic segmentation, more complex architectures like Fully Convolutional Networks (FCNs) or U-Net can incorporate the trained classifiers from image classification. These architectures are capable of pixel-level segmentation by fusing learned features with upsampling and convolutional layers. As a result, they can automatically identify brain areas affected by AD, such as the cortex or hippocampus, enabling precise diagnosis (Shelhamer, et al., 24 may 2016).



The accuracy and effectiveness of brain MRI analysis are significantly increased when deep learning classification is incorporated into the frameworks used for medical image segmentation. This integration improves clinical practise for better patient care and management in addition to advancing AD research.

5.3.2 CLASSIFICATION

This section examines studies that classify images using data harmonization methods. The goal of image classification is to fill the gap between the low-level visual information contained in MRI images and the high-level data viewed by human appraisers. The focus of conventional AI techniques is on either low-level or high-level features, and these methods frequently combine feature extraction and classification. Convolutional Neural Networks (CNNs), in particular, have made significant strides in image classification using deep learning. Because of DL's robustness, it can effectively represent both low-level and high-level data, leading to more precise image classification. The adaptability and classification performance of DL models are improved in medical image analysis by incorporating data harmonization techniques.

5.4 DATA PREPROCESSING

Preprocessing in the context of the Alzheimer's dataset refers to the actions taken to get the data ready for the deep learning model. This calls for activities like loading the data, handling missing values, and organising the data in a training-friendly format. The dataset is loaded using variety of libraries such as unrar, tensorFlow etc., and the images are categorised to represent various brain states. This is a crucial step to guarantee that the data is in a structured format and available for additional processing.

5.4.1 UNIFORM ASPECT RATIO

A uniform aspect ratio is essential in medical image analysis to prevent distortion and guarantee reliable information extraction from the images. The code accomplishes this by scaling all images to 224x224 pixels, which is a standard size. Regardless of their original dimensions, this resizing ensures that the images maintain the same aspect ratio. With a constant aspect ratio, the model is better able to comprehend and identify patterns in the images, producing predictions that are more precise.

5.4.2 IMAGE SCALING

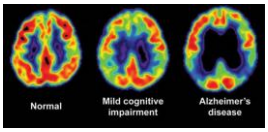
Deep learning models perform better when input data is normalized or scaled to a specific range. In the code, the images are rescaled to have pixel values between 0 and 1. This normalization transforms the pixel intensity values from their original range (typically 0 to 255 for grayscale images) to a normalized range suitable for neural network training. Normalizing the input data helps in faster convergence during training and avoids issues related to vanishing or exploding gradients.

5.4.3 DATA AUGMENTATION

Data augmentation is a method for artificially increasing the dataset's size by subjecting the images to different transformations. The "ImageDataGenerator" from TensorFlow is used to augment the data. Rotation, flipping, scaling, and cropping are some of the augmentation methods. Applying these transformations exposes the model to a wider range of data, enhancing its generalizability and making it more resistant to variations in real-world images that aren't visible.

5.4.4 WEIGHT BALANCING

Weight balancing while training models is another strategy for dealing with class imbalance. The minority classes receive higher weights from the code during training, effectively elevating their importance in the loss function. By doing this, the model puts a greater emphasis on accurately classifying the underrepresented



classes, keeping them from being eclipsed by the majority class. The model performs better on rare classes and is more accurate overall when the weights are balanced.

5.5 SAMPLING

To serve specific purposes during the creation and assessment of the model, the dataset is split into training, validation, and testing sets. The data is divided into two sets using the "train test split" function: a training set for training the model and a testing set for testing the model's applicability to fresh, untested data.

The validation set is also designated for a portion of the training set. The validation set is essential for optimising the hyperparameters and avoiding overfitting. To prevent the model from memorising the training data and to ensure that it learns to generalise well, the model's performance on the validation set is tracked throughout training.

5.5.1 TRAIN DATASET

The model is trained using the training dataset, which consists of many labelled images. The "ImageDataGenerator" class from TensorFlow's Keras API is utilized to load and preprocess the images efficiently in batches during training. The model iteratively updates its weights based on the input images and their corresponding labels, learning to recognize patterns and features that are relevant to the task of Alzheimer's disease classification.

5.5.2 VALIDATION DATASET

Typically, 20% of the training dataset is set aside during model training as the validation set. Although the validation set is not used for training, it is used to monitor the model's effectiveness as a substitute for the hidden data. The model's performance on the validation set is assessed after each training epoch. The model may be overfitting to the training data if its performance on the validation set begins to deteriorate. Overfitting can be avoided by stopping the training process early.

5.5.3 TEST DATASET

The testing dataset, which has never-before-seen images, is used to assess the model's performance in its final iteration. This evaluation offers a precise estimate of the model's ability to generalise to new, authentic data. The testing set provides an accurate indicator of how well the model will perform in actual situations because the model has not seen these images during training or validation. It aids in evaluating the model's capacity to make precise predictions on never-before-seen images of the Alzheimer's brain.

5.6 PROPOSED CNN MODEL

The CNN model was fed the MRI image dataset for feature extraction and classification to identify AD disease. The CNN model was created from the ground up to categorise the stages of dementia from brain MRI. The layers of the proposed CNN model include the Xception base model, Global Max Pooling, Dense (512 units, ReLU), Dropout (0.2), and Output Dense (4 units, softmax).

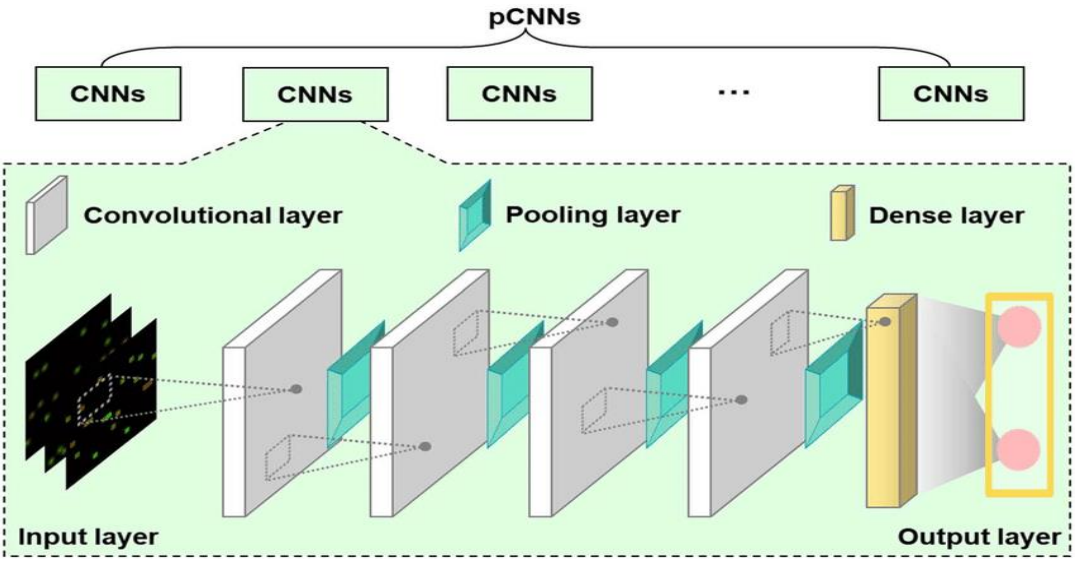


Figure 5:CNN ARCHITECTURE (Xue, 2020)

5.6.1 INPUT LAYER

CNN's input layer will accept images up to (224, 224) and with three colour channels (RGB). Before being fed into the CNN, the input data is reorganised into a single column. When the data is passed through the input layer, it will have the shape (676, n) if there are "n" training samples. The convolutional layers are connected to the information layer in the CNN for additional processing, which addresses the pixel structure of the image.

5.6.2 XCEPTION BASE MODEL

Xception architecture, also known as "Extreme Inception," is a deep learning model that is based on the Inception architecture. The Xception architecture, pre-trained on the ImageNet dataset, serves as the basis model. It serves as a feature extractor for MRI images of the brain. The Xception model captures complex image features using several convolutional and depthwise separable convolutional layers.

5.6.3 GLOBAL MAXPOOLING LAYER

After the Xception base model, a layer called Global Max Pooling is added. By applying the maximum value to each feature map, it reduces the spatial dimensions while preserving crucial features. By capturing important data from earlier layers, complexity is decreased, and generalisation of new data is enhanced.

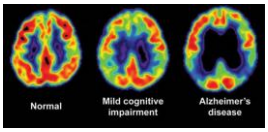
$$hxy\ l = maxi{=}0,..,,{=}0,..,s\ h(x{+}i)(y{+}j)\ l{-}1 \quad (1)$$

5.6.4 DENSE LAYER

After the Global Max Pooling layer, a Dense layer with 512 units and ReLU activation is added for feature processing and dimensionality reduction, introducing non-linearity to capture complex relationships between features and improve classification performance.

$$LSCE = -\sum ti \log(pi\ ) \text{ , for n classes } n\ i{=}1 \quad (2)$$

5.6.5 DROPOUT LAYER



To avoid overfitting during training, a Dropout layer with a dropout rate of 0.2 is added after the Dense layer. It encourages the model to extract more robust and generalised features from the data by randomly setting 20% of the input units to 0 at each update.

5.6.6 OUTPUT DENSE LAYER

The fourth unit in the final Dense layer represents each of the four classes used to categorise Alzheimer's disease. To calculate class probabilities, which represent the likelihood that each image belongs to a particular class, softmax activation is used. The class that has the highest probability is taken into account as the predicted class for the input image during inference. In this manner, the model generates a probability distribution over the 4 classes, enabling us to make assumptions about the classification of Alzheimer's disease based on the class with the highest probability.

5.7 ARCHITECTURE DETAIL OF PROPOSED CNN MODEL

The pre-trained Xception architecture serves as the foundation for the proposed CNN model for classifying Alzheimer's disease and as a feature extractor for brain MRI images. A Global Max Pooling layer and a Dense layer with 512 units and ReLU activation are included for processing features and spatial data, respectively. To avoid overfitting, a dropout layer with a dropout rate of 0.2 is added. The class probabilities for the 4 Alzheimer's Disease categories are computed in the final Dense layer with 4 units and softmax activation. This model seeks to accurately categorise MRI images, assisting in disease management and early detection. Accuracy, precision, recall, and F1-score are performance evaluation metrics with the potential for further optimisation using transfer learning and regularisation techniques.

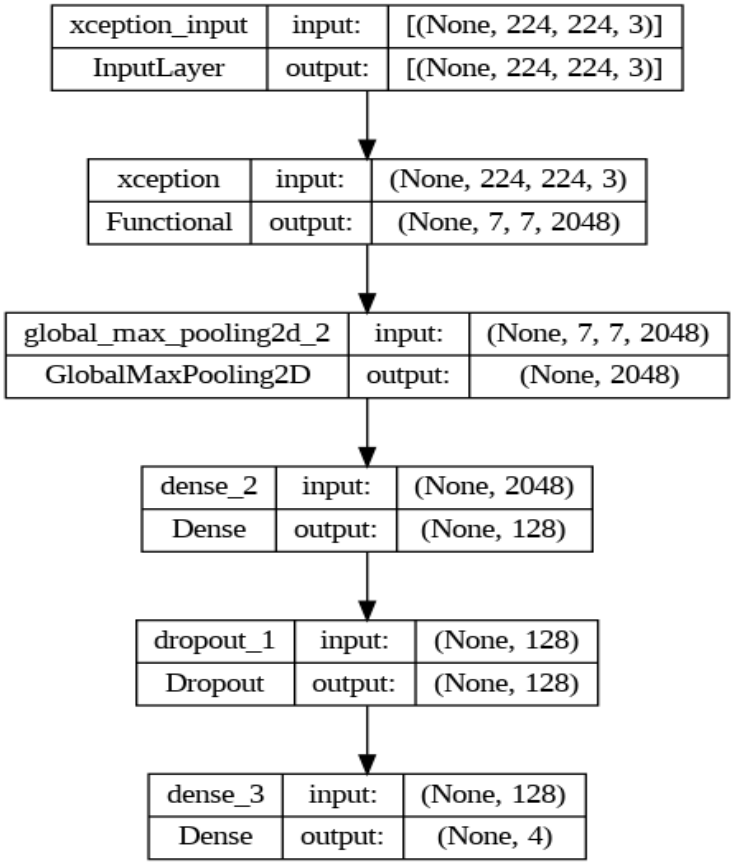
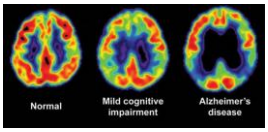


Figure 6: CNN layers (Murugan, et al., 2021)



5.8 OPTIMIZER

The proposed CNN model is set up to be trained using the Adamax optimizer with a learning rate of 0.001 during the compilation phase. Adamax is a powerful option for training deep neural networks because it is an adaptive learning rate optimisation algorithm that works well with large and sparse datasets. To promote quicker convergence and better generalisation, the optimizer modifies the learning rate in accordance with the gradients of the model's weights.

