A Deep Learning Approach for The Detection of Alzheimer Disease

A project report submitted in partial fulfillment of the requirement for the Award of the Degree of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

by

Santosh Naga Manideep Bhonagiri (160719733002)

Tauseef Ali Khan (160719733003)

Dhruv Teja Manjrekar (160719733023)

Under the Guidance of

Mr. A.A.R Senthil Kumar, Assistant Professor, Dept. of CSE

Department of Computer Science and Engineering

Methodist College of Engineering and Technology, King Koti, Abids, Hyderabad-500001.

2022-2023

A Deep Learning Approach for The Detection of Alzheimer Disease

A project report submitted in partial fulfillment of the requirement for the Award of the Degree of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

by

Santosh Naga Manideep Bhonagiri (160719733002)

Tauseef Ali Khan (160719733003)

Dhruv Teja Manjrekar (160719733023)

Under the Guidance of

Mr. A.A.R Senthil Kumar, Assistant Professor, Dept. of CSE

Department of Computer Science and Engineering

Methodist College of Engineering and Technology, King Koti, Abids, Hyderabad-500001.

2022-2023

Methodist College of Engineering and Technology,
King Koti, Abids, Hyderabad-500001.
Department of Computer Science and Engineering

DECLARATION BY THE CANDIDATES

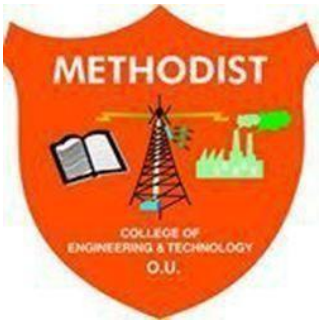
We, **Santosh Naga Manideep Bhonagiri (160719733002), Tauseef Ali Khan(160719733003)** and **Dhruv Teja Manjrekar (160719733023)** students of Methodist College of Engineering and Technology, pursuing Bachelor's degree in Computer Science and Engineering, hereby declare that this Project report entitled "**A Deep Learning Approach for The Detection of Alzheimer Disease",** carried out under the guidance of **Mr. A.A.R Senthil Kumar** submitted in partial fulfilment of the requirements for the degree of Bachelor of Engineering in Computer Science. This is a record work carried out by us and the results embodied in this report have not been reproduced/copied from any source.

Santosh Naga Manideep Bhonagiri (160719733002)

Tauseef Ali Khan (160719733003)

Dhruv Teja Manjrekar (160719733023)

Methodist College of Engineering and Technology,
King Koti, Abids, Hyderabad-500001.
Department of Computer Science and Engineering



CERTIFICATE BY THE SUPERVISOR

This is to certify that this Major project report entitled "**A Deep Learning Approach for The Detection of Alzheimer Disease",** being submitted *by* **Santosh Naga Manideep Bhonagiri (160719733002), Tauseef Ali Khan(160719733003)** and **Dhruv Teja Manjrekar (160719733023)** submitted in partial fulfilment of the requirements for the degree of Bachelor of Engineering in Computer Science and Engineering, during the academic year 2022-2023, is a bonafide record of work carried out by them.

Mr. A.A.R Senthil Kumar
Assistant Professor
Dept. of CSE

Date

Methodist College of Engineering and Technology,
King Koti, Abids , Hyderabad-500001.
Department of Computer Science and Engineering

CERTIFICATE BY HEAD OF THE DEPARTMENT

This is to certify that this Major Project entitled "**A Deep Learning Approach For The Detection Of Alzheimer Disease",** by **Santosh Naga Manideep Bhonagiri (160719733002), Tauseef Ali Khan(160719733003)** and **Dhruv Teja Manjrekar (160719733023)** submitted in partial fulfilment of the requirements for the degree of Bachelor of Engineering in Computer Science and Engineering of the Osmania University, Hyderabad, during the academic year 2022-2023, is a bonafide record of work carried out by them.

Dr. P. Lavanya

Professor & HOD
Dept. of CSE

Date:

Methodist College of Engineering and Technology,
King Koti, Abids, Hyderabad-500001.
Department of Computer Science and Engineering

PROJECT APPROVAL CERTIFICATE

This is to certify that this Major Project entitled "**A Deep Learning Approach for The Detection of Alzheimer Disease",** by **Santosh Naga Manideep Bhonagiri (160719733002), Tauseef Ali Khan (160719733003)** and **Dhruv Teja Manjrekar (160719733023)** submitted in partial fulfilment of the requirements for the degree of Bachelor of Engineering in Computer Science and Engineering of the Osmania University, Hyderabad, during the academic year 2022-2023, is a bonafide record of work carried out by them.

ACKNOWLEDGEMNTS

We would like to express our sincere gratitude to our Project guide **Mr. A.A.R Senthil Kumar, Assistant Professor, CSE,** for giving us the opportunity to work on this topic. It would never be possible for us to take this project to this level without his innovative ideas and his relentless support and encouragement.

We would like to thank our project coordinator **Mr. T. Praveen Kumar, Assistant Professor, CSE,** who helped us by being an example of high vision and pushing towards greater limits of achievement.

Our sincere thanks to **Dr. P. Lavanya, Professor** and **Head of the Department of Computer Science and Engineering,** for her valuable guidance and encouragement which has played a major role in the completion of the project and for helping us by being an example of high vision and pushing towards greater limits of achievement.

We would like to express a deep sense of gratitude towards the **Dr. Prabhu G. Benakop, Principal, Methodist College of Engineering and Technology,** for always being an inspiration and for always encouraging us in every possible way.

We would like to express a deep sense of gratitude towards the **Dr. Lakshmipathi Rao, Director, Methodist College of Engineering and Technology,** for always being an inspiration and for always encouraging us in every possible way.

We are indebted to the Department of Computer Science & Engineering and Methodist College of Engineering and Technology for providing us with all the required facilities to carry our work in a congenial environment. We extend our gratitude to the CSE Department staff for providing us to the need from time to time whenever requested.

We would like to thank our parents for allowing us to realize our potential, all the support they have provided us over the years was the greatest gift anyone has ever given us and also for teaching us the value of hard work and education. Our parents have offered us tremendous support and encouragement, thanks to our parents for all the moral support and the amazing opportunities they have given us over the years.

METHODIST COLLEGE OF ENGINEERING & TECHNOLOGY

Approved by AICTE, Affiliated to Osmania University, - College Code – 1607

Department of Computer Science & Engineering

Vision & Mission

VISION

To become a leader in providing Computer Science & Engineering education with emphasis on knowledge and innovation.

MISSION

M1: To offer flexible programs of study with collaborations to suit industry needs
M2: To provide quality education and training through novel pedagogical practices
M3: **To** Expedite high performance of excellence in teaching, research and innovations.
M4: To impart moral, ethical valued education with social responsibility.

Program Educational Objectives

Graduates of Computer Science and Engineering at Methodist College of Engineering and Technology will be able to:

PEO1: Apply technical concepts, Analyze, Synthesise data to Design and create novel products and solutions for the real life problems.

PEO2: Apply the knowledge of Computer Science Engineering to pursue higher education with due consideration to the environment and society.

PEO3: Promote collaborative learning and spirit of team work through multidisciplinary projects

PEO4: Engage in life-long learning and develop entrepreneurial skills.

Program Specific Outcomes

At the end of 4 years, Computer Science and Engineering graduates at MCET will be able to:

PSO1: Apply the knowledge of Computer Science and Engineering in various domains like networking and data mining to manage projects in multidisciplinary environments.

PSO2: Develop software applications with open-ended programming environments**.**

PSO3: Design and develop solutions by following standard software engineering principles and implement by using suitable programming languages and platform

PROGRAM OUTCOMES

PO1: Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialisation to the solution of complex engineering problems.

PO2: Problem analysis: Identify, formulate, review research literature, and analyse complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

PO3: Design/development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

PO4: Conduct investigations of complex problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

PO5: Modern tool usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modelling to complex engineering activities with an understanding of the limitations.

PO6: The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

PO7: Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

PO8: Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

PO9: Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

PO10: Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

PO11: Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

PO12: Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change

ABSTRACT

Alzheimer's disease is a neurodegenerative disorder that affects millions of people worldwide and is characterised by progressive cognitive decline and memory loss. Alzheimer's disease should be diagnosed early so that proper intervention and management can be started in a timely manner. Deep learning techniques have shown promise in accurately predicting Alzheimer's disease onset based on neuroimaging data and other clinical biomarkers. One such deep learning technique is the DenseNet model for robust and accurate prediction. The DenseNet architecture utilizes a feedforward model to link every layer of the network with every other layer. This creates a "dense" connectivity pattern between layers, hence the name "DenseNet". As part of this project, we aim to develop a deep learning approach that will help us to detect Alzheimer's disease that will be more accurate. The DenseNet model has been examined in detail here. Using a Kaggle dataset which consists of MRI images the data has four classes of images both in training and testing which are "Mild Demented", "Moderate Demented", "Non-Demented", and "Very Mild Demented", and the model has been trained on a DenseNet model to achieve the desired results. The model is a sequential model involving various custom layers such as dense, flatten, batch normalization to optimize and extract the maximum output of the model. During training and testing, the model's accuracy increased significantly. Most of the time, improving the accuracy of one's model results in better performance of the model, which is why certain callback parameters should be implied here, such as checkpoints and early stopping of the model, in order for it to perform better. As a result of the model, they are able to predict the stage of a disease in such a way that they can help detect it at an early stage.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1
INTRODUCTION

1.1. Alzheimer’s Disease

Ancient Greek and Roman philosophers and physicians connected dementia progression to ageing. Alois Alzheimer, a German psychiatrist, did not discover the first instance of the illness that bears his name in a fifty-year-old woman he called Auguste D. until 1901. He followed her case until she passed away in 1906, the year he made his first public report on it. Eleven similar cases, some using "Alzheimer's disease," were reported in the medical literature over the following five years. After suppressing some of the clinical (delusions and hallucinations) and pathological (arteriosclerotic changes) features found in Auguste D.'s initial report, Emil Kraepelin described the illness as a distinct disease. He mentioned Alzheimer's senile dementia by Kraepelin, as a subtype of senile dementia in the eighth edition of his Textbook of Psychiatry, published on 15 July, 1910.

For the 20th century, dementia symptoms in people between 45 and 65 were only diagnosed as Alzheimer's disease. A conference on Alzheimer's disease in 1977 concluded that clinical and pathological symptoms of senile and senile dementia were nearly identical. However, the authors also added that this did not rule out the possibility that they had different causes. This led to a change in terminology. This ultimately resulted in an age-independent Alzheimer's disease diagnosis. For a while, the condition in people over 65 was called senile dementia of the Alzheimer type (SDAT), while those younger were diagnosed with classical Alzheimer's disease.

The most widely used NINCDS-ADRDA Alzheimer's Criteria were developed by the National Institute of Neurological and Communicative Disorders and Stroke (NINCDS) and the Alzheimer's Disease and Related Disorders Association (ADRDA, now known as the Alzheimer's Association) in 1984[229]. They were extensively updated in 2007. For a clinical diagnosis of possibly or probably Alzheimer's disease, these criteria demand that cognitive impairment and a suspected dementia syndrome be confirmed by neuropsychological testing.