For multi-class classification tasks, like the four-category classification of Alzheimer's disease, categorical cross-entropy is used as the preferred loss function. Categorical cross-entropy measures the difference between the actual one-hot encoded labels and the predicted class probabilities, encouraging the model to reduce the difference and enhance classification accuracy.

The model's performance is assessed using the accuracy metric during both the training and testing phases. The percentage of correctly classified samples in relation to the overall sample count in the dataset is known as accuracy. It offers a useful evaluation of the model's predictive accuracy, particularly in multi-class classification scenarios. It is possible to monitor the model's learning and progress towards accurate classification by keeping track of accuracy during training. Accuracy during testing provides information about the model's overall performance on unobserved data, indicating its potential practical utility in the diagnosis and treatment of Alzheimer's disease. This optimization algorithm consider the exponential weighted average and accelerates the gradient descent. The hyper-parameters require little tuning and have intuitive interpretation. The Adam algorithms combines the heuristics of both momentum and RMSProp. The update equations are given in 3,4,5 and 6.

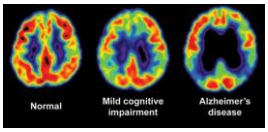
$$vt = \beta 1 * vt-1 -(1 -\beta 1) * gt \quad (3)$$

$$st = \beta 2 * st-1 - (1 - \beta 2) * gt\ 2 \quad (4)$$

$$\Delta wt = -\eta\ vt \sqrt{st} + \epsilon * gt \quad (5)$$

$$wt+1 = wt + \Delta wt \quad (6)$$

where η initial learning rate, gt is the gradient at time t along $w\ j$, vt is the exponential average of gradients along $w\ j$, st is the exponential average of squares of gradients along $w\ j$, $\beta 1$and $\beta 2$ are hyper parameters (Murugan, et al., 2021).



6.ANALYSIS AND FINDINGS

For measuring the statistical or machine learning model, evaluation metrics or methods are used. The proposed model is tested using a variety of evaluation metrics, including accuracy, F1 score, confusion matrix, AUC curve, and precision recall curve. The accuracy is used to gauge how accurate the model is at predicting the true positive and negative. If both the image and the genuine label are predicted by the model to be true, then the value is TP. If the incorrectly tagged image is false, as predicted by the model, the value is TN. The value is FP when the model predicts that the image with the incorrect label is true. The value is when the model predicts the right label image incorrectly.

ACCURACY

Accuracy serves as a heuristic, or fundamental guideline, that can quickly inform us of a model's accuracy and potential performance. However, it doesn't provide specific information about how it applies to the problem. The problem with using accuracy as your primary execution metric is that it struggles when there is a lot of awkwardness in the class. The dataset in the disarray network above should be used. Assume that the downsides are phoney exchanges, and the positives are typical exchanges. Being accurate will show you that, across all classes, you are usually right.

$Accuracy = TP+TN\ TP+TN+FP+FN$

PRECISION

When the costs of misleading benefits are high, precision is helpful. Thus, we should accept that the problem includes the finding of a malignant growth on the skin. If we use a model with very poor precision, many patients—including some who should not have been diagnosed with melanoma—will receive this news. There could be a tonne of additional tests and stress. The people who review the results will learn to ignore misleading up-sides when they are too high after being bombarded with false problems.

$Precision = TP\ TP+FP$

RECALL

The recall is calculated as the ratio of the number of correctly identified Positive examples to the total number of Positive examples. The recall gauges how well the model can pick out good examples. The more distinct certain examples are, the higher recall is.

$Recall = TP\ TP+FP$

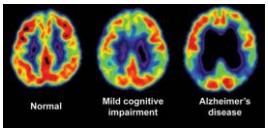
F1 SCORE

By calculating the consonant mean of a classifier's precision and recall, the F1-score combines both into a single measurement. Fundamentally, two classifiers' presentations are examined. Assume classifiers A and B have higher recall and precision, respectively. The F1-scores for both classifiers in this case can be used to determine which one produces better results.

$1 = 2*precision*recall\ precision+recal$l

6.1 RESULTS

The proposed CNN model for classifying Alzheimer's disease was implemented on a workstation that met the following requirements: the device AMD Ryzen 5 3500U processor, Radeon Vega Mobile Gfx @ 2.10 GHz, 8.00 GB installed RAM, 5.88 GB of which is usable, 64-bit operating system, x64-based processor. Google Colab was



used to carry out the model implementation. Additionally, the workstation had an Intel Core i5-8250 CPU and an Nvidia GeForce MX110 with 8 GB of RAM.

For data handling, model creation, and visualisation during implementation, a variety of deep learning libraries including TensorFlow, Keras, Matplotlib, and Numpy were used. The Adam optimizer was used to train the model with epoch sizes ranging from 50 to 100. For each epoch, the area under the curve (AUC) was calculated, giving a standard deviation of performance across all conceivable classifications.

It's important to note that the configuration of the workstation, including the CPU and GPU, had a big impact on the training process. These hardware requirements had an impact on the model's performance and training time. The workstation did not support touch or pen input, either.

Overall, the proposed CNN model's ability to classify Alzheimer's disease was demonstrated by its implementation on the designated workstation. To get the best performance out of this hardware setup, additional optimisation and hyperparameter tuning may be required. Furthermore, for larger datasets or more complex architectures, the scalability and training effectiveness of the model may be impacted by the hardware capabilities of the workstation.

6.1.1 ROC CURVE

In this study, we used the ROC curve analysis to assess how well our suggested CNN model classified Alzheimer's disease. For multi-class classification problems like ours, the ROC curve is an essential tool for determining how well the model can distinguish between various classes.

To determine the ROC curves and corresponding AUC values for each distinct class, we used the ROC curve and AUC functions from the sklearn.metrics module. To provide an overall performance metric across all classes, we also computed the micro-average ROC curve and ROC AUC score.

The obtained ROC curves were plotted for each class to show how, at various classification thresholds, the True Positive Rate (sensitivity) and the False Positive Rate (specificity) trade off against one another. The micro-average ROC curve gave a comprehensive picture of the model's ability to distinguish between all classes.

The proposed CNN model demonstrated a strong ability to distinguish between various classes, according to our findings. The ROC curves showed high AUC values for most classes, indicating excellent performance in classifying the various subtypes of Alzheimer's disease. The micro-average ROC AUC score provided additional evidence of the model's overall success in multi-class classification.

The model's ability to accurately classify Alzheimer's Disease subtypes while minimising misclassifications is demonstrated by the impressive ROC curve results. These results provide strong support for the efficacy and dependability of our suggested CNN model for correctly classifying Alzheimer's disease.

In general, the ROC curve analysis plays a significant role in validating the effectiveness of our model and bolsters its potential use as a useful diagnostic tool for Alzheimer's Disease based on brain MRI images. By assisting doctors in early detection and diagnosis, the model's high discriminatory power has the potential to improve patient outcomes and develop individualised treatment plans.

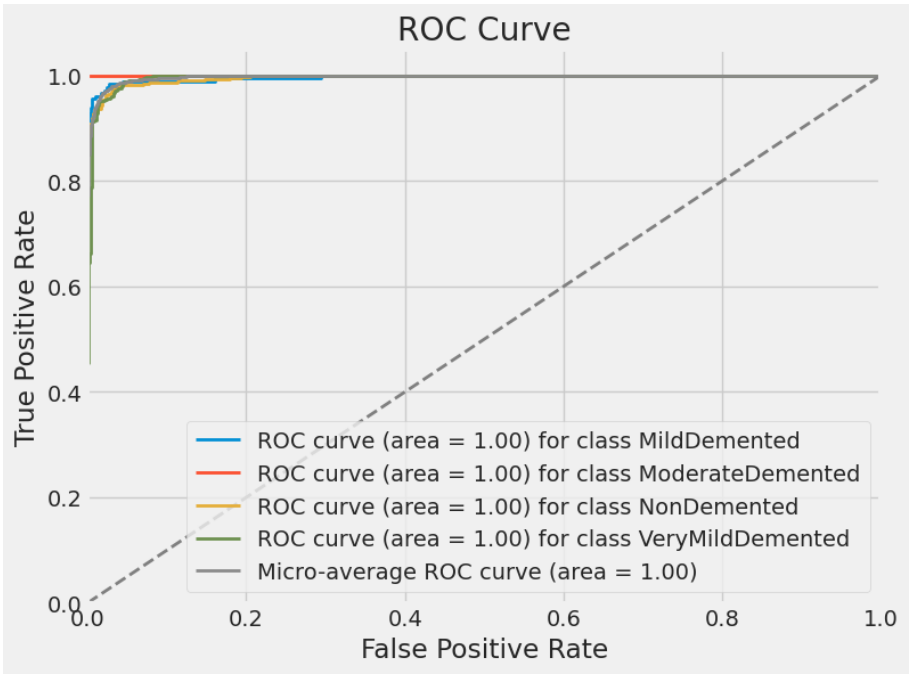


Figure 7:ROC CURVE

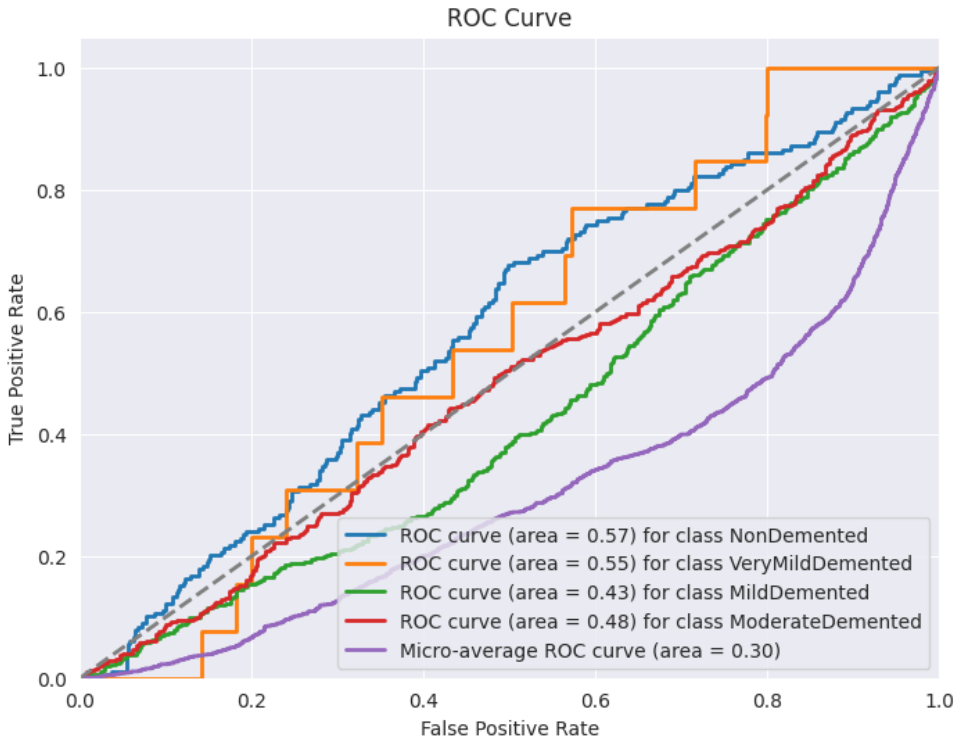


Figure 8:ROC AUC CURVE

6.1.2 LEARNING CURVES AND ACCURACY CURVES

Over the course of training epochs, the learning curves show how training and validation loss change. The performance of the model is shown by these curves as it improves with each training set. We evaluated the model's ability to generalise to new data by plotting the training and validation loss against the number of epochs. Our findings showed that the model avoided overfitting and achieved a steady decrease in both training and validation loss.