Alzheimer's disease (AD), a common form of dementia, is a far-reaching syndrome that causes declines in mental capacity and daily functioning. Memory loss and other critical mental abilities are continuously deteriorated by Alzheimer's disease. A person with Alzheimer's disease may initially experience mild confusion and memory loss as a result of the disease. Alzheimer's disease is one of the main causes of dementia in the modern world and is one of the most prevalent diseases. It ranks as the sixth leading cause of death in the US.

Alzheimer's disease harms the brain, killing brain cells and impaired thinking, memory, and behavior. It is characterized by a decline in cognitive abilities like memory and problem-solving, which has a significant impact.
On daily activities. There is a significant increase in the risk of Alzheimer's disease as you get older. Although there is currently no cure for Alzheimer's disease, if it is found early or in a mild form, the disease's progression can be slowed. A few medications that can halt or slow down the progression of Alzheimer's disease are available, and although there is no known treatment or way to prevent it, there are a few medications that have been found to be effective in treating the condition.

A growing number of people suffer from dementia, which affects about 70% of all Alzheimer's disease cases. Even though this condition is more common than some cancers, it has received much less funding for public health programs and research. There is an urgent need to investigate the best and most effective treatments for Alzheimer's disease in order to find a cure as soon as possible. Given that it eventually results in death, Alzheimer's disease has received a lot of attention in recent scientific research studies. A proper diagnosis of Alzheimer's disease, however, is based on a comprehensive clinical evaluation based on a patient's medical history, neuropsychological testing, and other pathological evaluations in addition to an accurate clinical evaluation.

1.1.1. Symptoms

Alzheimer's disease's primary signs and symptoms include:

1. Recall

Occasionally, we all experience memory loss, but the memory loss associated with Alzheimer's disease is persistent and worsens over time as the disease progresses. There is a gradual decline in one's ability to perform daily tasks at work or at home as a result of memory loss.

- People with Alzheimer's disease might repeatedly repeat questions and statements.
- Don't think about meetings, events, or conversations.

- Items are frequently misplaced and placed in odd places.
- Get lost in areas they used to know.
- Over time, people lose loved ones' names and familiar items.
- Have difficulty expressing ideas, describing things, or participating in conversations.

2. Thinking and reasoning

- Alzheimer's disease causes difficulty concentrating and thinking, especially about abstract concepts such as numbers.
- Doing more than one task at once is especially difficult. It may be challenging to manage finances, balance cheque books and pay bills on time. Eventually, a person with Alzheimer's disease may be unable to recognize and deal with numbers.

3. Making decisions and judgements

Alzheimer's disease deteriorates judgement and decision-making skills. For instance, a person might dress inappropriately for the weather or commit poor decisions in social situations. Some people may find it harder to respond to everyday issues as a result of their mental health condition. For instance, the person might be unable to make decisions while driving or deal with food burning on the stove.

4. Making plans and performing routine tasks

Simple tasks that involve following steps become challenging. This could involve organising and preparing a meal or engaging in a favourite activity. People with advanced Alzheimer's disease eventually lose the ability to perform fundamental tasks like dressing and taking a shower.

5. Changes in personality and behaviour

Brain changes that occur in Alzheimer's disease can affect moods and behaviours. Problems may include the following:

- Depression.
- Loss of interest in activities.
- Social withdrawal.

- Mood swings.
- Distrust in others.
- Anger or aggression.
- Changes in sleeping habits.

1.1.2 Risk factors

The major risk factors that can cause and effect the Alzhemeir’s Disease are:

1.Age

- The single most significant factor is age. After age 65, your chance of Alzheimer's disease doubles every five years.
- However, anyone can be at risk for developing Alzheimer's disease. One in twenty individuals with the condition is under 65.
- People as young as 40 can develop this type of Alzheimer's disease, also known as early- or young-onset Alzheimer's.

2. Family history

- Although the actual increase in risk is minimal, the genes you inherit from your parents may increase your risk of developing Alzheimer's disease.
- However, the risk of passing on the disease is much higher in families where Alzheimer's disease is caused by the inheritance of a single gene.
- If several members of your family have experienced dementia over the years, especially when they were young, you might want to consider genetic counselling. This will enable you to learn more about your risk of Alzheimer's disease in the future.
- More details regarding dementia genetics can be found on the website of the Alzheimer's Society.

3. Down's syndrome

- People with Down's syndrome are at a higher risk of developing Alzheimer's disease.
- This is because the genetic changes that cause Down's syndrome can also cause amyloid plaques to build up in the brain over time, which can lead to Alzheimer's disease in some people.

4. Head injuries

- People who have had a severe head injury may be at higher risk of developing Alzheimer's disease, but much research is still needed in this area.

5. Cardiovascular disease

Research shows that several lifestyle factors and conditions associated with cardiovascular disease can increase the risk of Alzheimer's disease.

These include:

- smoking
- obesity
- diabetes
- high blood pressure
- high cholesterol

6. Other risk factors

These include:

- hearing loss
- untreated depression (though depression can also be one of the symptoms of Alzheimer's disease)
- loneliness or social isolation
- a sedentary lifestyle

1.1.3. Benefits of Early Detection

1. You receive a precise diagnosis so you can prepare yourself.

Dementia-like symptoms can be caused by some treatable conditions. For instance, low vitamin levels, thyroid issues, sleep issues, alcoholism, or depression. Similar to this, impaired vision or hearing are additional potential sources of confusion. Because of this, if you notice any changes in abilities or behaviours, it's crucial to arrange for a thorough medical assessment.

2. You can take a more active role in making personal decisions, such as health decisions.

You have more control over your own healthcare choices and future plans earlier in the disease process.

3. You can employ remedies more successfully

Early intervention in the treatment of Alzheimer's disease and other dementias is typically the key to treatment success. This covers both prescription drugs and complementary therapies.

4. You can concentrate on what matters.

An early diagnosis enables you to make decisions about your priorities, such as when to stop working, when to travel, or whether to pursue new goals.

5. You have the ability to take action.

An early diagnosis enables you to communicate your wishes to your loved ones and make informed decisions about your care, finances, and legal options.

6. You can benefit from these resources.

Programmes offered by your local Alzheimer Society that provide information, support, and education can help you and your family learn how to live well with dementia.

7. Your family will be more than willing to help you.

Your family will be better able to support you and seek the necessary care if they know the disease you are dealing with and the difficulties that come with its progression.

8. You are better equipped to voice your opinion.

You can unite with other dementia sufferers who are speaking out. This will bring attention to the condition, the need for high-quality care, and the need for more funding for research.

9. You can help advance research

You can participate in clinical trials and other research to help improve diagnosis and enhance care.

10. You can help reduce stigma

You can continue to live life to the fullest. Sharing your experience of living with dementia can be very helpful in reducing the stigma of the disease and in encouraging others to reach out for support.

1.2 Introduction to Deep learning

A type of machine learning model called neural networks is motivated by the human brain's structure and operation. They are made up of layers of networked processing nodes (also known as neurons) that recognize relationships and patterns in data. In a neural network, input data is fed through one or more layers of neurons. Each layer applies a set of weights to the input and passes it through an activation function to create an output. Until the final output is produced, the output from one layer is fed as input to the following layer. A process known as training allows a neural network to learn the weights between its neurons. Predictive modelling, speech and image recognition, natural language processing-all can be done using neural networks. They often outperform conventional machine learning models for these tasks because they can learn from complex, high-dimensional data.

Artificial neural networks are used in deep learning, a branch of machine learning, to model and resolve complex issues. It is referred to as "deep" because it makes use of deep neural networks, which have numerous hidden layers and are capable of learning to identify and categories patterns in massive amounts of data. State-of-the-art performance has been attained using deep learning in a variety of fields, including speech and image recognition, natural language processing, and gaming. Convolutional neural networks, recurrent neural networks, and deep belief networks are examples of popular deep learning techniques.

Deep learning's capacity to automatically learn features from data rather than relying on manually created features or heuristics is one of its key advantages. As a result, it excels at tasks where traditional machine learning techniques might have trouble, like speech and image recognition, where the underlying features can be complicated and challenging to describe.

1.2.1 Densenet Model

The CNN is utilized in a variety of applications, including object detection, speech recognition, driverless cars, earthquake prediction, natural language processing, and image classification. High-level neural networks employ the DenseNet-169 model to boost accuracy declines brought on by vanishing gradients. The model accuracy started to saturate at some point as developers added more layers to CNN models in an effort to increase accuracy. Developers were then delivered from this

problem by the DenseNet-169 model, whose layered architecture makes it simple to train the model with minimal training errors.

The Convolution layer, the Pooling layer, and the fully connected layer are the three layers that make up the CNN model.

The convolution layer, from which CNN got its name Convolution Neural Network, is the first layer of the DenseNet model. DenseNet-169 was chosen because despite having a depth of 169 layers it is relatively low in parameters compared to other models, and the architecture handles the vanish gradient problem well.

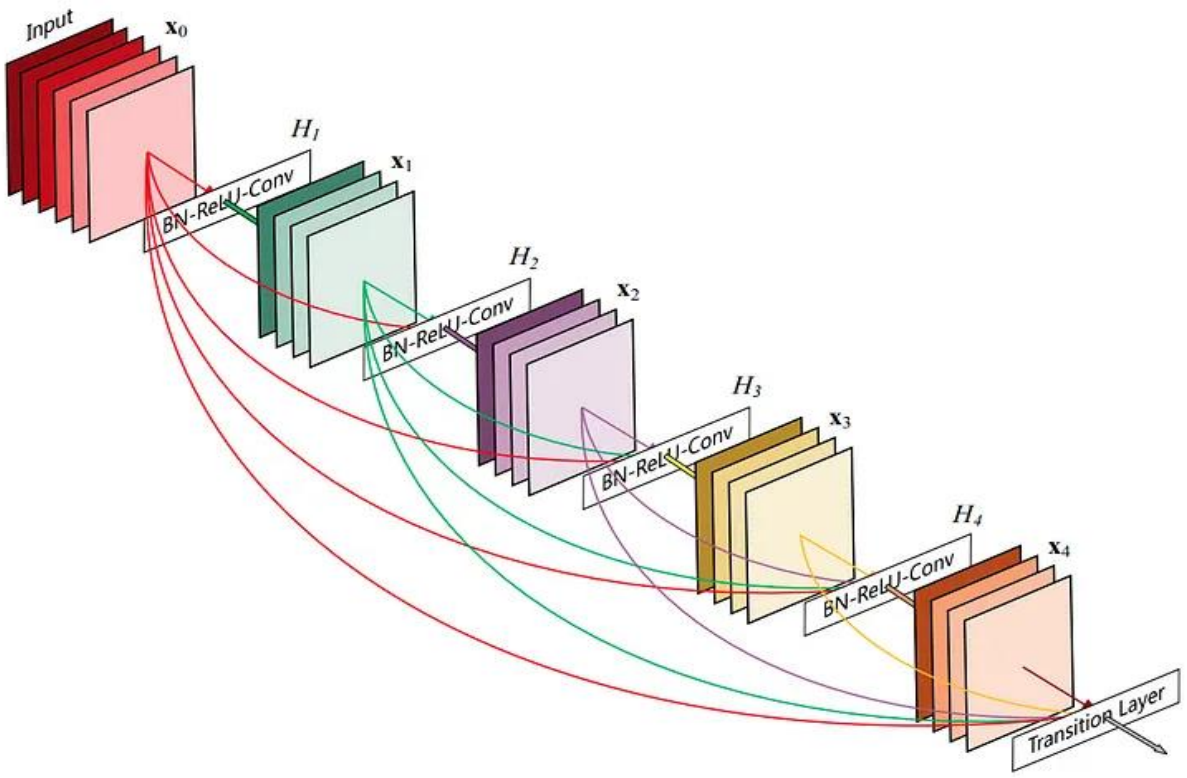


Fig 1.1: Architecture of DenseNet

1.3 Dataset Description

About the Dataset:

In Alzheimer's disease, memory and other mental functions are destroyed over time, as the disease progresses. Eventually, the connections between the brain cells degenerate and die, and memory and other mental functions that are important to the functioning of the brain are lost.