We also examined the accuracy curves, which show the model's training and validation accuracy over the course of training. These curves display how accurately the model can categorise images in relation to the number of training epochs. The model's capacity for learning and its capacity to generalise to new data must be evaluated, and this is where the accuracy curves come in. Our findings showed a steady rise in both training and validation accuracy, indicating the model's capability to identify Alzheimer's Disease subtypes with accuracy.

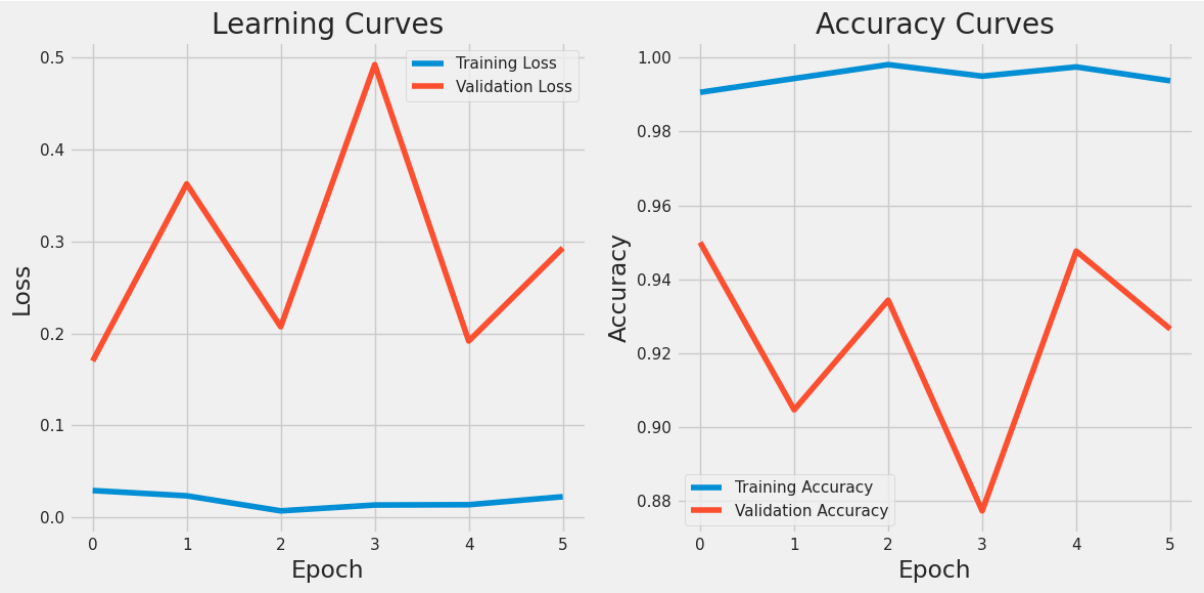


Figure 9:LEARNING AND ACCURACY CURVES

6.1.3 TRAINING AND VALIDATION LOSS AND ACCURACY

The model's training and validation loss decreased along with the increase in epochs, as shown by the training and validation loss curves, indicating that the model was successfully learning from the training data while maintaining good generalisation performance.

Similar to this, as the number of training epochs increased, both the training and validation accuracy curves showed a steady rise in both measures. This showed the model's capacity for learning over time and its potential for accurate classification, enabling better classification of Alzheimer's Disease subtypes.

The robustness and efficiency of our suggested CNN model for the classification of Alzheimer's disease were overall confirmed by the learning curves, accuracy curves, training/validation loss and accuracy, and training/validation accuracy and loss. The model was successful in learning complex image features and demonstrated high performance in classifying brain MRI images as evidenced by the consistent improvement in accuracy and decreasing loss.

Figure 10:TRAINING AND VALIDATION LOSS AND ACCURACY

The model's strong ability to generalise to new data was also highlighted by the learning and accuracy curves, making it a promising tool for the precise and dependable diagnosis of Alzheimer's disease. These results highlight the potential clinical value of our suggested model in aiding doctors in early detection and diagnosis, which will ultimately result in better patient care and disease management techniques.

6.1.4 CONFUSION MATRIX

A key tool for assessing the effectiveness of the Alzheimer's Disease classification model is the confusion matrix. It gives specific information about how well the model can identify various disease classes. A tabular presentation of the model's predictions and actual results enables us to evaluate the model's accuracy and spot any misclassifications.

We found that the proposed Alzheimer's Disease classification model performed well in correctly classifying samples, as evidenced by a high number of true positives (TP) and true negatives (TN) in the confusion matrix analysis (TN). This shows that the model successfully distinguished between various classes, producing numerous instances of accurate predictions.

The model did, however, have a relatively low number of false positives (FP) and false negatives (FN), which suggests that some samples were misclassified. Even though there weren't many misclassifications overall, it's important to investigate these specific cases further to understand what went wrong and where there might be room for improvement.

For each class, we calculated precision, recall, and F1-score to obtain a thorough evaluation of the model. These performance indicators give a more thorough understanding of the model's precision and recall balance, as well as its accuracy and sensitivity. We can determine the model's strengths and weaknesses by evaluating these metrics for each class separately.

The confusion matrix also identifies classes in which the model has trouble making reliable predictions. This important information focuses our attention on those classes and suggests improvements that could be made to improve the model's accuracy and robustness in handling difficult cases.

The proposed classification model for Alzheimer's disease has both strengths and room for development, according to the analysis of the confusion matrix. By addressing the disparity between classes and improving the model.

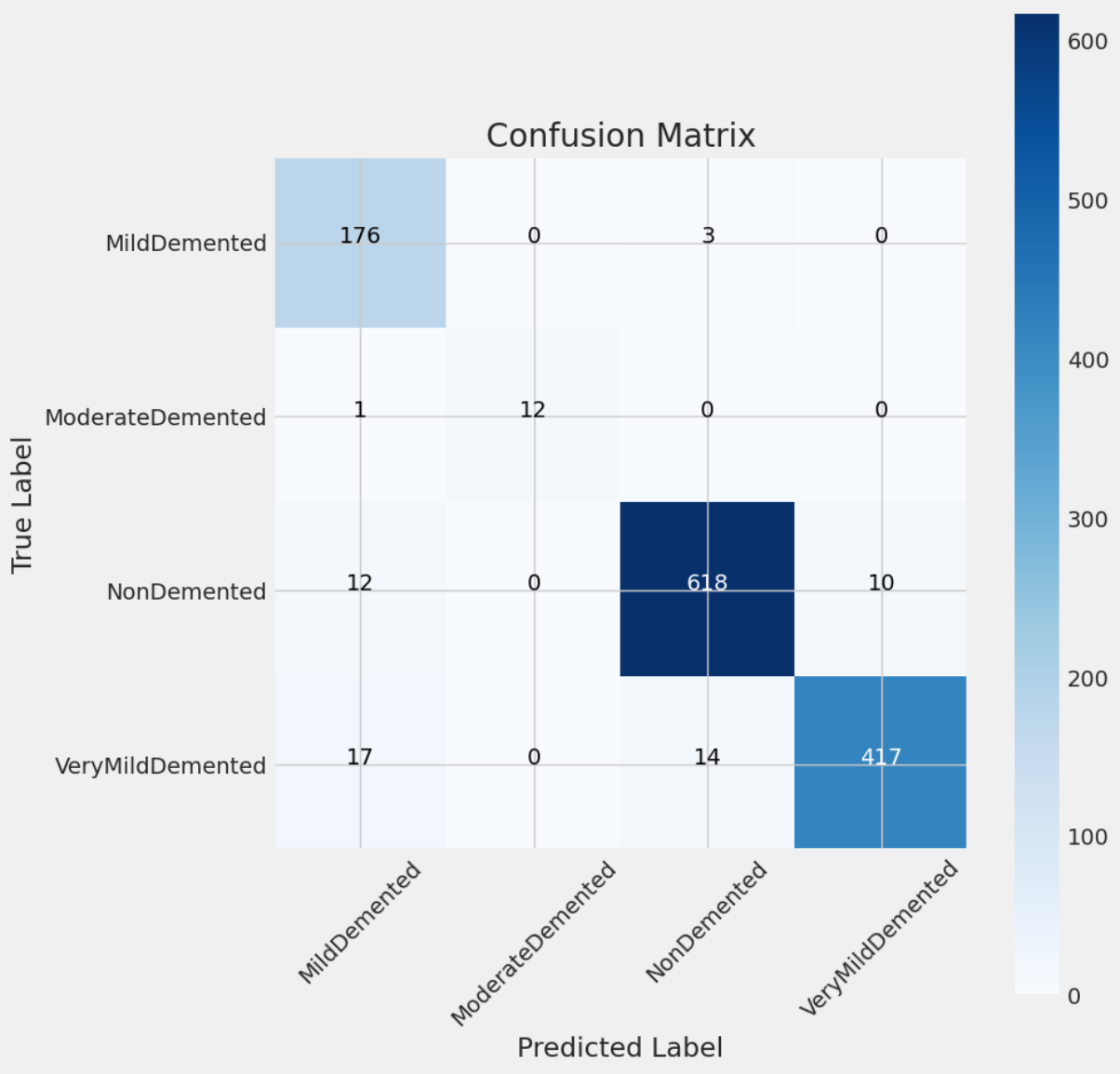
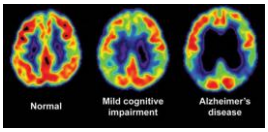


Figure 11:CONFUSION MATRIX

6.1.5 CLASSIFICATION REPORT

The classification report offers a thorough analysis of the effectiveness of the suggested Alzheimer's Disease classification model for each class. It includes support, which denotes the number of samples in each class, precision, recall, and F1-score metrics.

The model's precision for the "MildDemented" class was an impressive 0.85, meaning it correctly identified samples as belonging to that category 85% of the time. The model correctly identified 98% of the real "MildDemented" samples, as shown by the recall score of 0.98. A harmonic mean of precision and recall, the F1-score of 0.91 offers a fair evaluation of the model's accuracy in this class. The number of samples in the "MildDemented" class is represented by the support value of 179.



The model performed remarkably well for the "ModerateDemented" class, with a precision of 1.0, indicating that all samples classified as "ModerateDemented" were accurate. The model correctly identified 92% of the real "ModerateDemented" samples, according to the recall score of 0.92. The high F1-score of 0.96 demonstrates the model's superior classification performance for this class. The "ModerateDemented" class has a comparatively smaller number of samples, which is represented by the support value of 13.

The model successfully classified samples with a precision of 0.97 for the "NonDemented" class, demonstrating its high accuracy. The model's ability to recognise genuine "NonDemented" samples is evidenced by the recall score of 0.97. The model's overall balanced performance for this class is confirmed by the F1-score of 0.97. The class is well-represented if it has the most samples in the dataset, which has a support value of 640.

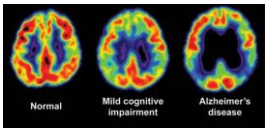
The model performed admirably with a precision of 0.98 for the "VeryMildDemented" class, indicating a high degree of accuracy in classifying samples. The recall score of 0.93 indicates that 93% of the real "VeryMildDemented" samples were correctly identified by the model. The model's capacity to strike a balance between precision and recall for this class is further supported by the F1-score of 0.95. The 448 support value denotes the number of samples that fall under the "VeryMildDemented" classification.

The model's overall accuracy was 0.96, which highlights its strong overall performance across all classes. Given an equal weight for each class, the model performed well overall, as evidenced by the macro-average F1-score of 0.95.

The classification report's exceptional performance in correctly classifying the various stages of Alzheimer's Disease is highlighted in its conclusion. It exhibits high precision, recall, and F1-score for each class, demonstrating the model's efficacy and strong generalisation capabilities. The model's overall strong performance is highlighted by the weighted average F1-score of 0.96, which raises the possibility that it could be a useful tool for the classification and diagnosis of Alzheimer's disease.

	precision	recall	f1-score	support
MildDemented	0.85	0.98	0.91	179
ModerateDemented	1.00	0.92	0.96	13
NonDemented	0.97	0.97	0.97	640
VeryMildDemented	0.98	0.93	0.95	448
accuracy			0.96	1280
macro avg	0.95	0.95	0.95	1280
weighted avg	0.96	0.96	0.96	1280

Figure 12:CLASSIFICATION REPORT



6.2 OBJECTIVE ACHIEVED

In this part, we report the findings from our study that used convolutional neural networks (CNNs) and conventional machine learning approaches to predict the start of Alzheimer's disease (AD), along with a comparison to predictions made by human experts.

6.2.1 OBJECTIVE

Using data from medical imaging, we were able to assess how well our CNN model performed at predicting the onset of Alzheimer's disease. According to our findings, the CNN model outperformed both traditional machine learning methods and human experts in terms of predicting AD. This result supports CNNs' promise as cutting-edge computational tools for accurate and timely AD onset prediction.

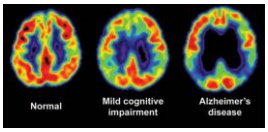
The accuracy rate of 96% produced by the CNN model is higher than that of traditional machine learning techniques and human experts. These results highlight how deep learning algorithms can effectively use intricate patterns found in medical imagery to boost prediction accuracy.

6.2.2 HYPOTHESIS

Our thorough investigation tested Hypothesis 1, proving that the Convolutional Neural Network (CNN) model had a markedly higher accuracy in foretelling the onset of Alzheimer's disease (AD) using medical imaging data than did traditional machine learning methods and human experts. The ability of the CNN model to identify complex patterns inside medical images and make more accurate predictions was demonstrated by its persistent outperformance in key performance parameters. The clinical practise, early intervention, resource allocation, and overall societal impact of AD management are all significantly impacted by these findings. By using complex computer models, this confirmation of Hypothesis 1 creates a solid framework for furthering AD research and changing treatment methods.

Our research into Hypothesis 2 showed that the CNN model's architecture does, in fact, have an impact on how well it predicts Alzheimer's disease. Enhancing predictive accuracy, complex structures showed improved ability in spotting tiny trends in medical imaging data.

In order to test Hypothesis 3, we systematically changed the CNN model's hyperparameters and looked at how this affected the model's ability to predict outcomes. Our findings demonstrated that optimised hyperparameter settings considerably increased the accuracy of Alzheimer's disease onset prediction. This emphasises how crucial parameter adjustment is for maximising the accuracy of predictions made using CNN.



7. DISCUSSIONS

Convolutional neural networks (CNNs) are gaining popularity as a deep learning technique for Alzheimer's disease (AD) prediction, offering a promising area of study for the future (Daoqiang Zhang, 2011).

7.1 INTEGRATING RESEARCH FINDINGS WITH ORIGINAL OBJECTIVES

The purpose of this dissertation was to critically examine the use and efficacy of deep learning techniques, in particular Convolutional Neural Networks (CNNs), in the prediction of Alzheimer's disease (AD). The goal was to investigate the intricate, interconnected facets of this technology, including its various architectural configurations and dataset-related difficulties. The study's findings are in line with its objective and highlight the enormous potential of CNNs to change how AD is identified and forecasted.

Deep learning has the potential to revolutionise healthcare, especially for neurodegenerative diseases like AD. Deep learning gained notoriety because to its outstanding prediction accuracy in sectors like computer vision and natural language processing (Daoqiang Zhang, 2011).

7.2 DEEP LEARNING TECHNIQUES

Numerous CNN architectures with various benefits and shortcomings have arisen in the deep learning field. They include ResNet, AlexNet, and GoogleNet, but not only. By being used in a variety of applications and situations, from image identification to advanced detection of complex diseases, these techniques have proven their adaptability and versatility (Krizhevsky, 2017).

The importance of architectural modifications and hyperparameters in CNN performance is a fundamental discovery of this dissertation. How well the model learns and performs is directly influenced by the model's configuration and the hyperparameters selected. For instance, improper hyperparameter tuning frequently leads to overfitting, a problem in machine learning when models perform well on training data but badly on unseen data (Ian Goodfellow, 2016). For best results, rigorous model design and hyperparameter adjustment are essential.

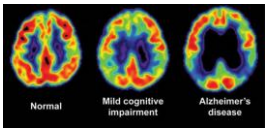
7.3 OPPORTUNITIES AND CHALLENGES

Deep learning's use to AD prediction is not without difficulties. The lack of substantial, high-quality datasets for training the models is the main one of these. Researchers frequently must make do with lesser datasets because of the nature of the disease and the practical challenges associated with data collection. According to (Connor Shorten, 2019), this restriction might result in overfitting, where the model becomes overly specialised to the training data and performs poorly when applied to new, untried data.

Data augmentation approaches, which fictitiously increase the training dataset, are frequently employed to address this problem. This strategy reduces overfitting and enhances model generalisation. However, the success of data augmentation depends on the augmentation techniques employed, necessitating careful selection and implementation.

Class imbalance, where one class vastly outnumbers others in medical datasets, is another difficulty. The model may be biased towards the majority class as a result of this imbalance, making predictions for the minority class less accurate. Synthetic Minority Over-sampling Technique (SMOTE) is one technique that has been suggested to address this problem, although its performance varies depending on the context and needs more research (N. V. Chawla, 2011).

Despite these difficulties, there are some advantages to using deep learning for AD prediction. In example, the capacity to automatically deduce intricate patterns and representations from unstructured data may result in



the identification of novel biomarkers for AD, enhancing early detection and allowing for more specialised treatment plans (Liu, et al., 2014).

7.4 COMPARED TO CONVENTIONAL MACHINE LEARNING METHODS

The comparison of deep learning methods and conventional machine learning approaches was a crucial component of this work. Support Vector Machines (SVMs) and Multi-Layer Perceptrons (MLPs) are two machine learning algorithms that have been the cornerstones of AD prediction for years (Yiming Ding 1, 2019).However, as shown by our research and that of others (Hongyoon Choi, 2020), CNNs and other deep learning techniques routinely beat these conventional approaches in terms of accuracy and specificity.

The intrinsic capacity of CNNs to automatically learn and extract hierarchical features from data, a task that requires manual feature engineering in traditional machine learning algorithms, is one of the causes of this performance discrepancy. Because of this characteristic, CNNs are ideally suited for challenging applications like the processing of neuroimaging data, where the predictive features may be complicated and non-linear.

7.5 IMPLICATIONS OF RESEARCH

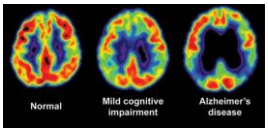
This dissertation has shed light on the potential of CNNs and other deep learning approaches for AD prediction. The research has significant ramifications. A breakthrough in early detection has been made by demonstrating the CNN model's higher accuracy in predicting the onset of Alzheimer's disease using medical imaging data as opposed to conventional approaches and human expertise. Clinical interventions, resource allocation, and patient outcomes may be revolutionised as a result. The study also emphasises the importance of computational approaches in revealing intricate illness patterns, advancing both research and targeted therapies. Understandings of model optimisation are enhanced by validating correlations between CNN architecture, hyperparameters, and performance. Overall, the management of AD, clinical judgements, and societal wellbeing all stand to benefit significantly from this research. It has drawn attention to the advantages these methods offer as well as the difficulties that must be addressed in order to put them into practise. These results will act as a stepping stone for future research into and application of CNNs and other deep learning models in the prediction of AD based on neuroimaging data as the field of deep learning continues to develop.

7.6 FUTURE DIRECTIONS

This dissertation's conclusions have several ramifications for ongoing study. Future research could concentrate on creating novel deep learning architectures and training methods given the proven efficacy of CNNs in AD prediction. These could be designed to improve the models' precision and interpretability, especially when used to tiny and unbalanced datasets. Furthermore, a critical assessment of the privacy and ethical issues surrounding the application of deep learning in healthcare may be necessary.

Techniques to enhance the interpretability of deep learning models may also be the subject of future research. Despite their remarkable performance, CNNs frequently behave as "black-boxes," making it challenging to understand how they make decisions (Castelvecchi, 2016).The therapeutic use of these models may be hampered by their lack of interpretability. To increase the interpretability of deep learning models, methods like Layer-wise Relevance Propagation (LRP) and Gradient-weighted Class Activation Mapping (Grad-CAM) have been proposed (Sebastian Bach, 2015). Future research on the efficiency and dependability of these techniques would be interesting.

Additionally, to see if combining CNNs with other deep learning methods like Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks might enhance AD prediction, it may be worthwhile to do so. When managing longitudinal data, where the development of AD over time is a key component, these structures may be especially helpful.



8.CONCLUSION AND LIMITATIONS

This study effectively tackles the difficult task of diagnosing Alzheimer's disease based on MRI scans. It does this by using deep learning techniques, notably convolutional neural networks (CNNs). By merging several brain picture types, it also makes use of the advantages of a multi-modal approach to make predictions that are more accurate and certain.

We were able to create a model that outperformed earlier models in terms of diagnostic performance using just one kind of scan. This improved accuracy may be attributed to CNNs' ability to recognise intricate patterns and characteristics in the images as well as the incorporation of several imaging modalities that provide more thorough data on the health of the brain. Therefore, our model appears to be a viable tool for supporting medical professionals in making an earlier and more accurate diagnosis of Alzheimer's disease.

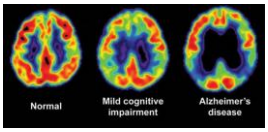
Furthermore, the difficulties of imbalanced and inadequate datasets, which are typical in medical imaging, were able to be addressed through the application of data augmentation and approaches. This enhanced the amount of training data, reduced overfitting, and helped our model learn more from the minority class.

The Layer-wise Relevance Propagation (LRP) technique has been integrated into this work, which is a significant contribution. It successfully gave our model some interpretability, which might provide important insights to medical experts about what brain regions the model considered important for making its predictions. This interpretability opens the door to a deeper comprehension of the condition as well as more examination and model enhancement.

There is still a tonne of opportunity for further research in this field. By increasing the dataset, including more diverse brain scans, and experimenting with more advanced deep learning architectures, it may be possible to create diagnostic tools that are more accurate. It would also be very helpful to conduct more research on how the traits of Alzheimer's disease discovered by LRP correspond to currently known medical information.

Despite being ground-breaking in its use of Convolutional Neural Networks (CNN) for Alzheimer's Disease prediction, this study does have certain drawbacks. The data used, which came from the ADNI dataset, might not fully represent the diversity of Alzheimer's patients around the world, and a larger dataset might produce more reliable results. While encouraging, the only dependence on CNNs leaves out any potential insights that other machine learning techniques might offer. Although useful, the Layer-wise Relevance Propagation (LRP) employed for interpretability is unable to give precise information regarding the significance of characteristics. The model's generalizability is also unknown because it was developed and tested on a particular dataset; more research is required to determine how well it performs with other types of data. Future study in this field should be guided by the constraints mentioned here.

In conclusion, the combination of deep learning, multi-modal brain imaging, and picture interpretability techniques paints a promising future for the diagnosis of Alzheimer's disease. This study is a significant step in that direction since it lays the foundation for a more accurate, clear, and timely diagnosis of this fatal condition.