The main symptoms of Alzheimer's disease are memory loss and confusion. There is no cure for this disease, but medications and management strategies may be able to temporarily alleviate the symptoms.

The data consists of MRI images. The data has four classes of images both in training as well as a testing set.

1.3.1 MRI images

A non-invasive imaging technique called magnetic resonance imaging (MRI) creates three-dimensional, intricate anatomical images. For disease detection, diagnosis, and treatment monitoring, it is frequently employed. Based on cutting-edge technology, it stimulates and detects changes in the rotational axis of protons in the water that makes up living tissues. Protons in the body are forced to align with the magnetic field created by strong magnets used in MRI machines. The protons are stimulated and spin out of equilibrium when a radiofrequency current is pulsed through the patient. This causes them to struggle against the magnetic field. When the radiofrequency field is turned off, the MRI sensors are able to detect the energy released as the protons realign with the magnetic field. The time it takes for the protons to realign with the magnetic field, as well as the amount of energy released, changes depending on the environment and the chemical nature of the molecules. Physicians are able to tell the difference between various types of tissues based on these magnetic properties. To obtain an MRI image, a patient is placed inside a large magnet and must remain very still during the imaging process in order not to blur the image. Contrast agents (often containing the element Gadolinium) may be given to a patient intravenously before or during the MRI to increase the speed at which protons realign with the magnetic field. The faster the protons realign, the brighter the image.

MRI scanners are especially well suited to imaging soft tissues or non-bony parts. They are distinct from computed tomography (CT) in that they do not utilize x-rays' harmful ionizing radiation. Because MRI provides a much clearer image of the brain, spinal cord, and nerves than conventional x-rays and computed tomography (CT), it is frequently used to image knee and shoulder injuries. Aneurysms and tumors can also be identified using MRI, which can also distinguish between white and grey matter in the brain. When frequent imaging is needed for diagnosis or therapy, particularly in the brain, MRI is the imaging modality of choice. This is because it does not use x-rays or other radiation. However, MRI is more expensive than x-ray imaging or CT scanning.

MRI uses a powerful magnetic field even though it does not emit the ionizing radiation present in x-ray and CT imaging. A wheelchair could be thrown across the room by the magnetic field. This extends beyond the machine and exerts strong forces on iron, some steels and other magnetizable objects. Before having an MR scan, patients should let their doctors know if they have any medical device or implant.

1.3.2 Uses of MRI in machine learning

Magnetic resonance imaging (MRI) is a medical imaging technique that uses a strong magnetic field and radio waves to create detailed images of internal body structures. Machine learning (ML) algorithms can be applied to MRI data to extract information and make predictions about various medical conditions. Here are some MRI applications in machine learning:

1. Disease Diagnosis: Machine learning algorithms can analyze MRI images and diagnose various diseases, such as Alzheimer's, Parkinson's, and cancer. By analyzing patterns in the data, ML models can detect early signs of disease and provide accurate diagnoses.

2. Image Segmentation: MRI images can be very complex, with many overlapping structures that need to be separated for accurate diagnosis. Machine learning algorithms can be trained to segment MRI images into individual structures, such as different regions of the brain, for further analysis.

3. Predicting Treatment Response: By analyzing MRI images before and after treatment, machine learning algorithms can predict how patients will respond to different therapies. This can help doctors tailor treatment plans to individual patients and improve outcomes.

4. Brain-Computer Interfaces: MRI can capture brain activity, which can then be analysed using machine learning algorithms to develop brain-computer interfaces. These interfaces allow people to control devices, such as prosthetics or computers, using their thoughts.

5. Medical Image Registration: MRI images from different sources or time points can be difficult to compare, as they may be taken from different angles or with different parameters. Machine learning algorithms can register these images, aligning them in space and time for accurate comparison.

Overall, MRI data provides a rich source of information that can be analysed using machine learning algorithms to improve medical diagnosis, treatment, and research.

1.3.3 Stages of Alzheimer’s Disease

Alzheimer's disease is a progressive neurodegenerative disorder that affects the brain, leading to cognitive and behavioral changes. The stages of Alzheimer's disease are generally classified into four broad categories: early, middle, and late-stage.

1. Preclinical stage (Very Mild Alzheimer's Disease):

Changes in the brain begin years before a person shows any signs of the disease. This time period is called preclinical or very mild Alzheimer disease and it can last for years.

2. Early-stage (Mild Alzheimer's Disease):

In the early stage of Alzheimer's, individuals may experience mild memory loss and have difficulty remembering recent events, names, and details. They may also have trouble planning and organising tasks, find it difficult to find the right words during conversations, and experience changes in mood and personality. However, they can still perform their daily activities independently.

3. Middle-stage (Moderate Alzheimer's Disease):

In the middle stage of Alzheimer's, individuals experience more significant changes in memory, language, and behaviour. They may forget critical details, such as their address or phone number, and have difficulty recognizing familiar people and objects. They may also have trouble with daily living activities, such as bathing, dressing, and grooming, and require assistance with these tasks. In addition, they may experience mood swings, changes in personality, and become agitated or aggressive.

4. Severe-Stage (Non demented Alzheimer's Disease):

In the late stage of Alzheimer's, individuals often depend on others for their care. They may lose communication skills, experience difficulty swallowing, and have limited mobility. They may also experience seizures, infections, and other complications, which can ultimately lead to death.

It's important to note that Alzheimer's disease progresses at different rates for each individual, and the exact stages and symptoms can vary. However, understanding the general progression of the disease can help caregivers and family members prepare for the changes that may occur.

CHAPTER 2

LITERATURE SURVEY

2.1Research Survey

As part of our literature survey phase, we surveyed more than 100 research papers as part of our literature review. In which more than 50 research articles were devoted to information about Introduction to Alzheimer's disease. In approximately 20 research papers, there was an introduction and a description of the classification results, but no description of the methodology was included. According to our literature survey, around 25 research papers have been shortlisted for consideration.

MRI images from Kaggle, ADNI MRI images, OASIS CSV and different private datasets are among the most commonly used datasets for the detection of Alzheimer's disease.

There are different approaches that are used in different research papers, including various kinds of machine learning classifiers, from basic classifiers such as support vector machines and random forest classifiers to decision trees classifiers, XGBoost algorithms and voting classifiers to Neural Network classifiers such as CNNs and DNN algorithms.

In various research papers, various evaluation parameters have been used. These parameters range from Accuracy, Precision, Recall, and AUC, ROC, Sensitivity, Confusion Matrix, Correlation coefficient, Information gain, to Chi-Square Specificity.

Through our literature review, we have obtained a better understanding of what research has been done up to now which will constitute a basis for our project. For instance, we discovered that the major focus during the past decade has been in binary classification between AD, MCI, and HC at a lower accuracy range of 65-70% and a higher accuracy range of 90-94%. In contrast, a smaller amount of focus was placed on categorical classification, which is why we are using Kaggle Datasets to do this analysis.

	Authors	Title	Methodology	Evaluation Parameters	Scope	Result
1	Dan Pan, An Zeng, Longfei Jia,Yin Huang,Tory Frizzell and Xiaowei Song	Early Detection of Alzheimer's Disease Using Magnetic Resonance Imaging: A Novel Approach Combining Convolutional Neural Networks and Ensemble Learning	On ADNI Dataset, A novel deep learning approach that combined CNN and EL and applied it to the most commonly acquired anatomical MRI of the brain. objectives: i.e., classification of AD or MCIc vs. HC, and MCIc vs. MCInc.	Accuracy. standard deviation	It can also be used for detecting other brain diseases such as Parkinson's disease, schizophrenia and severe depression. This is especially useful for identifying candidate neuroimaging biomarkers for other little-known brain disorders, in a data-driven way.	The result revealed an accuracy rate of 0.84 ± 0.05, 0.79 ± 0.04, and 0.62 ± 0.06, respectively, for classifying AD vs. HC, MCIc vs. HC, and MCIc vs. MCInc.
2	C.Kavitha, Vinodini Mani, S.R Sri Vidhya, Osmah Ibrahim, Carlos Andres	Early-Stage Alzheimer's Disease Prediction Using Machine Learning Models.	The Alzheimer disease predictions using various machine learning algorithms such as SVM, RF, DT, XGBoost and Voting classifier to effectively distinguish affected individuals with high degree of efficiency and speed using OASIS and Kaggle dataset-.	Accuracy Precision Recall F1 score confusion matrix Correlation coefficient, Information gain, and Chi-Square.	Alzheimer's disease can be detected in a number of different ways, with different machine learning algorithms and micro-simulation methods. In future work will focus on the extraction and analysis of new features that will be more likely to aid in the detection of Alzheimer's Disease.	Decision tree classifier (80.46%), Random forest classifier(86.92 %), SVM(81.67%), XGBoost (85.92%), Voting classifier (85.12%)

3	Weiming lin, Tong Tong, Qinquan Gao, Di Guo, Xiaofeng Du, Yonngui Yang, Gang Guo, Min Xiao, Min Du, Xiaobo Qu	Convolutional Neural Networks-Based MRI Image Analysis for the Alzheimer's Disease Prediction From Mild Cognitive Impairment.	A framework for predicting MCI to AD using MRI biomarkers, SVM, and other ML algorithms was developed with ADNI data. More morphological data from MRI images was mined using FreeSurfer. This included each region of interest's cortical volume, surface area, average thickness, and std of hickness.	Accuracy Standard deviation. Confidential interval.	MCI subjects were classified into MCI non-converters and MCI converters groups by who converted to AD. But the main disadvantage is that conversion might still happen half a year or even 1 month later. Our future work would investigate the use of CNN in modelling AD progression.	AD/NC Trained with: AD/NC 88.79% 0.61% (0.8862, 0.8897] MCIC/MCInc Trained with: MCIC/MCInc 68.68% 1.63% (0.6821, 0.6914] MCIC/MCInc Trained with: AD/NC 73.04% 1.31% (0.7265, 0.7343)
4	Vasco Sá Diogo, Hugo Alexandre Ferreira & Diana Prata	Early diagnosis of Alzheimer’s disease using machine learning: a multi-diagnostic, generalizable approach	A study performed on structural changes in the brain for early detection of AD. The method for multi-diagnostic classification of HC, MCI, and AD is an ensemble of 3 binary classifiers using L-SVM, RF, EF, LDA, LR, LR-SGD, MCC, CV, SD methods and techniques.	(MCC), AUC, BAC, ROI	The classifiers did not make use of cognitive features or scores, which could be of high predictive value and easily acquired. It is still unclear whether classifier outputs would be correctly interpreted by clinicians without a detailedunderstanding of the statistical behind ML algorithms.	The AUC metric (77.7% CI 95%: 77.2 -- 82.6%) into OR resulting in a large OR of 7.06 (CI 95%: 6.75 -- 11.1), which corresponds to a grade 4.