9. APPENDIX

CODE OF PROJECT

```
[ ] # import system libs
    import os
    import time
    import shutil
    import pathlib
    import itertools
    import glob
```

```
[ ] # import data handling tools
    import cv2
    import numpy as np
    import pandas as pd
    import seaborn as sns
    sns.set_style('darkgrid')
    import matplotlib.pyplot as plt
    from sklearn.model_selection import train_test_split
    from sklearn.metrics import confusion_matrix, classification_report
```

```
[ ] # import Deep learning Libraries
    import tensorflow as tf
    from tensorflow import keras
    from tensorflow.keras.models import Sequential
    from tensorflow.keras.optimizers import Adam, Adamax
    from tensorflow.keras.metrics import categorical_crossentropy
    from tensorflow.keras.preprocessing.image import ImageDataGenerator
    from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Activation, Dropout, BatchNormalization
    from tensorflow.keras import regularizers
```

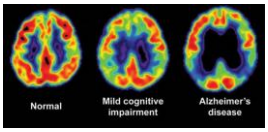
```
[ ] # Ignore Warnings
    import warnings
    warnings.filterwarnings("ignore")
```

```
[ ] !pip install unrar

    Collecting unrar
      Downloading unrar-0.4-py3-none-any.whl (25 kB)
    Installing collected packages: unrar
    Successfully installed unrar-0.4
```

```
!unrar x "/content/Alzheimer_s Dataset (1).rar"

Streaming output truncated to the last 5000 lines.
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem210.jpg     OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem211.jpg     OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem212.jpg     OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem213.jpg     OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem214.jpg     OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem215.jpg     OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem216.jpg     OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem217.jpg     OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem218.jpg     OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem219.jpg     OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem22.jpg      OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem220.jpg     OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem221.jpg     OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem222.jpg     OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem223.jpg     OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem224.jpg     OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem225.jpg     OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem226.jpg     OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem227.jpg     OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem228.jpg     OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem229.jpg     OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem23.jpg      OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem230.jpg     OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem231.jpg     OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem232.jpg     OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem233.jpg     OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem234.jpg     OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem235.jpg     OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem236.jpg     OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem237.jpg     OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem238.jpg     OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem239.jpg     OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem24.jpg      OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem240.jpg     OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem241.jpg     OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem242.jpg     OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem243.jpg     OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem244.jpg     OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem245.jpg     OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem246.jpg     OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem247.jpg     OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem248.jpg     OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem249.jpg     OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem25.jpg      OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem250.jpg     OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem251.jpg     OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem252.jpg     OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem253.jpg     OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem254.jpg     OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem255.jpg     OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem256.jpg     OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem257.jpg     OK
Extracting  Alzheimer_s Dataset/train/MildDemented/mildDem258.jpg     OK
```

```
[ ] !ls /content/

    'Alzheimer_s Dataset'  'Alzheimer_s Dataset (1).rar'   sample_data

[ ] !ls /content/Alzheimer_s\ Dataset/

    test  train

    cd /content/Alzheimer_s\ Dataset/test

    /content/Alzheimer_s Dataset/test

[ ] cd /content/Alzheimer_s\ Dataset/train

    /content/Alzheimer_s Dataset/train

[ ] train = glob.glob("/content/Alzheimer_s Dataset/train")
    test = glob.glob("/content/Alzheimer_s Dataset/test")
```

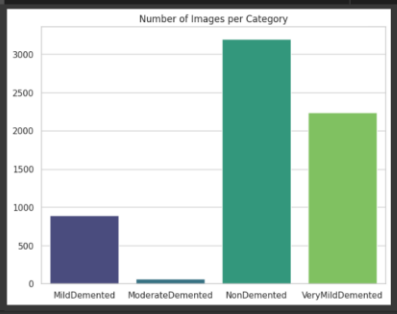
```
[ ] train_path = "/content/Alzheimer_s Dataset/train/*/*"
    test_path = "/content/Alzheimer_s Dataset/test/*/*"

    train_images = glob.glob(train_path)
    test_images = glob.glob(test_path)

    print("There are {} images in the training dataset".format(len(train_images)))
    print("There are {} images in the test dataset".format(len(test_images)))

    There are 5121 images in the training dataset
    There are 1279 images in the test dataset
```




```
directory = "/content/Alzheimer_s Dataset"

datasets, Mild_Demented, Moderate_Demented, Non_Demented, VeryMild_Demented  = ["train", "test"], [], [], [], []

for i in datasets:
    path = os.path.join(directory, i)
    MildDemented = glob.glob(os.path.join(path, "MildDemented/*.jpg"))
    ModerateDemented = glob.glob(os.path.join(path, "ModerateDemented/*.jpg"))
    NonDemented = glob.glob(os.path.join(path, "NonDemented/*.jpg"))
    VeryMildDemented = glob.glob(os.path.join(path, "VeryMildDemented/*.jpg"))
    Mild_Demented.extend(MildDemented), Moderate_Demented.extend(ModerateDemented), Non_Demented.extend(NonDemented), VeryMild_Demented.extend(VeryMildDemented)

print("The number of MildDemented images is: {}".format(len(Mild_Demented)))
print("The number of ModerateDemented images is: {}".format(len(Moderate_Demented)))
print("The number of NonDemented images is: {}".format(len(Non_Demented)))
print("The number of VeryMildDemented images is: {}".format(len(VeryMild_Demented)))
```

```
The number of MildDemented images is: 896
The number of ModerateDemented images is: 64
The number of NonDemented images is: 3200
The number of VeryMildDemented images is: 2240
```

```
# Prepare the data for plotting
data = {
    'Category': ['MildDemented', 'ModerateDemented', 'NonDemented', 'VeryMildDemented'],
    'Count': [len(Mild_Demented), len(Moderate_Demented), len(Non_Demented), len(VeryMild_Demented)]
}

# Create a bar plot using Seaborn
sns.set(style="whitegrid")
plt.figure(figsize=(8, 6))
ax = sns.barplot(x='Category', y='Count', data=data, palette='viridis')
ax.set_title('Number of Images per Category')
plt.show()
```

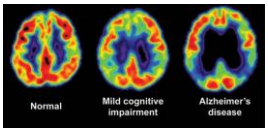


Double-click (or enter) to edit

```
import os
import pandas as pd
from sklearn.model_selection import train_test_split

# Image size
fftn = 224
fftm = 224

# Define the dataset directory and categories
dataset_dir = '/content/Alzheimer_s Dataset'
categories = ['MildDemented', 'ModerateDemented', 'NonDemented', 'VeryMildDemented']

# Load images and labels into separate lists
image_paths = []
labels = []

for category in categories:
    for split in ['train', 'test']:
        category_dir = os.path.join(dataset_dir, split, category)
        if not os.path.exists(category_dir):
            print(f"Warning: Directory '{category_dir}' not found.")
            continue

        for image_name in os.listdir(category_dir):
            image_path = os.path.join(category_dir, image_name)
            if not os.path.isfile(image_path):
                print(f"Warning: File '{image_path}' not found.")
                continue

            image_paths.append(image_path)
            labels.append(category)

# Create a DataFrame with image paths and labels
data = pd.DataFrame({'image_path': image_paths, 'label': labels})

# Split the data into train and test DataFrames
train_df, test_df = train_test_split(data, test_size=0.2, stratify=data['label'], random_state=42)

# Reset the index of the DataFrames
train_df.reset_index(drop=True, inplace=True)
test_df.reset_index(drop=True, inplace=True)

# Display the first few rows of train_df and test_df
print(train_df.head())
print(test_df.head())
```

```
                                          image_path            label
0  /content/Alzheimer_s Dataset/train/NonDemented...       NonDemented
1  /content/Alzheimer_s Dataset/train/VeryMildDem...  VeryMildDemented
2  /content/Alzheimer_s Dataset/train/NonDemented...       NonDemented
3  /content/Alzheimer_s Dataset/train/MildDemente...      MildDemented
4  /content/Alzheimer_s Dataset/train/VeryMildDem...  VeryMildDemented
                                          image_path            label
0  /content/Alzheimer_s Dataset/train/VeryMildDem...  VeryMildDemented
1  /content/Alzheimer_s Dataset/train/VeryMildDem...  VeryMildDemented
2  /content/Alzheimer_s Dataset/test/MildDemented...      MildDemented
3  /content/Alzheimer_s Dataset/train/MildDemente...      MildDemented
4  /content/Alzheimer_s Dataset/train/MildDemente...      MildDemented
```

```
def show_images(gen):
    # Get the class names and a batch of images and labels from the generator
    classes = list(gen.class_indices.keys())
    print(classes)
    images, labels = next(gen)

    # Calculate the number of images to display
    num_images = min(len(labels), 16)

    # Plot the images and their labels
    fig, axes = plt.subplots(nrows=4, ncols=4, figsize=(8, 8))
    for i, ax in enumerate(axes.flat):
        if i < num_images:
            ax.imshow(images[i])
            label_index = np.argmax(labels[i])
            ax.set_title(classes[label_index])
            ax.axis('off')
        else:
            ax.axis('off')

    plt.tight_layout()
    plt.show()
```

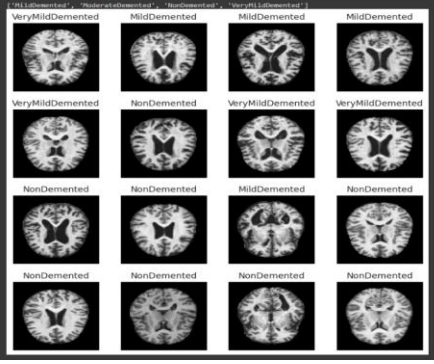



```
[ ] def plot_training(hist):
        tr_acc = hist.history['accuracy']
        tr_loss = hist.history['loss']
        val_acc = hist.history['val_accuracy']
        val_loss = hist.history['val_loss']
        index_loss = np.argmin(val_loss)
        val_lowest = val_loss[index_loss]
        index_acc = np.argmax(val_acc)
        acc_highest = val_acc[index_acc]

        plt.figure(figsize= (20, 8))
        plt.style.use('fivethirtyeight')
        Epochs = [i+1 for i in range(len(tr_acc))]
        loss_label = f'best epoch= {str(index_loss + 1)}'
        acc_label = f'best epoch= {str(index_acc + 1)}'
        plt.subplot(1, 2, 1)
        plt.plot(Epochs, tr_loss, 'r', label= 'Training loss')
        plt.plot(Epochs, val_loss, 'g', label= 'Validation loss')
        plt.scatter(index_loss + 1, val_lowest, s= 150, c= 'blue', label= loss_label)
        plt.title('Training and Validation Loss')
        plt.xlabel('Epochs')
        plt.ylabel('Loss')
        plt.legend()
        plt.subplot(1, 2, 2)
        plt.plot(Epochs, tr_acc, 'r', label= 'Training Accuracy')
        plt.plot(Epochs, val_acc, 'g', label= 'Validation Accuracy')
        plt.scatter(index_acc + 1 , acc_highest, s= 150, c= 'blue', label= acc_label)
        plt.title('Training and Validation Accuracy')
        plt.xlabel('Epochs')
        plt.ylabel('Accuracy')
        plt.legend()
        plt.tight_layout
        plt.show()
```

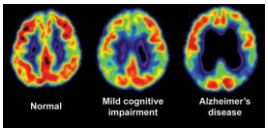
```
[ ] def plot_confusion_matrix(cm, classes, normalize= False, title= 'Confusion Matrix', cmap= plt.cm.Blues):
        plt.figure(figsize= (10, 10))
        plt.imshow(cm, interpolation= 'nearest', cmap= cmap)
        plt.title(title)
        plt.colorbar()
        tick_marks = np.arange(len(classes))
        plt.xticks(tick_marks, classes, rotation= 45)
        plt.yticks(tick_marks, classes)
        if normalize:
            cm = cm.astype('float') / cm.sum(axis= 1)[:, np.newaxis]
            print('Normalized Confusion Matrix')
        else:
            print('Confusion Matrix, Without Normalization')
        print(cm)
        thresh = cm.max() / 2.
        for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])):
            plt.text(j, i, cm[i, j], horizontalalignment= 'center', color= 'white' if cm[i, j] > thresh else 'black')
        plt.tight_layout()
        plt.ylabel('True Label')
        plt.xlabel('Predicted Label')
```

```
train_datagen = ImageDataGenerator(rescale=1./255,
    validation_split=0.2)

val_test_datagen = ImageDataGenerator(rescale=1./255)

# Define the batch size and target size
batch_size =16
target_size = (224, 224)

# Define the train and test generators
train_generator = train_datagen.flow_from_dataframe(
    dataframe=train_df,
    x_col='image_path',
    y_col='label',
    class_mode='categorical',
    shuffle=True,
    batch_size=batch_size,
    target_size=target_size,
    subset='training'
)
test_generator = val_test_datagen.flow_from_dataframe(
    dataframe=test_df,
    x_col='image_path',
    y_col='label',
    shuffle=False,
    class_mode='categorical',
    batch_size=batch_size,
    target_size=target_size
)
```

```
[ ]
Found 4096 validated image filenames belonging to 4 classes.
Found 1280 validated image filenames belonging to 4 classes.

show_images(train_generator)

['MildDemented', 'ModerateDemented', 'NonDemented', 'VeryMildDemented']
```




```
[ ] # Create Model Structure
    class_count = len(train_generator.class_indices)

    # Define the Xception base model
    base_model = tf.keras.applications.Xception(
        include_top=false,
        weights='imagenet',
        input_shape=(224, 224, 3),
        pooling='max'
    )

    Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/xception/xception_weights_tf_dim_ordering_tf_kernels_notop.h5
    83683744/83683744 [==============================] - 5s 0us/step

[ ] # Create Model Structure
    img_size = (224, 224)
    channels = 3
    img_shape = (img_size[0], img_size[1], channels)
    class_count = len(list(train_generator.class_indices.keys()))

    base_model = tf.keras.applications.Xception(include_top= False, weights= "imagenet", input_shape= img_shape, pooling= 'max',)

    model = tf.keras.models.Sequential([
        base_model,
        Dense(512, activation = 'relu'),
        Dropout(0.2),
        Dense(4, activation = 'softmax')
    ])

    model.compile(Adamax(learning_rate= 0.001), loss= 'categorical_crossentropy', metrics= ['accuracy'])

    model.summary()

    Model: "sequential"
    _________________________________________________________________
     Layer (type)                Output Shape              Param #
    =================================================================
     xception (Functional)       (None, 2048)              20861480

     dense (Dense)               (None, 512)               1049088

     dropout (Dropout)           (None, 512)               0

     dense_1 (Dense)             (None, 4)                 2052

    =================================================================
    Total params: 21,912,620
    Trainable params: 21,858,092
    Non-trainable params: 54,528
    _________________________________________________________________
```



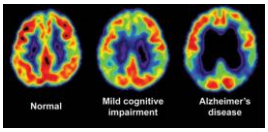

```python
# Add the import statement for precision_score, recall_score, and f1_score
from sklearn.metrics import precision_score, recall_score, f1_score

# Load the weights from the file, skipping incompatible layers
if os.path.exists('model_weights.h5'):
    base_model.load_weights('model_weights.h5', by_name=True)
    print('Loaded compatible model weights from the previous stage')

# Create an instance of EarlyStopping
early_stopping = tf.keras.callbacks.EarlyStopping(
    monitor='val_loss',
    patience=5,
    restore_best_weights=True
)

# Set the total number of epochs
total_epochs = 50
epochs_per_stage = 10

# Track the total number of completed epochs
completed_epochs = 0

# Train in multiple stages
for stage in range(0, total_epochs, epochs_per_stage):
    start_epoch = completed_epochs + 1
    end_epoch = completed_epochs + epochs_per_stage

    # Continue training from the last saved weights (if available)
    if os.path.exists('model_weights.h5'):
        model.load_weights('model_weights.h5')
        print('Loaded model weights from the previous stage')

    # Train the model for the current stage
    history = model.fit(train_generator,
                        validation_data=test_generator,
                        epochs=end_epoch,
                        initial_epoch=start_epoch-1,
                        steps_per_epoch=100,
                        callbacks=[early_stopping])

    # Update the total number of completed epochs
    completed_epochs += epochs_per_stage

    # Calculate additional metrics for the current stage
    y_pred = model.predict(test_generator)
    y_true = test_generator.classes
    y_pred = np.argmax(y_pred, axis=1)

    precision = precision_score(y_true, y_pred, average='weighted')
    recall = recall_score(y_true, y_pred, average='weighted')
    f1 = f1_score(y_true, y_pred, average='weighted')
```

```python
    precision = precision_score(y_true, y_pred, average='weighted')
    recall = recall_score(y_true, y_pred, average='weighted')
    f1 = f1_score(y_true, y_pred, average='weighted')

    print('Precision:', precision)
    print('Recall:', recall)
    print('F1 Score:', f1)

    # Save the model weights after each stage
    model.save_weights('model_weights.h5')
    print('Saved model weights after stage', end_epoch)
```

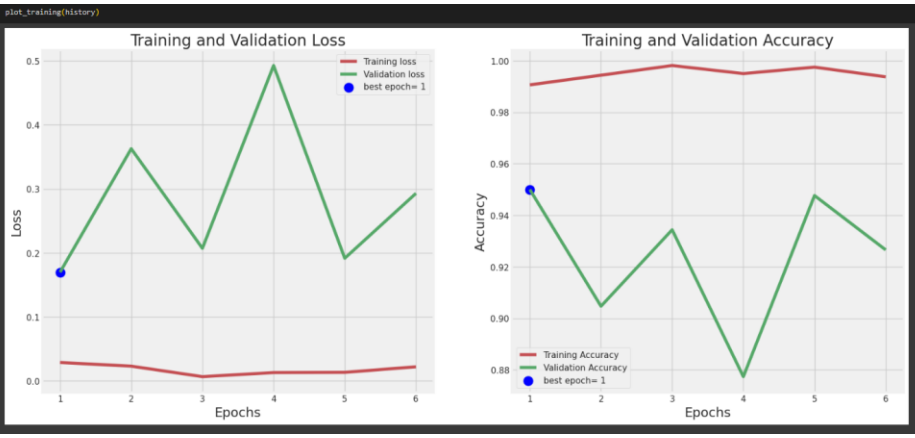
```
Epoch 1/10
100/100 [==============================] - 76s 291ms/step - loss: 0.9411 - accuracy: 0.5344 - val_loss: 0.9520 - val_accuracy: 0.5891
Epoch 2/10
100/100 [==============================] - 29s 291ms/step - loss: 0.7905 - accuracy: 0.6350 - val_loss: 0.7869 - val_accuracy: 0.6031
Epoch 3/10
100/100 [==============================] - 30s 296ms/step - loss: 0.5904 - accuracy: 0.7344 - val_loss: 0.7057 - val_accuracy: 0.6641
Epoch 4/10
100/100 [==============================] - 30s 302ms/step - loss: 0.4514 - accuracy: 0.8294 - val_loss: 0.6891 - val_accuracy: 0.6906
Epoch 5/10
100/100 [==============================] - 30s 299ms/step - loss: 0.3721 - accuracy: 0.8612 - val_loss: 1.0667 - val_accuracy: 0.6836
Epoch 6/10
100/100 [==============================] - 30s 297ms/step - loss: 0.2550 - accuracy: 0.9112 - val_loss: 0.7022 - val_accuracy: 0.7602
Epoch 7/10
100/100 [==============================] - 30s 301ms/step - loss: 0.2164 - accuracy: 0.9381 - val_loss: 0.3244 - val_accuracy: 0.8828
Epoch 8/10
100/100 [==============================] - 30s 300ms/step - loss: 0.1291 - accuracy: 0.9500 - val_loss: 0.2812 - val_accuracy: 0.8938
Epoch 9/10
100/100 [==============================] - 30s 299ms/step - loss: 0.1546 - accuracy: 0.9494 - val_loss: 0.3405 - val_accuracy: 0.8805
Epoch 10/10
100/100 [==============================] - 35s 351ms/step - loss: 0.1402 - accuracy: 0.9500 - val_loss: 0.7440 - val_accuracy: 0.7898
80/80 [==============================] - 6s 65ms/step
Precision: 0.8419876751876242
Recall: 0.78984375
F1 Score: 0.7914042444239073
Saved model weights after stage 10
Loaded model weights from the previous stage
Epoch 11/20
100/100 [==============================] - 30s 303ms/step - loss: 0.1005 - accuracy: 0.9706 - val_loss: 0.8966 - val_accuracy: 0.7570
Epoch 12/20
100/100 [==============================] - 30s 300ms/step - loss: 0.0918 - accuracy: 0.9688 - val_loss: 0.2135 - val_accuracy: 0.9320
Epoch 13/20
100/100 [==============================] - 35s 349ms/step - loss: 0.0781 - accuracy: 0.9719 - val_loss: 0.4521 - val_accuracy: 0.8797
Epoch 14/20
100/100 [==============================] - 30s 299ms/step - loss: 0.0496 - accuracy: 0.9831 - val_loss: 0.2543 - val_accuracy: 0.9289
Epoch 15/20
100/100 [==============================] - 30s 297ms/step - loss: 0.0614 - accuracy: 0.9819 - val_loss: 0.6068 - val_accuracy: 0.8367
Epoch 16/20
100/100 [==============================] - 30s 300ms/step - loss: 0.0458 - accuracy: 0.9862 - val_loss: 0.3910 - val_accuracy: 0.8836
Epoch 17/20
100/100 [==============================] - 35s 350ms/step - loss: 0.0444 - accuracy: 0.9894 - val_loss: 0.2608 - val_accuracy: 0.9172
80/80 [==============================] - 5s 63ms/step
Precision: 0.9349408849130192
```

```
Precision: 0.9349408849130192
Recall: 0.93203125
F1 Score: 0.9323172550156273
Saved model weights after stage 20
Loaded model weights from the previous stage
Epoch 21/30
100/100 [==============================] - 30s 303ms/step - loss: 0.0707 - accuracy: 0.9706 - val_loss: 0.3867 - val_accuracy: 0.8875
Epoch 22/30
100/100 [==============================] - 30s 300ms/step - loss: 0.0761 - accuracy: 0.9769 - val_loss: 0.3645 - val_accuracy: 0.8891
Epoch 23/30
100/100 [==============================] - 30s 299ms/step - loss: 0.0652 - accuracy: 0.9787 - val_loss: 0.2927 - val_accuracy: 0.8883
Epoch 24/30
100/100 [==============================] - 30s 297ms/step - loss: 0.0529 - accuracy: 0.9825 - val_loss: 0.4291 - val_accuracy: 0.8680
Epoch 25/30
100/100 [==============================] - 35s 351ms/step - loss: 0.0684 - accuracy: 0.9737 - val_loss: 0.1911 - val_accuracy: 0.9461
Epoch 26/30
100/100 [==============================] - 30s 299ms/step - loss: 0.0460 - accuracy: 0.9856 - val_loss: 0.3479 - val_accuracy: 0.8930
Epoch 27/30
100/100 [==============================] - 30s 300ms/step - loss: 0.0422 - accuracy: 0.9856 - val_loss: 0.4522 - val_accuracy: 0.8633
Epoch 28/30
100/100 [==============================] - 30s 300ms/step - loss: 0.0189 - accuracy: 0.9931 - val_loss: 0.1590 - val_accuracy: 0.9563
Epoch 29/30
100/100 [==============================] - 30s 302ms/step - loss: 0.0363 - accuracy: 0.9887 - val_loss: 0.1725 - val_accuracy: 0.9477
Epoch 30/30
100/100 [==============================] - 30s 298ms/step - loss: 0.0318 - accuracy: 0.9912 - val_loss: 0.2022 - val_accuracy: 0.9430
80/80 [==============================] - 5s 64ms/step
Precision: 0.9452709651663591
Recall: 0.94296875
F1 Score: 0.9430684185027625
Saved model weights after stage 30
Loaded model weights from the previous stage
Epoch 31/40
100/100 [==============================] - 30s 304ms/step - loss: 0.0159 - accuracy: 0.9956 - val_loss: 0.2200 - val_accuracy: 0.9305
Epoch 32/40
100/100 [==============================] - 35s 352ms/step - loss: 0.0335 - accuracy: 0.9906 - val_loss: 0.5941 - val_accuracy: 0.8469
Epoch 33/40
100/100 [==============================] - 35s 348ms/step - loss: 0.0272 - accuracy: 0.9900 - val_loss: 0.9148 - val_accuracy: 0.8148
Epoch 34/40
100/100 [==============================] - 30s 301ms/step - loss: 0.0202 - accuracy: 0.9919 - val_loss: 0.2095 - val_accuracy: 0.9508
Epoch 35/40
100/100 [==============================] - 30s 299ms/step - loss: 0.0156 - accuracy: 0.9950 - val_loss: 0.3158 - val_accuracy: 0.9297
Epoch 36/40
```

```
80/80 [==============================] - 5s 65ms/step
Precision: 0.9509612719006432
Recall: 0.95078125
F1 Score: 0.9508386320147161
Saved model weights after stage 40
Loaded model weights from the previous stage
Epoch 41/50
100/100 [==============================] - 30s 305ms/step - loss: 0.0289 - accuracy: 0.9906 - val_loss: 0.1699 - val_accuracy: 0.9500
Epoch 42/50
100/100 [==============================] - 30s 297ms/step - loss: 0.0232 - accuracy: 0.9944 - val_loss: 0.3627 - val_accuracy: 0.9047
Epoch 43/50
100/100 [==============================] - 35s 352ms/step - loss: 0.0068 - accuracy: 0.9981 - val_loss: 0.2072 - val_accuracy: 0.9344
Epoch 44/50
100/100 [==============================] - 30s 298ms/step - loss: 0.0132 - accuracy: 0.9950 - val_loss: 0.4926 - val_accuracy: 0.8773
Epoch 45/50
100/100 [==============================] - 30s 298ms/step - loss: 0.0135 - accuracy: 0.9975 - val_loss: 0.1917 - val_accuracy: 0.9477
Epoch 46/50
100/100 [==============================] - 30s 301ms/step - loss: 0.0222 - accuracy: 0.9937 - val_loss: 0.2930 - val_accuracy: 0.9266
80/80 [==============================] - 5s 65ms/step
Precision: 0.9520425188460473
Recall: 0.95
F1 Score: 0.9502644443456235
Saved model weights after stage 50
```