5	Taeho jo, Kwangsik Nho, Andrew J.Saykin	Deep Learning in Alzheimer's Disease: Diagnostic Classification and Prognostic Prediction Using Neuroimaging Data	Interest has grown rapidly in computer-aided machine learning approaches to integrative analysis. Well-known pattern analysis methods, such as LDA, LPBM, LR, SVM and SVM-RFE, have been used and hold promise for early detection of AD and prediction of AD progression.	Gradient Computation Accuracy	Research on AD using deep learning is shifting toward a model that uses only deep learning algorithms rather than hybrid methods, although methods need to be developed to integrate different formats of data in a deep learning network.	Contains the result of 16 individual papers and their performers, each paper has its unique algorithm or method to derive a better accuracy for detection of Alzheimer's disease.
6	Carol Y cheung, An Ran Ran, Shujun Wang, Victor TT Chan, Kaiser Sham, Saima hilal	A deep learning model for detection of Alzheimer's disease based on retinal photographs: a retrospective, multicentre case-control study	A deep learning algorithm showed consistently accurate performance for differentiating between patients with Alzheimer's disease-dementia and individuals with no dementia. In particular, the performance was similar for differentiating between amyloid positive and amyloid negative people.	Accuracy, Sensitivity, Specificity, AUROC	Alzheimer's disease research might explore any increments in sensitivity and specificity when combining retinal photography with blood-based biomarkers, such as brain amyloid.	the bilateral model had 83·6% (SD 2·5) accuracy, 93·2% (SD 2·2) sensitivity, 82·0% (SD 3·1) specificity, and an AUROC of 0·93 (0·01) for detection of Alzheimer's disease-dementia.

7	Roobaea Alrobaea Seifeddine Mechti, Mariem Haoues, Saeed Rubaiee	Alzheimer's Disease Early Detection Using Machine Learning Techniques	An interesting approach for early detection of AD by working on ADNI Dataset using various machine learning approaches to calculate F1 score, accuracy, precision, recall percentages using SVM, RF, K-NN, DT, LR algorithms.	Accuracy Precision Recall F1 score	Further research work could be conducted to focus on other diseases. It will be also interesting to be restricted on MRI without any handcrafted features.	achieve better performances for the k-nn with 97.55% compared to 84.27%, for the svm with 99.10% compared to 83.15%, and for the rf classifier with 99.43% compared to 85.77%.
8	Muhammed Raees, Vinu Thomas	Automated detection of Alzheimer Disease using Deep Learning in MRI	An automated machine learning tool for Alzheimer's disease prediction using a deep learning algorithm has been successfully designed and implemented in this work. The performance levels of SVM and DNN models were also examined.	Image classification, Transfer learning, MRI, ReLU,DNN architecture	An automated machine learning tool for Alzheimer's disease prediction using a DNN The performance levels of SVM and DNN models were also examined. Deep learning showed a high accuracy level of 80-90% in Alzheimer's disease prediction.	SVM based classifiers had 70-80% accuracy, while DNN models had 80-90% accuracy. DNN Models required more data and a better computing platform can improve accuracy and response time further.

9	Sheng Liu, Arjun V. Masukar, Henry Rusinek, Jingyun Chen, Ben Zhang, Weicheng Zhu,	Generalizable deep learning model for early Alzheimer's disease detection from structural MRIs	A deep learning model 3D convolutional neural network (CNN) for multiclass classification, with an architecture that is specifically optimised for the task of distinguishing CN, MCI, and AD status based on MRIs for AD detection that achieves an (AUC) of 85.12 (84.98–85.26) when distinguishing MCI or mild Alzheimer's dementia in the independent NACC cohort.	Sensitivity Accuracy ROI-based features ROC curve	The forecasting ability of the deep learning model is therefore significantly higher than that of the ROI-volume/thickness based model. Our results suggest that deep-learning models could be effective in forecasting the progression of Alzheimer's disease.	On the ADNI held-out test set, the proposed deep-learning model achieved the following AUCs: 87.59 (95% CI: 87.13–88.05) for CN vs the rest, 62.59 (95% CI: 62.01–63.17) for MCI vs the rest, and 89.21 (95% CI: 88.88–89.54) for AD vs the rest.
10	Hiroki Fuse,Kota Oishi,Norihide Maikusa, Tadanori Fukami	Detection of Alzheimer's Disease with Shape Analysis of MRI Images	The method using brain shape information for classification of healthy subjects and AD patients. A P-type Fourier descriptor was used as shape information, and the lateral ventricle excluding the septum lucidum was analysed.	Accuracy, Sensitivity, selectivity	In the current study, we examined the effectiveness of classification of AD patients and HS using brain shape information. Therefore, the findings suggest that shape information may be more useful for diagnosis compared with conventional volume ratio.	The results accuracy of 87.1%. The accuracy rate of the method exceeds the accuracy of volume information methods, which are widely used conventional evaluation.

11	Hiroki Fuse,Kota Oishi,Norihide Maikusa ,Tadanori Fukami	Classification of Patients with Alzheimer's Disease and Healthy Subjects from MRI Brain Images Using the Existence Probability of Tissue Types	The study examined the effectiveness of classification of patients with Alzheimer's disease (AD) and healthy subjects (HS) based on brain magnetic resonance imaging (MRI) using the existence probability of various tissue types.	Accuracy, Sensitivity,selectivity	Future research aims to improve classification accuracy, try a computational intelligence based approach, and combine MRI and PET images to detect AD at early stages. This is due to functional changes occurring before structural changes.	The performed classification of AD patients and HS, although nine CPCs produced the greatest accuracy (77%) on average at four time points, the maximum accuracy was 95% when non-diagonal
12	Michele Donini, Joao M. Monteiro, Massimiliano Pontil, John Shawe-Taylor and Janaina Mourao-Miranda	A Multimodal Multiple Kernel Learning Approach To Alzheimer's Disease Detection	MKL(Multiple Kernel Learning) is an effective approach to combining information from different sources in a high dimensional space using only a small set of examples. It considers each source of information as a kernel to identify which information is discriminative for a specific task.	Balanced Accuracy	Exploiting this new algorithm of EasyMKL, we showed how the selection of the relevant feature, in synergy with the correct trade-off between MR images and clinical information is able to improve the prediction performance for the challenging task Alzheimer's disease versus healthy controls of the ADNI dataset.	Using all the features, we start with an 89% of balanced accuracy for EasyMKL to a 96% for EasyMKLFS. In the second setting, with only 33 clinical variables, EasyMKL obtains 88% and EasyMKLFS 92%.

13	Deepti Pachauri, Chris Hinrichs, Moo K. Chung, Sterling C. Johnson, and Vikas Singh	Topology-Based Kernels With Application to Inference Problems in Alzheimer's Disease	Novel techniques to compute similarity matrices for topologically-based attributed data, using recent developments to characterise signals motivated by their topological features, leading to simple constructions of kernel matrices.	Correlation, Accuracy, AUC	The method enables easy computation of meaningful similarity measures in a non-generative manner for use in machine learning methods, showing possible and scientifically important inferences that can be derived using proposed features, such as AD.	Left Hemisphere accuracy is 75-83% Right Hemisphere accuracy is 75-79%
14	Gabriel Lizarraga, Niovi Rojas, Mercedes Cabrerizo, Malek Adjouadi	A Web Platform for Data Acquisition and Analysis for Alzheimer's Disease	This study introduces a new implementation of a Web Interface and Web Services for the automatic acquisition and processing of data for neurological studies, with a focus on Alzheimer's Disease (AD).	Sensitivity Specificity Accuracy	The system can be easily adapted to process MRIs for epilepsy research, maintaining many of the current features. Any user with basic computer skills can create databases, classify AD, and use the Web Platform. Data mining algorithms can be implemented to extract knowledge from the databases.	Subcortical Volume 78% , 92% , 87% Hippocampus Subfield 72% 90% , 83% Cortical Volume 71% , 92%, 84% Thickness Average (cortical) 74% (92-61) 90%(97-81) 87% (94-79) Surface Area 48% (76-26) 84% (96-61) 71% (79-61)

15	Qi Zhou, Mohammed Goryawala, Mercedes Cabrerizo, Warren Barker, David Loewenstein, Ranjan Duara and Malek Adjouadi	Multivariate Analysis of Structural MRI and PET (FDG and 18FAV-45) for Alzheimer's Disease and Its Prodromal Stages	A multivariate analysis method, orthogonal partial least squares to latent structures (OPLS), was used to discriminate Alzheimer's disease (AD), early and late mild cognitive impairment (EMCI and LMCI) from cognitively normal control (CN) using MRI and PET measures.	mean standard deviation	This study aimed to investigate the predictive power of MRI measures, AV-45 and FDG PET in discriminating AD, LMCI, and EMCI from control using OPLS as a multivariate analysis tool.The model is still considered significant and the results are considered reliable.	The Perfect separation of AD using all features was shown as Fig. 1A, with a high Q 2 of 0.721. It showed high efficiency of separating LMCI and EMCI from CN as well.
16	Mohamed Mahyoub, Martin Randles, Thar Baker, Po Yang	Effective Use of Data Science Toward Early Prediction of Alzheimer's Disease	The paper investigates data for 9 common Alzheimer's Disease risk factors, from three different categories; Medical History, Lifestyle, and Demography.	Sensitivity Specificity Accuracy Precision F1 score AUC	The study aims to improve the accuracy of early diagnosis of Alzheimer's Disease and build a predictive dynamic framework to support Healthcare Professionals with diagnosis decision-making and provide insight into the disease.	During both the training stage and test stage, the five different classifiers were applied consecutively for 30 times for a better accuracy. As expected, the classifiers have performed better during the training stage.

17	Fan Yang, Yuxia Li, Ying Han, Jiehui Jiang	Use of multilayer network modularity and spatiotemporal networkswitching rate to explore changes of functional brainnetworksinAlzheimer's disease	It proposes multilayer network modularity and spatiotemporal network switching rate (stNSR) as novel parameters for fMRI of the brain. Using Pearson correlation, sliding Hamming window, and Louvain algorithm, it is calculated. We explore the viability of stNSR as a potential metric to represent AD patients' dynamic connectivity.	Correlation analysis, P-value	Traditional fMRI brain network metrics investigate brain network complexity. They only present changes in brain networks over a full time quantum, ignoring minor changes over a short period of time, lacking space information. The stNSR method was used to find five ROIs that can be used as future work.	The results showed a significant negative correlation between stNSR and ALFF/DC in AD patients, with the higher the ALFF/DC, the lower the frequency of brain network function switched. This result was reasonable.
18	H. M. Tarek Ullah, Zishan Ahmed Onik, Riashat Islam, Dr. Dip Nandi	Alzheimer's Disease And Dementia Detection From 3D Brain MRI Data Using Deep Convolutional Neural Networks	The paper has an alternative approach that has been discussed, that is fast, costs less and is more reliable. DL represents the true bleeding edge of MI CNN are biologically inspired Multilayer perceptrons specially capable of image processing.	Accuracy, Loss	To reduce the weights of these models, they can be downloaded and chopped off the top layer and replaced with a classification layer. The final layer can then be retrained and all other layers left untouched. This method has been proven effective in many applications.	This model has been trained using Floydhub's GPU. After 545 epochs the model has shown 80.25% accuracy.