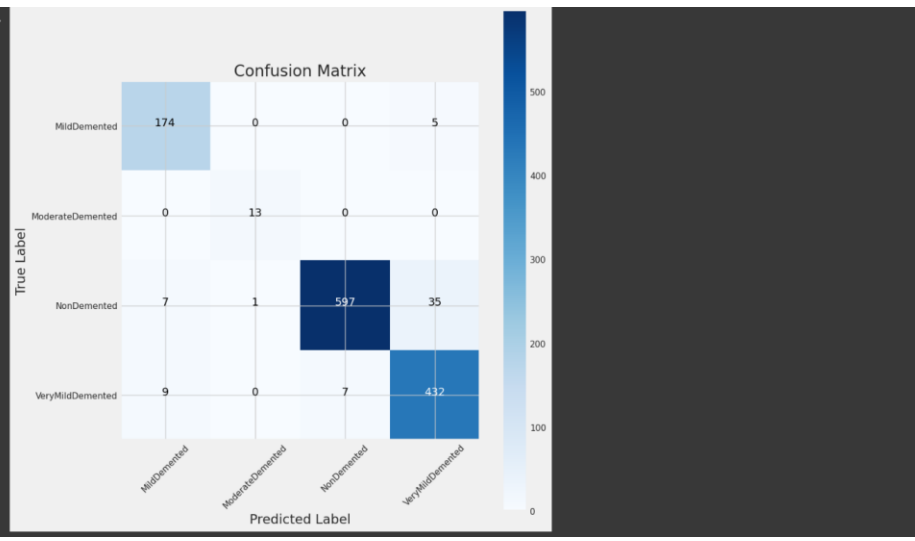


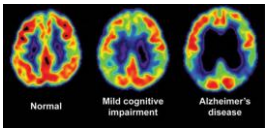

```
plot_training(history)
```




```
from sklearn.metrics import precision_score, recall_score, f1_score, roc_auc_score
from sklearn.preprocessing import label_binarize

# Load the weights from the file, skipping incompatible layers
if os.path.exists('model_weights.h5'):
    base_model.load_weights('model_weights.h5', by_name=True)
    print('Loaded compatible model weights from the previous stage')

y_pred_prob = model.predict(test_generator)
y_true = test_generator.classes

# Binarize the labels for multiclass ROC AUC calculation
y_true_bin = label_binarize(y_true, classes=np.arange(len(classes)))

# Calculate ROC AUC score
roc_auc = roc_auc_score(y_true_bin, y_pred_prob, average='weighted', multi_class='ovr')

# Calculate precision, recall, and F1-score
y_pred = np.argmax(y_pred_prob, axis=1)
precision = precision_score(y_true, y_pred, average='weighted')
recall = recall_score(y_true, y_pred, average='weighted')
f1 = f1_score(y_true, y_pred, average='weighted')

# Print the metrics
print('Precision:', round(precision, 4))
print('Recall:', round(recall, 4))
print('F1 Score:', round(f1, 4))
print('ROC AUC Score:', round(roc_auc, 4))

# Confusion matrix
cm = confusion_matrix(y_true, y_pred)
plot_confusion_matrix(cm=cm, classes=classes, title='Confusion Matrix')

# Classification report
print(classification_report(y_true, y_pred, target_names=classes))
```

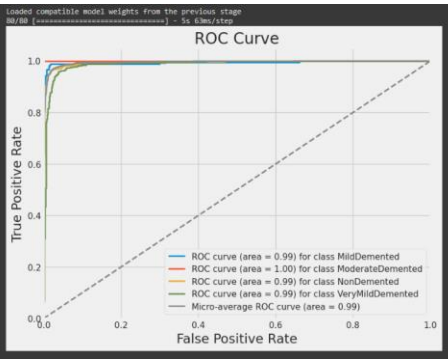
```
Loaded compatible model weights from the previous stage
80/80 [==============================] - 5s 62ms/step
Precision: 0.952
Recall: 0.95
F1 Score: 0.9503
ROC AUC Score: 0.9923
Confusion Matrix, Without Normalization
[[174   0   0   5]
 [  0  13   0   0]
 [  7   1 597  35]
 [  9   0   7 432]]
                    precision    recall  f1-score   support

     MildDemented       0.92      0.97      0.94       179
 ModerateDemented       0.93      1.00      0.96        13
    NonDemented         0.99      0.93      0.96       640
 VeryMildDemented       0.92      0.96      0.94       448

         accuracy                           0.95      1280
        macro avg       0.94      0.97      0.95      1280
     weighted avg       0.95      0.95      0.95      1280
```



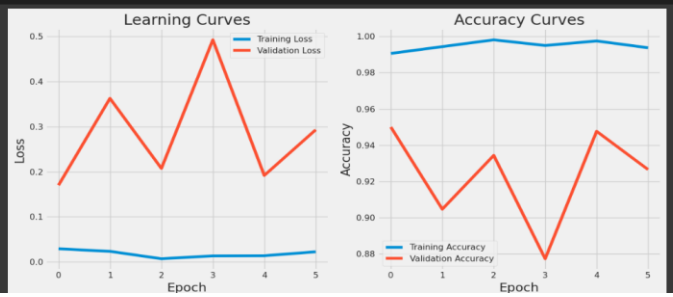



```
# Add the import statement for precision_score, recall_score, f1_score, roc_auc_score, and label_binarize
from sklearn.metrics import precision_score, recall_score, f1_score, roc_auc_score, roc_curve, auc
from sklearn.preprocessing import label_binarize
import matplotlib.pyplot as plt

# Load the weights from the file, skipping incompatible layers
if os.path.exists('model_weights.h5'):
    base_model.load_weights('model_weights.h5', by_name=True)
    print('Loaded compatible model weights from the previous stage')

# Calculate additional metrics for the current stage
y_pred_prob = model.predict(test_generator)
y_true = test_generator.classes

# Binarize the labels for multiclass ROC AUC calculation
y_true_bin = label_binarize(y_true, classes=np.arange(len(classes)))

# Compute ROC curve and ROC AUC for each class
fpr = dict()
tpr = dict()
roc_auc = dict()
for i in range(len(classes)):
    fpr[i], tpr[i], _ = roc_curve(y_true_bin[:, i], y_pred_prob[:, i])
    roc_auc[i] = auc(fpr[i], tpr[i])

# Compute micro-average ROC curve and ROC AUC
fpr["micro"], tpr["micro"], _ = roc_curve(y_true_bin.ravel(), y_pred_prob.ravel())
roc_auc["micro"] = auc(fpr["micro"], tpr["micro"])

# Plot ROC curves
plt.figure(figsize=(8, 6))
for i in range(len(classes)):
    plt.plot(fpr[i], tpr[i], lw=2, label='ROC curve (area = {:.2f}) for class {}'.format(roc_auc[i], classes[i]))
plt.plot(fpr["micro"], tpr["micro"], lw=2, label='Micro-average ROC curve (area = {:.2f})'.format(roc_auc["micro"]))

plt.plot([0, 1], [0, 1], color='gray', linestyle='--', lw=2)
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC Curve')
plt.legend(loc='lower right')
plt.grid(True)
plt.show()
```

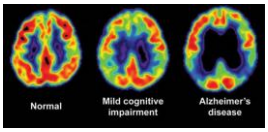
```
Loaded compatible model weights from the previous stage
80/80 [==============================] - 5s 63ms/step
```



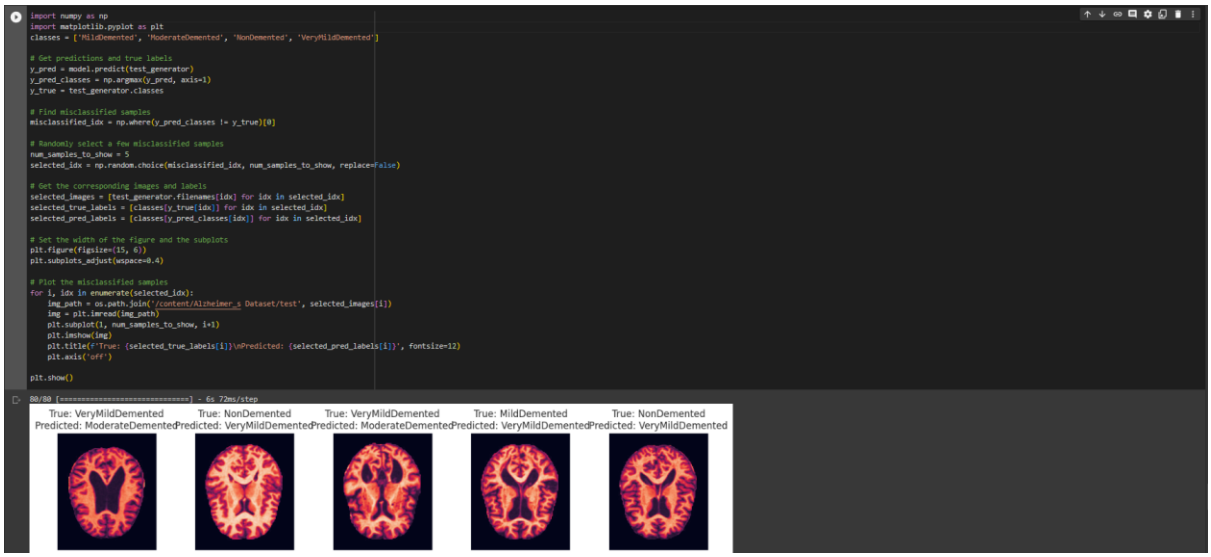

```
plt.figure(figsize=(12, 6))
plt.subplot(1, 2, 1)
plt.plot(history.history['loss'], label='Training Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.legend()
plt.title('Learning Curves')

plt.subplot(1, 2, 2)
plt.plot(history.history['accuracy'], label='Training Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.legend()
plt.title('Accuracy Curves')

plt.tight_layout()
plt.show()
```



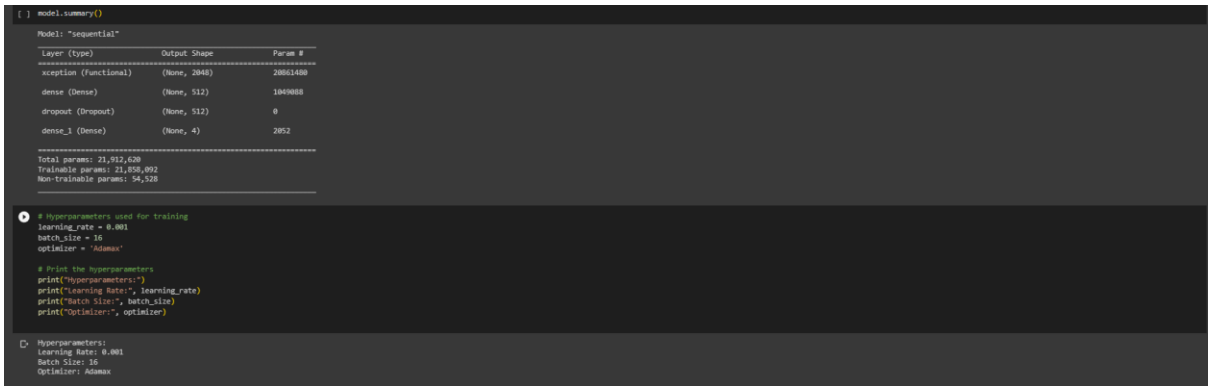

```
classes = train_df['label'].unique()

# Add the import statement for precision_score, recall_score, f1_score, roc_auc_score, and label_binarize
from sklearn.metrics import precision_score, recall_score, f1_score, roc_auc_score, roc_curve, auc
from sklearn.preprocessing import label_binarize
import matplotlib.pyplot as plt


# Load the weights from the file, skipping incompatible layers
if os.path.exists('model_weights.h5'):
    base_model.load_weights('model_weights.h5', by_name=True)
    print('Loaded compatible model weights from the previous stage')

# Calculate additional metrics for the current stage
y_pred_prob = model.predict(test_generator)
y_true = test_generator.classes

# Binarize the labels for multiclass ROC AUC calculation
y_true_bin = label_binarize(y_true, classes=np.arange(len(classes)))

# Compute ROC curve and ROC AUC for each class
fpr = dict()
tpr = dict()
roc_auc = dict()
for i in range(len(classes)):
    fpr[i], tpr[i], _ = roc_curve(y_true_bin[:, i], y_pred_prob[:, i])
    roc_auc[i] = auc(fpr[i], tpr[i])

# Compute micro-average ROC curve and ROC AUC
fpr["micro"], tpr["micro"], _ = roc_curve(y_true_bin.ravel(), y_pred_prob.ravel())
roc_auc["micro"] = auc(fpr["micro"], tpr["micro"])

# Plot ROC curves
plt.figure(figsize=(8, 6))
for i in range(len(classes)):
    plt.plot(fpr[i], tpr[i], lw=2, label='ROC curve (area = {:.2f}) for class {}'.format(roc_auc[i], classes[i]))
plt.plot(fpr["micro"], tpr["micro"], lw=2, label='Micro-average ROC curve (area = {:.2f})'.format(roc_auc["micro"]))

plt.plot([0, 1], [0, 1], color='gray', linestyle='--', lw=2)
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC Curve')
plt.legend(loc='lower right')
plt.grid(True)

# Save the ROC AUC curve plot
plt.savefig('roc_auc_curve.png')
```

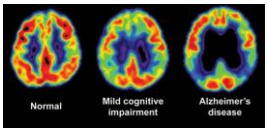

Normal
Mild cognitive impairment
Alzheimer's disease


80/80 [==============================] - 5s 65ms/step
ROC Curve
True Positive Rate
False Positive Rate
ROC curve (area = 0.57) for class NonDemented
ROC curve (area = 0.55) for class VeryMildDemented
ROC curve (area = 0.43) for class MildDemented
ROC curve (area = 0.48) for class ModerateDemented
Micro-average ROC curve (area = 0.30)


import numpy as np
import matplotlib.pyplot as plt
classes = ['MildDemented', 'ModerateDemented', 'NonDemented', 'VeryMildDemented']
# Get predictions and true labels
y_pred = model.predict(test_generator)
y_pred_classes = np.argmax(y_pred, axis=1)
y_true = test_generator.classes
# Find misclassified samples
misclassified_idx = np.where(y_pred_classes != y_true)[0]
# Randomly select a few misclassified samples
num_samples_to_show = 5
selected_idx = np.random.choice(misclassified_idx, num_samples_to_show, replace=False)
# Get the corresponding images and labels
selected_images = [test_generator.filenames[idx] for idx in selected_idx]
selected_true_labels = [classes[y_true[idx]] for idx in selected_idx]
selected_pred_labels = [classes[y_pred_classes[idx]] for idx in selected_idx]
# Set the width of the figure and the subplots
plt.figure(figsize=(15, 6))
plt.subplots_adjust(wspace=0.4)
# Plot the misclassified samples
for i, idx in enumerate(selected_idx):
img_path = os.path.join('/content/Alzheimer_s Dataset/test', selected_images[i])
img = plt.imread(img_path)
plt.subplot(1, num_samples_to_show, i+1)
plt.imshow(img)
plt.title(f'True: {selected_true_labels[i]}\nPredicted: {selected_pred_labels[i]}', fontsize=12)
plt.axis('off')
plt.show()
80/80 [==============================] - 6s 72ms/step
True: VeryMildDemented
Predicted: ModerateDemented
True: NonDemented
Predicted: VeryMildDemented
True: VeryMildDemented
Predicted: ModerateDemented
True: MildDemented
Predicted: VeryMildDemented
True: NonDemented
Predicted: VeryMildDemented


import tensorflow as tf
from tensorflow.keras.utils import plot_model
# Plot the model architecture and save it as an image
tf.keras.utils.plot_model(model, to_file='model_architecture.png', show_shapes=True)
from IPython.display import Image
Image('model_architecture.png')
xception_input | input: | [(None, 224, 224, 3)]
InputLayer | output: | [(None, 224, 224, 3)]
xception | input: | (None, 224, 224, 3)
Functional | output: | (None, 2048)
dense | input: | (None, 2048)
Dense | output: | (None, 512)
dropout | input: | (None, 512)
Dropout | output: | (None, 512)
dense_1 | input: | (None, 512)
Dense | output: | (None, 4)


model.summary()
Model: "sequential"
Layer (type) Output Shape Param #
xception (Functional) (None, 2048) 20861480
dense (Dense) (None, 512) 1049088
dropout (Dropout) (None, 512) 0
dense_1 (Dense) (None, 4) 2052
Total params: 21,912,620
Trainable params: 21,858,092
Non-trainable params: 54,528
# Hyperparameters used for training
learning_rate = 0.001
batch_size = 16
optimizer = 'Adamax'
# Print the hyperparameters
print("Hyperparameters:")
print("Learning Rate:", learning_rate)
print("Batch Size:", batch_size)
print("Optimizer:", optimizer)
Hyperparameters:
Learning Rate: 0.001
Batch Size: 16
Optimizer: Adamax



10.REFERENCES

1. Afzal, S. et al., 2019. A Data Augmentation-Based Framework to Handle Class Imbalance Problem for Alzheimer's Stage Detection. *IEEE.*

2. Al-Shoukry, S., Rassem, T. H. & Makbol, N. M., 2020. Alzheimer's Diseases Detection by Using Deep Learning Algorithms: A Mini-Review. *IEEE,* pp. 77131 - 77141.

3. Basher, A., Kim, B. C., Lee, K. H. & Jung, H. Y., 2021. Volumetric Feature-Based Alzheimer's Disease Diagnosis From sMRI Data Using a Convolutional Neural Network and a Deep Neural Network. *IEEE.*

4. Bijen Khagi, G.-R. K., 2020. 3D CNN Design for the Classification of Alzheimer's Disease Using Brain MRI and PET. *IEEE.*

5. Castelvecchi, D., 2016. Can we open the black box of AI?. *semantic scholar,* p. Davide Castelvecchi.

6. Chima S Eke, E. J. X. L. C. C. S. P. E. I., 2021. Early Detection of Alzheimer's Disease with Blood Plasma Proteins Using Support Vector Machines. *IEEE,* pp. 218-226.

7. Connor Shorten, T. K., 2019. A survey on Image Data Augmentation for Deep Learning. *JOURNAL OF BIG DATA.*

8. Daoqiang Zhang, Y. W. L. Z. H. Y. D. S., 2011. Multimodal classification of Alzheimer's disease and mild cognitive impairment. *NeuroImage,* 55(3), pp. 856-867.

9.Francisco J. Martinez-Murcia, A. O. ,. J.-M. G. ,. J. R. ,., 2020. Studying the Manifold Structure of Alzheimer's Disease: A Deep Learning Approach Using Convolutional Autoencoders. *IEEE.*

10. Hong, X. et al., 2019. Predicting Alzheimer's Disease Using LSTM. *IEEE,* pp. 80893 - 80901.

11. Hongyoon Choi, M. P., 2020. Predicting Cognitive Decline with Deep Learning of Brain Metabolism and. *Alzheimer's Disease Nueroimaging initiative,* Volume 365, pp. 112-119.

12. Ian Goodfellow, Y. B. a. A. C. D. l., 2016. DEEP LEARNING. *The MIT Press,* p. 800.

13. Jadhav , S. M., 2019. Deep convolutional neural network based medical image classification for disease diagnosis. *Journal of Big Data.*

14. Krizhevsky, A. S. I. &. H. G. E., 2017. ImageNet classification with deep convolutional neural networks. *Communications of the ACM,* 60(6), pp. 84-90.

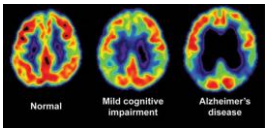
15. Liu, S. et al., 2014. Early diagnosis of Alzheimer's disease with deep learning. *IEEE.*

16. Mitchell, A. J., October 2018. "A Review of Mixed Methods, Pragmatism and Abduction Techniques" has now been published in The Electronic Journal of Business Research Methods, Volume 16 Issue 3. *researchgate,* 16(3), pp. 103-111.

17. Murugan, S. et al., 2021. DEMNET: A Deep Learning Model for Early Diagnosis of Alzheimer Diseases and Dementia From MR Images. *IEEE,* pp. 90319 - 90329.

18. N. V. Chawla, K. W. B. L. O. H. W. P. K., 2011. SMOTE: Synthetic Minority Over-sampling Technique. *Journal Of Artificial Intelligence Research,* Volume 16, pp. 321-357.

19. Qian, H. & Jun Zhou, B. D., 2018. Activation functions and their characteristics in deep neural networks. *IEEE.*



20. Sebastian Bach, A. B. ,. M. K.-R. M. ,. S., 2015. On Pixel-Wise Explanations for Non-Linear Classifier Decisions by Layer-Wise Relevance Propagation. *plus one.*

21. Shelhamer, E., Long, J. & Darrell, T., 24 may 2016. Fully Convolutional Networks for Semantic Segmentation. *IEEE Xplore ,* 39(4), pp. 640-651.

22. Shui-Hua Wang, P. P. Y. S. B. L. M. Y. &. H. C., 2018. Classification of Alzheimer's Disease Based on Eight-Layer Convolutional Neural Network with Leaky Rectified Linear Unit and Max Pooling. *Journal of Medical Systems .*

23. USC, K. S. o. M. o., n.d. *ADNI.* [Online]
Available at: https://ida.loni.usc.edu/login.jsp?project=ADNI

24. Xue, Y., 2020. *research gate.* [Online]
Available at: https://www.researchgate.net/figure/The-network-architecture-of-pCNNs-A-single-CNN-model-contained-11-layers-including-one_fig2_340461296

25. Yiming Ding 1, J. H. S. 1. M. G. K. 1. H. T. 1. R. H. 1. N. W. J. 1. D. L. 1. T. P. C. 1. M. S. A. 1. C. M. A. 1. S. C. B. 1. R. R. F. 1. S.-Y. H. 1. K. A., 2019. A Deep Learning Model to Predict a Diagnosis of Alzheimer Disease by Using 18F-FDG PET of the Brain. *National library of medicines,* pp. 456-464.