19	Sivakani. R and Gufran Ahmad Ansari	Machine Learning Framework for Implementing Alzheimer's Disease	Initially, they did the research for finding the drug for this disease then the focusing turned on analysis and prediction of the disease. Now the research is on prediction in the early stage. In this paper the feature extraction process are performed using the ML algorithm, and then classification is done on the oasis longitudinal dataset.	CC, MAE, RMSE, RAE, RRSE	Our future work enhancement is to be done with this work for improving the performance. Feature extraction and selection is one of the important key factors for the classification. So in future work we need to investigate the feature extraction and selection for getting better classification.	The oasis dataset is processed using EM, best first, Gaussian process, linear regression, and decision stump algorithms to extract features, select features, classify features, and construct trees.
20	Guangyu He, An Ping, Xi Wang, Yufei Zhu	Alzheimer's Disease Diagnosis Model Based on Three-dimensional Full Convolutional DenseNet	proposes paper method of dataset by weighted combination of positive and negative samples and a learning method for small number of samples, and establishes a 3D full CNN classification model, which can not only obtain better image feature information, but also improve the generalisation ability of the model.	Accuracy Recall rate F1 score	This paper proposes an incremental dataset method and a 3D CNN full convolution DenseNet classification model to improve image feature information and generalisation ability. The AD classification model can meet the requirements of practical application.	A dense block CNN model is adopted, and the F1 value of its training results is 0.942. It shows that the application of densely connected CNN further improves the prediction ability of the model.

21	Jonathan Young, Marc Modat, Manuel J Cardoso, John Ashburner and Sebastien Ourselin.	Classification Of Alzheimer's Disease Patients And Controls With Gaussian Processes	Gaussian processes can be applied to structural neuroimaging data to perform classification of Alzheimer's disease subjects in a Bayesian framework, with probabilistic predictions that may be useful in a clinical context while maintaining the same accuracy as a state-of-the-art discriminative classifier.	Sensitivity Specificity Accuracy	This formulation allows automatic variable selection via maximum likelihood, avoiding the effects of overfitting. It reduces the feature dimensionality by focusing on a small region of interest, such as the area around the hippocampus.	The SVM correctly classified 31 out of the 40 test subjects, giving an accuracy of 77.5%. The GP model correctly classified 33 out of the 40 test subjects, equal to an accuracy of 82.5%.
22	Binglin Wang, Wei Li, Wenliang Fan, Xi Chen, Dongrui Wu	Alzheimer's Disease Brain Network Classification Using Improved Transfer Feature Learning with Joint Distribution Adaptation	The joint distribution adaptation method was used to project the source and target domain samples into a new feature space. A classifier was built that focused on the target domain, assigning larger weights to the target domain samples and minimising weighted loss. Experimental results verified the effectiveness of the proposed approach	accuracy	The approach mapped the samples in two domains onto a more consistent feature space and assigned larger weights to the target domain samples. This ensured that the designed classifier worked well in both domains, but emphasised the target domain.	when the number of features is approximately 220, the performance of the classifier is the best, and its accuracy is 80.4%.

2.2 MOTIVATION

There is no simple model to screen for Alzheimer's disease, partly because the diagnosis of Alzheimer's disease itself is complex—typically involving expensive and sometimes invasive tests not commonly available outside highly specialized clinical settings. We aimed to develop a deep learning algorithm that could use MRI images to predict the classification of Alzheimer's Disease. Alzheimer's Disease is a neuro-degenerative disorder which affects brain function and shrinkage of brain cells. It's a progressive disease which makes it much more deadly.

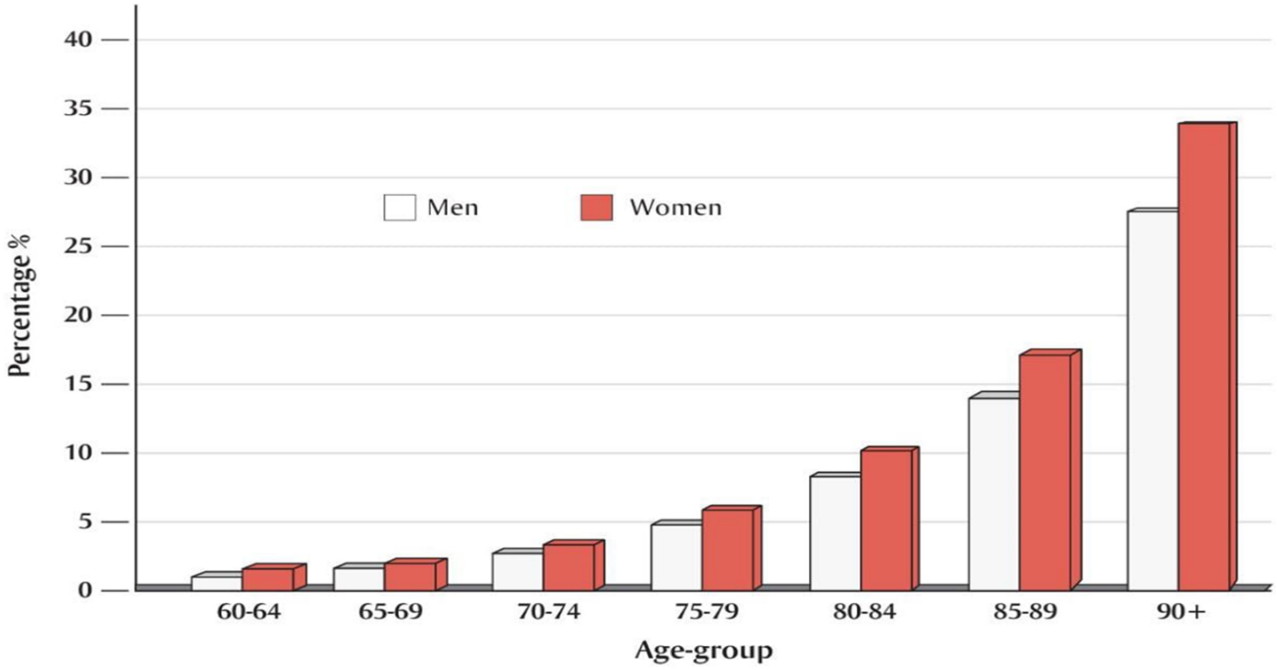


Fig 2.1: Alzheimer's Age-group graph

The above graph depicts the comparison of Alzheimer's disease in both men and women respectively. We can identify the changes in percentages with respect to increase in age. As we can conclude that the Alzheimer's disease increases gradually as age progresses. The graph shows that the women are more likely to get affected by Alzheimer's than men.

The below graph shows the country-wise prevalence of Dementia. In this graph we can observe that China has more peopl affected with dementia than most of the countries. By this data we can conclude that there are many countries that are affected with dementia and are in need of an optimal solution which is faster and reliable to prevent the progression of Dementia.

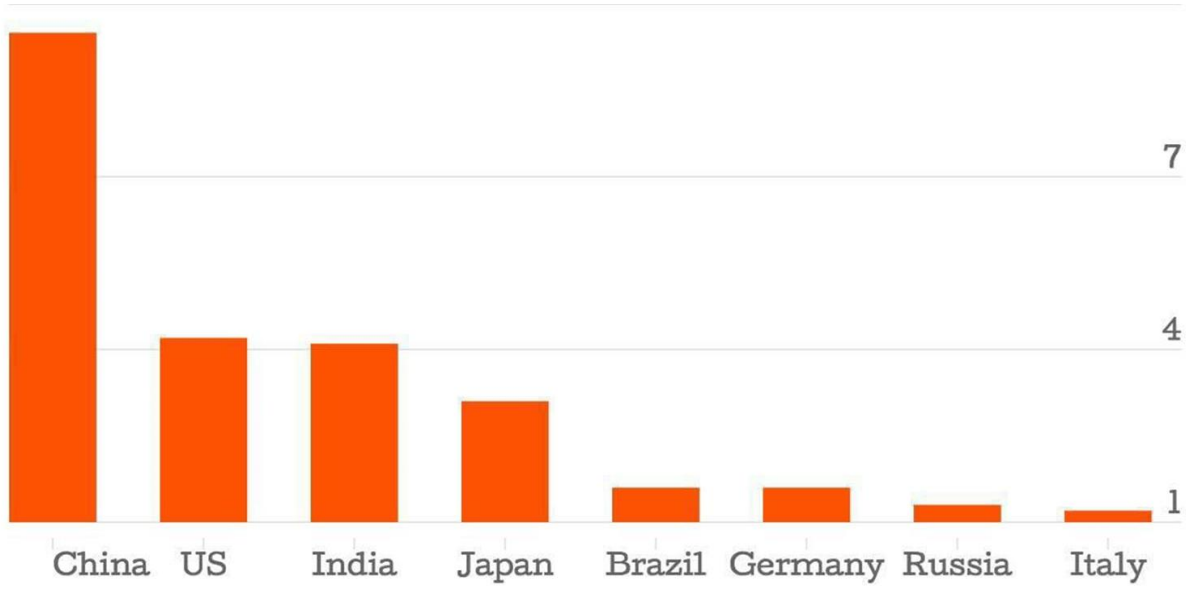


Fig 2.2: Alzheimer's Country Wise graph

There is no permanent cure invented for Alzheimer's Disease. It results in death in its final stage. It's also difficult to detect AD as it is numerous and quite common. As a result, significant research is applied through various government and private organizations to control and cure Alzheimer's Disease.

2.3 OBJECTIVES

The main objectives for our project include:

- Enhance the early detection of Alzheimer's disease and shorten the process to increase the likelihood of treatment success.
- Creating interventions to slow or stop Alzheimer's disease.
- Using a variety of deep learning algorithms in addition to conventional ML techniques.
- It is possible to classify data accurately and robustly.
- The proposed study prioritizes binary detection over categorical classification.
- The focus is on finding people with Alzheimer's disease who are 60 years or older, even though early diagnosis can improve treatment outcomes.
- Unbiased data is more accurate, while most datasets attribute between 40 and 50 percent of Alzheimer's cases to genetic factors.

2.4 Problem Statement

Alzheimer's disease is a progressive neurodegenerative disorder that causes dementia and memory loss. There has been a growing concern and recognition that as the world population ages, Alzheimer's disease will place an enormous burden on one's country's healthcare and economic system. Alzheimer's disease has no cure, and symptoms are common.

But early diagnosis and treatment can slow the disease progression, so there is a need to develop a solution which can detect the Alzheimer's disease at its earliest phase. A Deep Learning method for Early detection of Alzheimer’s disease using Deep Convolution Neural Networks applied with Transfer Learning mechanism is implemented to detect the various stages of Alzheimer's Disease that will help in effective diagnosis.

CHAPTER 3

DESIGN ANALYSIS

3.1 System Architecture

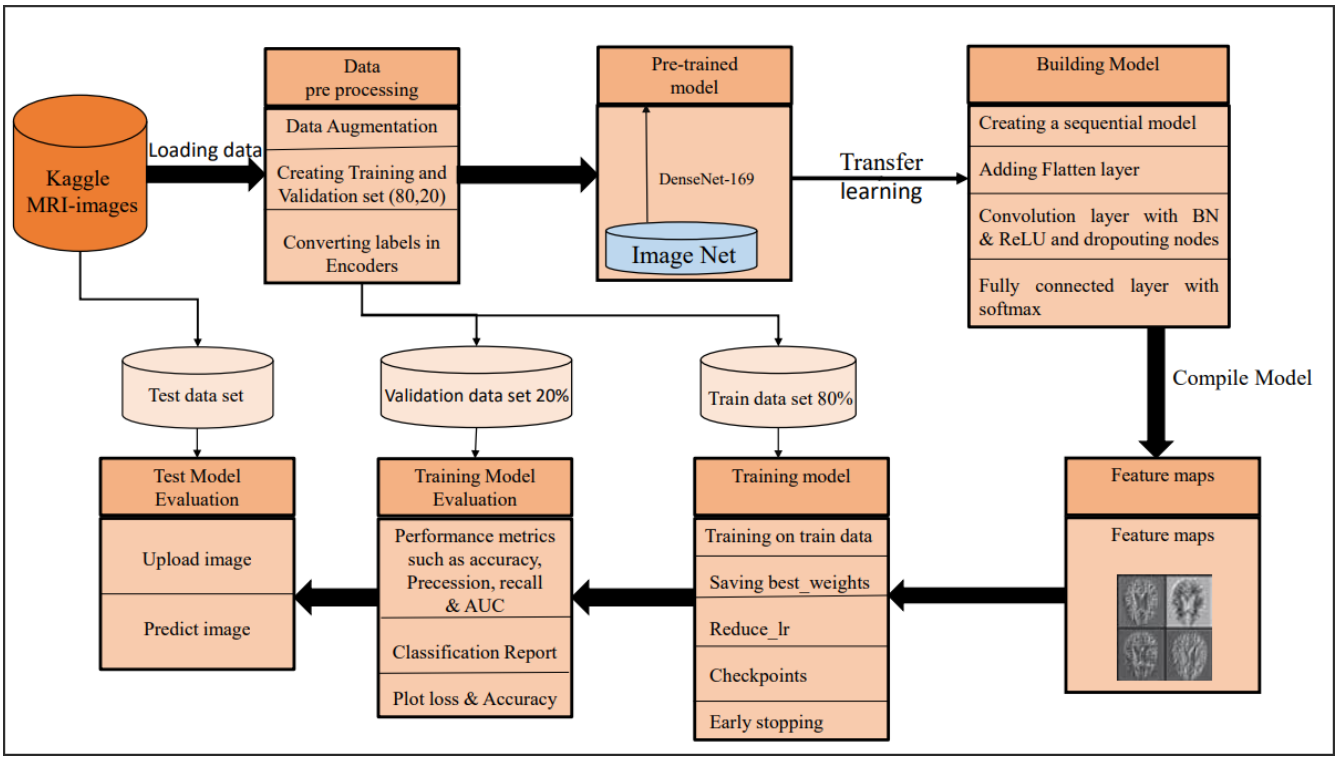


Fig 3.1: System Architecture

The above system architecture depicts our model's skeleton. Using the Kaggle MRI brain image dataset, the flow of the model is shown in a simple manner. In data preprocessing data augmentation has been done and creating a training and validation set of 80:20 then the data is trained on a pre trained model DenseNet 169 in this we create a sequential model and add flatten layers and then compiling the model where feature extraction takes place. Then the model is trained using training data and implying certain call backs on it. The performance metrics used to evaluate train models reveal loss and accuracy. Finally in test model evaluation the image is uploaded and predicted.

3.1.1 Data Collection:

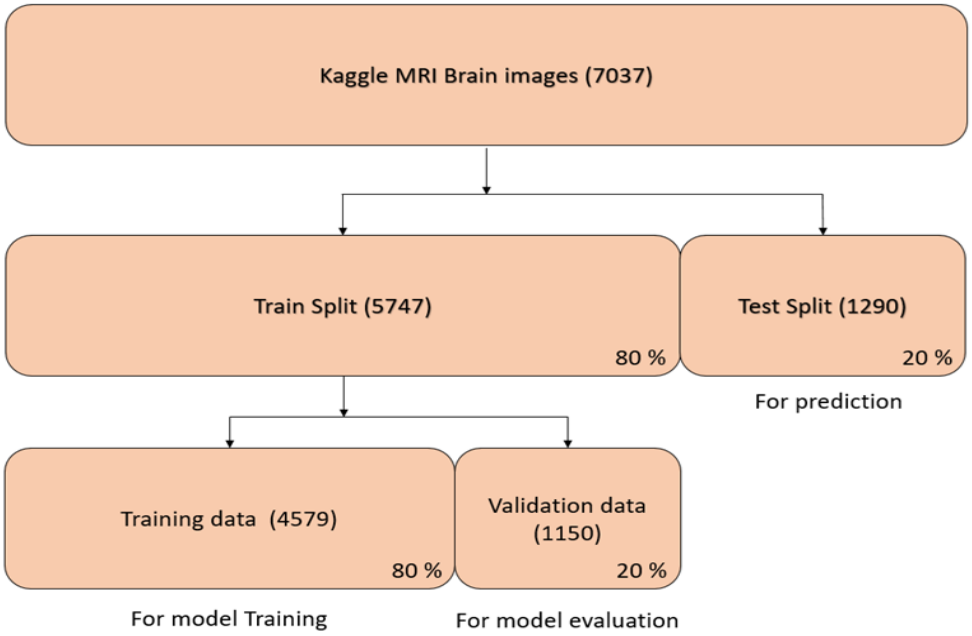


Fig 3.2: Data representation

The dataset used in the project is acquired from the Kaggle database. The original data consists of 6400 MRI images of the brain. By using data augmentation, we have increased the size to 7037 images. From those 7037 images, we have split the images into two parts consisting of train and test data. The train dataset consists of 80% of data i.e, 5747 images and test dataset consist of 20% of the data i.e, 1290 images. The train dataset is further divided into training dataset and validation dataset. The training dataset consists of 80%of train data i.e, 4579 images and validation data consist of 20 % of train data i.e, 1150 images respectively.

Stages	Train data images	Test data images
VeryMildDemeted	1998	448
MildDemented	670	179
ModerateDemented	454	23
NonDemented	2625	640

Table 3.1: Data Description

3.1.2 Data Pre-processing:

Machine learning algorithms require raw data to be transformed into well-formed data sets through data preprocessing. In data preprocessing we have done:

1. Augmentation of data: It helps make the data rich and sufficient and thus makes the model perform better and accurately. Data Augmentation can be achieved by using the existing data to generate new data points.
2. Image to array conversion: It is important to train a machine learning model based on the features of an image. In this conversion of MRI images to arrays for the functioning of keras and numpy libraries.
3. Splitting of dataset: The dataset is splitted into both train and test datasets, also the train dataset is split into both training and validation dataset respectively.
4. Conversion of labels into encoders: Conversion of labels into encoders as categorical data is converted into their own specific numerical values.

3.1.3 Pre-trained Model:

Pretrained models are machine learning models that have been trained on a large dataset by a third party and are available for use in deep learning, NLP, and computer vision applications. We have used a pretrained model of Densenet 169 which is pretrained in the imagenet database. Convolutional neural network (CNN) model DenseNet-169 is a member of the DenseNet family of models. The DenseNet-169 model has 169 layers and employs dense blocks, where each layer is feedforward connected to every other layer, to enhance feature propagation and gradient flow. Additionally, the model has transition layers that limit the spatial dimensionality of the dense block output and manage the expansion of the number of filters. The model is trained on a sizable dataset of labelled images before being used to categories newly created images, which is how DenseNet-169 is typically used for image classification tasks.

The DenseNet 169 model also has an ability to overcome the vanishing gradient problem. DenseNet models can be found in Tensorflow (Keras) and PyTorandes.

3.1.4 Building Model

Initially the model is built by creating a sequential model. This particular sequential model is used to specify the network of the model. This drop outing is responsible for prevention of overfitting as it prevents all neurons in a layer from synchronously optimizing their weights. After drop outing the layers, Batch normalization is applied. Batch normalization is the process of adding additional layers to a deep neural network that allows neural networks to become faster and more stable. On the input of a layer coming from a previous layer, the new layer applies standardizing and normalizing operations.

In the next layer, the convolution layer with RELU, the pooling layer, as well as the fully connected layer, the RELU activation function reduces non-linearity and simplifies the process. This layer aids in the detection of features by reducing the image's nonlinearity and turning negative pixels into zero. Fully-connected layers, also known as linear layers, connect every input neuron to every output neuron and are commonly used in neural networks. With the help of SoftMax activation function the transformation of the raw outputs of the neural network into a vector of probabilities, essentially a probability distribution over the input classes takes place. This is basically used to normalize the outputs.

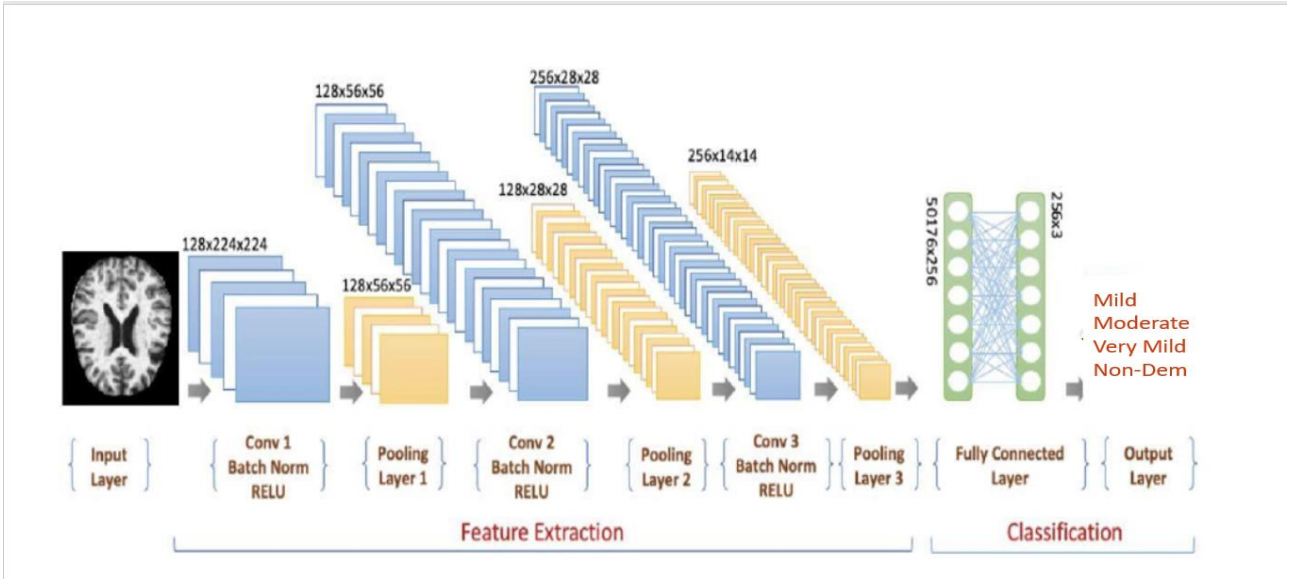


Fig 3.3: Working of DenseNet169

3.1.5 Feature Maps

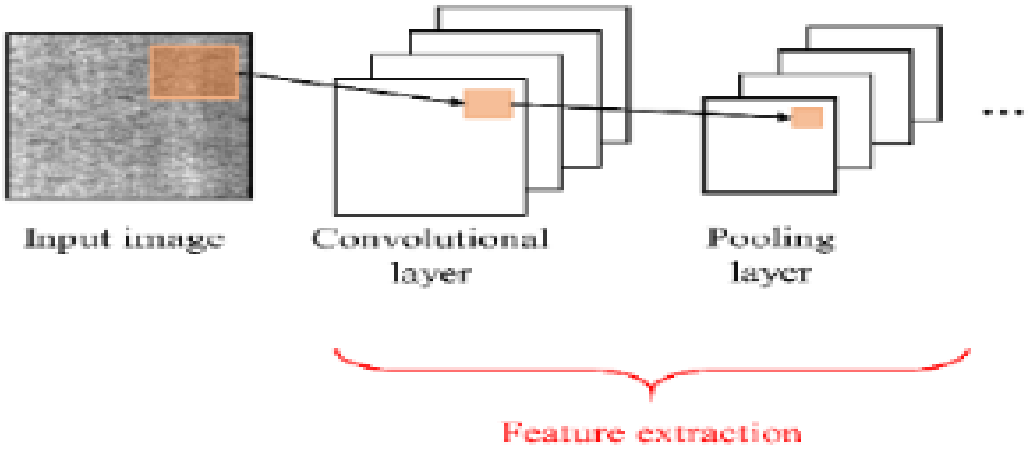


Fig 3.4: Feature extraction process

With the help of the model's convolutional layers, DenseNet-169's feature extraction technique extracts useful and pertinent features from images. The feature extraction is used to retain and acquire necessary features of the image data points. In this feature maps are generated through convolutional layers with help of pooling layers. Max pooling uses the most instances of the feature map, while average pooling uses four numbers from each block multiplied by four and divided by four. This results in a final value of 4, which is updated in the 2D feature map. The feature map generation at each convolutional layer is obtained using a feature detector called a Kernel.

3.1.6 Training model

The model training on train dataset is implemented at this level. While training the model on the train data the required best weights are saved in the process. A special function known as “Callbacks” is introduced. The Callbacks are specialized utilities or functions that are executed at certain steps in the training process. Callbacks can assist you with a number of tasks, such as preventing overfitting, visualizing training results, troubleshooting your programmed, generating logs, creating a Tensor Board, etc. The three callback functions that are used here are:

1. **Checkpoints:** The Checkpoints are mainly used to save weights and improve or can reduce memory usage.

2. Reduce Learning rate: This callback is necessary to reduce the learning rate when required in the model training stages. This helps in improving accuracy during the training process.

3. Early stopping: This function is used to stop the training of the model when the accuracy does not improve through a constant number of epochs. It reduces time and optimizes the performance.

3.1.7 Model Evaluation

The deep learning model is evaluation based on the following metrics:

1. Loss:

Loss is the amount that indicates the total of model errors. It evaluates the model's performance. The loss will be high if errors are high, indicating that the model does not perform well.

2. Accuracy:

Accuracy is a metric that generally describes how the model performs across all classes. It is useful when all classes are of equal importance. It is calculated as the ratio between the number of correct predictions to the total number of predictions.

$$Accuracy = \frac{True_{positive} + True_{negative}}{True_{positive} + True_{negative} + False_{positive} + False_{negative}}$$

3. Precision:

The precision is determined by dividing the number of positive samples by the number of correctly identified samples. Model precision measures how accurately a model performed in classifying a sample as positive.

$$Precision = \frac{True_{positive}}{True_{positive} + False_{positive}}$$

4. Recall:

The recall is determined as the proportion of Positive samples that were correctly identified as Positive to all Positive samples. The recall gauges how well the model can identify positive samples. The more positive samples detected, the higher the recall.

$$Recall = \frac{True_{positive}}{True_{positive} + False_{negative}}$$

5. F1 score:

The harmonic mean of recall and precision is used to calculate the F1 score. If a model's Precision and Recall are both high, it will receive a high F1 score. If a model's Precision and Recall are both low, it will receive a low F1 score.

$$F1\ score = 2 * \frac{Precision\ * Recall}{Precision + Recall}$$

6. AUC:

AUC stands for "Area under the ROC Curve." AUC represents separability level or measurement. It reveals how well the model distinguishes between classes. The model is more accurate at classifying 0 classes as 0, and classifying 1 class as 1, the higher the AUC

3.2 Data flow diagram

A data flow diagram (DFD) visually depicts data flow across a system. It describes the internal workings of a system, the information that flows between its many components, and the entities that interact with it. A data flow diagram (DFD) depicts the movement of data through an information system from the point of data collection to the point of final destination. It is an effective tool for studying and designing information systems since it helps discover and eliminate system inefficiencies and redundancies. DFD's collection of symbols and linkages represents many system components, such as operations, data storage, data flows, and external entities. Normally, the image has several levels, with the top level showing the highest level.

3.2.1 DFD Level 0

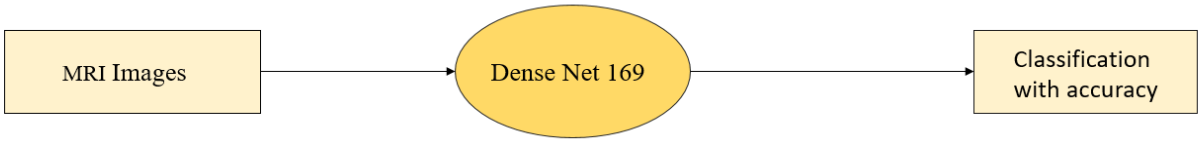


Fig 3.5 DFD Level-0

The data flow diagram level 0 depicts the flow of a dataset of MRI images as input to Dense Net 169 model which in return gives the output that is it classification with accuracy of the image.

3.2.2 DFD Level 1

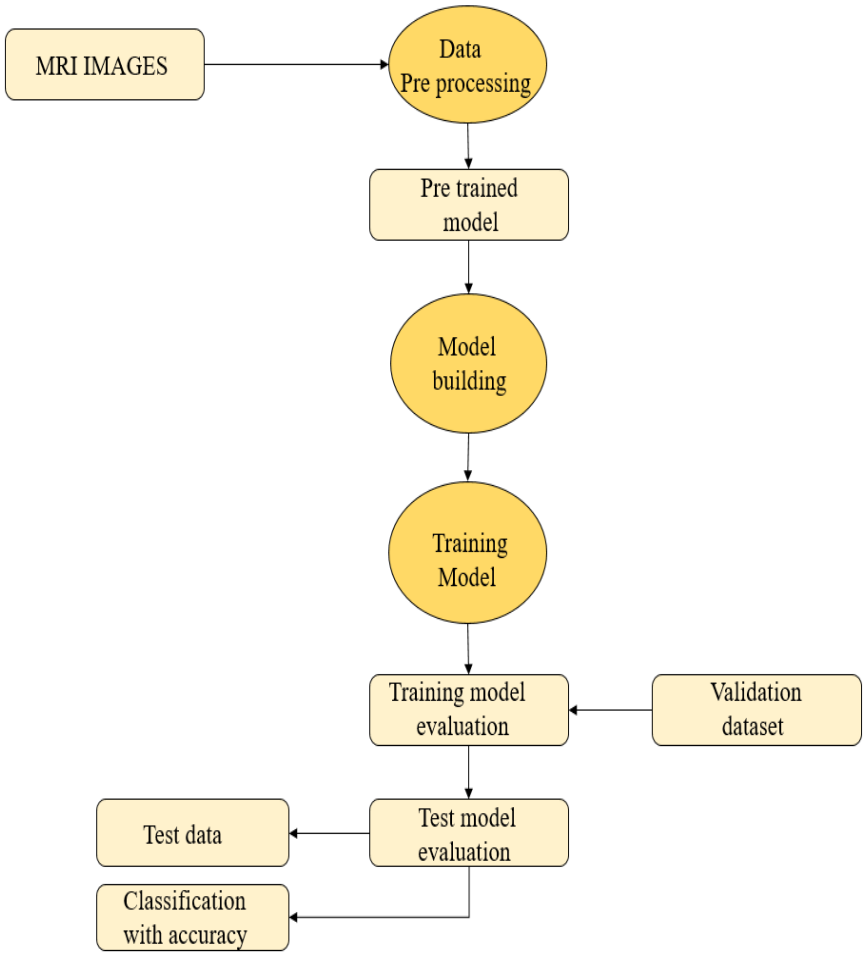


Fig 3.6 DFD Level-1.

Data flow diagram level 1 shows the detailed flow of data, the data set of MRI images is taken as input and passed for data preprocessing process after the preprocessing it is passed to pre trained model in which the data is trained and then model building process is done in which the data is passed through sequential block layers and then the train, test, validation accuracy is evaluated in final step the classification of image is done with certain accuracy.

CHAPTER 4

SYSTEM ANALYSIS

4.1 Model Analysis

Cell: Importing Headers

```
import numpy as np
import os
import matplotlib.pyplot as plt
import tensorflow as tf
import cv2
from tqdm import tqdm
import io
import seaborn as sns
import tensorflow_addons as tfa
import ipywidgets as widgets
from PIL import Image
from sklearn.utils import shuffle # Shuffle arrays or sparse matrices in a consistent way
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score
from skimage.io import imread, imshow
from tensorflow.keras.utils import to_categorical
from tensorflow.keras.layers import InputLayer, BatchNormalization, Dropout, Flatten, Dense,
Activation
from tensorflow.keras.applications import DenseNet169
from tensorflow.keras.models import Sequential
from tensorflow.keras.callbacks import EarlyStopping, ReduceLROnPlateau,ModelCheckpoint
from keras.layers.pooling.max_pooling2d import MaxPooling2D
from keras.utils.vis_utils import plot_model
from IPython.display import display, clear_output
```

Explanation:

This code imports several Python libraries commonly used in machine learning and computer vision tasks. Here's a brief summary of what each library does:

- OS': A module that interacts with the operating system, used here for file and directory operations.

- numpy`: A library for numerical computing in Python, used for manipulating arrays and performing operations on numerical data:
- `matplotlib.pyplot`: A plotting library for Python, used for creating data visualisations.
- 'Tensorflow': A popular open-source machine learning framework developed by Google, used for building and training deep learning models.
- 'cv2': A library for computer vision tasks, used here for reading and manipulating image files.
- 'tqdm': A library for creating progress bars in Python, used here for tracking operations that take a long time to complete.
- IO': A module that handles binary data in Python, used here for reading and writing image data.
- Seaborn': A data visualisation library for Python, used here for creating more advanced visualisations.
- 'ipywidgets': A library for creating interactive widgets in Jupyter notebooks, used here for creating interactive user interfaces.
- 'PIL': A library for working with image files in Python, used here for loading and manipulating images.
- 'sklearn': A machine learning library for Python that provides a range of tools for data analysis and modelling, used here for data preprocessing and evaluation.
- keras': A high-level neural networks API, built on top of TensorFlow, used here for building and training deep learning models.
- 'IPython.display': A module for displaying interactive widgets in Jupyter notebooks, used here for displaying images and other visualizations.

Cell: Loading Data

```
labels = ['MildDemented','ModerateDemented', 'NonDemented', 'VeryMildDemented']
X_train = [] #Training Dataset
Y_train = [] #Training Labels
image_size=224
for i in labels:
    folderPath = os.path.join('/content/drive/MyDrive/Data', 'train', i)
    for j in tqdm(os.listdir(folderPath)):
        image = cv2.imread(os.path.join(folderPath, j))
        image = cv2.resize(image, (image_size, image_size))
        X_train.append(image)
        Y_train.append(i)
```

```
Xtrain = np.array(X_train)  # converted into array
Ytrain = np.array(Y_train)
```

Explanation:

This code reads in images from the specified train directory and resizes them to a square shape of size 224x224 pixels. It also stores the corresponding labels for each image in a separate list. The labels variable is a list of the different label categories for the images. The X_train list stores the training dataset images after they have been read in and resized. The Y_train list stores the corresponding labels for each training dataset image. The image_size variable specifies the desired size of each image after resizing. The for loop iterates over each label category in labels, and then over each image file in the corresponding directory. For each image file, it reads in the image using OpenCV (CV2), resizes it to the desired size, and then appends it to the X_train list. It also appends the corresponding label to the Y_train list. Finally, the X_train and Y_train lists are converted into numpy arrays Xtrain and Ytrain, respectively, for use in machine learning algorithms.

Cell: Shuffling and splitting Train and validation data

```
Xtrain, Ytrain = shuffle(Xtrain, Ytrain, random_state=44)

#train test split which is divided train dataset and validation dataset

xtrain,xtest, Ytrain,ytest = train_test_split(Xtrain, Ytrain,test_size=0.2, random_state=42)
```

Explanation:

The code randomly shuffles the Xtrain and Ytrain arrays along their first axis, which is the dimension of the samples. This is done to ensure that the training data is randomised and that any patterns or order in the data do not interfere with the training process. Next data divides the shuffled Xtrain and Ytrain arrays into a training and a validation set (xtrain, Ytrain). The Scikit-learn library's train_test_split function is used to do the split, with the test_size argument set to 0.2 indicating that 20% of the data should be used for validation.

Cell: Converting labels into encoder

```
ytrain_new = []
ytest_new = []
for i in Ytrain:
    ytrain_new.append(labels.index(i))#Converting String Label to integer i.e
ytrain = to_categorical(ytrain_new) #Converts a class vector (integers) to binary class matrix

for i in ytest:
    ytest_new.append(labels.index(i))
ytest = to_categorical(ytest_new)
```

Explanation:

The provided code transforms string labels to integers and encodes them as binary class matrices. Two empty lists, ytrain_new and ytest_new, are created to hold the converted integer labels. For each element in the Ytrain list, the code finds the corresponding integer index of the label in the labels list using the index() method, and appends it to ytrain_new. Afterward, ytrain_new is converted to a binary class matrix using the to_categorical() function.

The same process is then applied to the ytest list, where the string labels are converted to integer indices and stored in ytest_new. Finally, ytest_new is transformed into a binary class matrix using to_categorical(). This code prepares the labels for machine learning by converting them from strings to integer representations and then encoding them as binary class matrices for further analysis or modelling.

Cell: Building DenseNet169 model

```
model_d =DenseNet169(weights='imagenet',include_top=False, input_shape=(image_size, image_size, 3))
```

Explanation:

This code initialises a DenseNet169 model for image classification. Let's break down the arguments and their meanings:

- ImageNet weights were used to specify that the model should be initialised with pre-trained weights from the ImageNet dataset.
- Fully connected top (classification) layer of the model should not be included which means, the final fully connected layer, which performs the actual classification, is omitted.

- This specifies the input shape of the images that the model expects. `image_size` represents the desired size of the input images, and `3` indicates that the images are in the RGB (Red-Green-Blue) colour format.

Cell: Freezing Layers

```
#Freezing Layers
for layer in model_d.layers:
    layer.trainable=False
```

Explanation:

This code cell iterates over each layer in a pre-defined DenseNet169 model and sets the trainable property of each layer to False. This effectively freezes the weights of the layers, preventing them from being updated during training. This can be useful in transfer learning scenarios where only specific layers or newly added layers need to be trained while keeping the pre-trained layers fixed.

Cell: Custom Layers

```
model=Sequential()
model.add(model_d)
model.add(Flatten())
model.add(BatchNormalization())
model.add(Dense(2048,kernel_initializer='he_uniform'))
model.add(BatchNormalization())
model.add(Activation('relu'))
model.add(Dropout(0.5))
model.add(Dense(1024,kernel_initializer='he_uniform'))
model.add(BatchNormalization())
model.add(Activation('relu'))
model.add(Dropout(0.5))
model.add(Dense(4,activation='softmax'))
```

Explanation:

Sequential model, which allows us to stack layers on top of each other in a linear manner. adding the previously defined `model_d` (DenseNet169) as a layer to the sequential model. Adding a flatten layer to the model. Adding a batch normalisation layer. Batch normalisation is a technique that normalises the

inputs of a layer, helping with training stability and improving generalisation. Adding a fully connected layer with 2048 neurons to the model. The he_uniform sets the initialization method for the weights of this layer using the He uniform initialization. Adding a batch normalisation layer. Adding ReLU activation function to introduce non-linearity after the batch normalisation layer. Adding a dropout layer with a rate of 0.5. Dropout randomly sets a fraction of the input units to 0 during training, which helps prevent overfitting. Adding a fully connected layer with 1024 neurons to the model. The he_uniform sets the initialization method for the weights of this layer using the He uniform initialization.Adding a batch normalisation layer. Adding ReLU activation function to introduce non-linearity after the batch normalisation layer. Adding a dropout layer with a rate of 0.5. Dropout randomly sets a fraction of the input units to 0 during training, which helps prevent overfitting.

Cell: Summary

```
[ ]  model.summary()

Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 densenet169 (Functional)    (None, 7, 7, 1664)        12642880

 flatten (Flatten)           (None, 81536)             0

 batch_normalization (BatchN  (None, 81536)            326144
 ormalization)

 dense (Dense)               (None, 2048)              166987776

 batch_normalization_1 (Batc  (None, 2048)             8192
 hNormalization)

 activation (Activation)     (None, 2048)              0

 dropout (Dropout)           (None, 2048)              0

 dense_1 (Dense)             (None, 1024)              2098176

 batch_normalization_2 (Batc  (None, 1024)             4096
 hNormalization)

 activation_1 (Activation)   (None, 1024)              0

 dropout_1 (Dropout)         (None, 1024)              0

 dense_2 (Dense)             (None, 4)                 4100

=================================================================
Total params: 182,071,364
Trainable params: 169,259,268
Non-trainable params: 12,812,096
_________________________________________________________________
```

Fig 4.1: Model Summary

Explanation:

Shows the summary of the model after adding custom layers providing information such as type of model, custom layers appended into it and and total number of parameters including Trainable and Non-trainable Parameters.

Cell: Metrics for evaluating model

```
METRICS = [
    tf.keras.metrics.CategoricalAccuracy(name='accuracy'),
    tf.keras.metrics.Precision(name='precision'),
    tf.keras.metrics.Recall(name='recall'),
    tf.keras.metrics.AUC(name='auc'),
    tfa.metrics.F1Score(num_classes = 4, name='f1_score')
]
```

Explanation:

1. `tf.keras.metrics.CategoricalAccuracy(name='accuracy')`: This metric calculates the accuracy of the model's predictions by comparing them to the true labels. It is specifically designed for categorical classification problems, where each sample belongs to one and only one class.
2. `tf.keras.metrics.Precision(name='precision')`: Precision measures the proportion of true positive predictions (correctly identified positives) out of all positive predictions. It is a metric commonly used to evaluate the model's ability to correctly identify positive instances.
3. `tf.keras.metrics.Recall(name='recall')`: Recall, also known as sensitivity or true positive rate, measures the proportion of true positive predictions out of all actual positive instances. It is used to evaluate the model's ability to identify all positive instances correctly.
4. `tf.keras.metrics.AUC(name='auc')`: AUC (Area Under the Curve) is a metric that calculates the area under the Receiver Operating Characteristic (ROC) curve. It provides an overall measure of the model's ability to distinguish between different classes by considering the trade-off between true positive rate and false positive rate.
5. `tfa.metrics.F1Score(num_classes=4, name='f1_score')`: F1 Score is a metric that combines precision and recall by calculating their harmonic mean. It provides a single value that represents the balance between precision and recall. The `num_classes` parameter indicates the number of classes in the classification task.

Cell: Compiling Model

```
model.compile( optimizer='adam',
   loss='categorical_crossentropy',
   metrics=METRICS)
```

Explanation:

Adam optimizer is used during training. Adam is an adaptive optimization algorithm commonly used in deep learning that adjusts the learning rate based on the gradient of the loss function.

Categorical_crossentropy defines the loss function to be used during training. Categorical cross-entropy is a common loss function for multi-class classification problems. It measures the dissimilarity between the true labels and the predicted probabilities for each class.

METRICS parameter specifies the metrics to be computed and evaluated during training and evaluation. `METRICS` is a list that contains various metrics such as accuracy, precision, recall, AUC, and F1 score. These metrics will be calculated and monitored to assess the model's performance during training and evaluation.

By calling `model.compile()` with the specified optimizer, loss function, and metrics, the model is prepared for training. This step configures the necessary settings for the model to optimise its parameters using the specified optimizer, compute the loss during training, and track the defined metrics for performance evaluation.

Cell: Defining Callbacks

```
filepath = '/content/drive/MyDrive/weight/best_weights_MRI.hdf5'
earlystopping = EarlyStopping(monitor = 'val_auc',
                              mode = 'max' ,
                              patience = 15,
                              verbose = 1)
checkpoint    = ModelCheckpoint(filepath,
                                monitor = 'val_auc',
                                mode='max',
                                save_best_only=True,
                                verbose = 1)
reduce_lr = ReduceLROnPlateau(monitor='accuracy',
                              factor=0.3,
                              patience=2,verbose=1,
                              mode='auto',
                             min_delta=0.001)
callback_list = [earlystopping, checkpoint, reduce_lr]
```

Explanation:

EarlyStopping callback monitors the validation AUC (Area Under the Curve) during training. It stops the training process if the monitored metric (val_auc) does not improve for `patience` number of epochs. Checkpoint callback saves the model's weights to the specified `file path` whenever the validation AUC improves. The `save_best_only` parameter ensures that only the weights with the best performance on the validation AUC are saved. ReduceLR callback reduces the learning rate (lr) when a metric (accuracy in this case) plateaus. It multiplies the learning rate by the specified `factor` (0.3) to decrease its value. The `patience` parameter defines the number of epochs to wait before reducing the learning rate. The `verbose` parameter enables logging of messages.

By utilising these callbacks, the training process can be monitored, and actions such as early stopping, saving the best model weights, and adjusting the learning rate can be performed based on the specified conditions and metrics.

Cell: Training Model

```
history = model.fit(xtrain, ytrain,
                    batch_size=32,
                    validation_split=0.1,
                    epochs=100,
                    verbose=1,
                    callbacks=[earlystopping, checkpoint, reduce_lr])
```

Explanation:

The training process is initiated by calling `model.fit()` with the specified arguments. The model will be trained using the provided data and configurations, and the training progress and metrics will be logged based on the verbosity level. The training will stop early if the monitored metric does not improve, the best weights will be saved, and the learning rate may be adjusted based on the defined callbacks. The training history, containing information about the loss and metrics at each epoch, will be stored in the `history` variable.

CHAPTER 5
SYSTEM IMPLEMENTATION

5.1 Model Testing

Our model has been tested with untrained data split into 4 classes: Mild Demented, Moderate Demented, Very Mild Demented and Non-Demented. For each test case we tested each class image to predict the outcome and verify the result. Streamlit is a Python application that allows you to load and run a prediction model in order to make a prediction about the image.

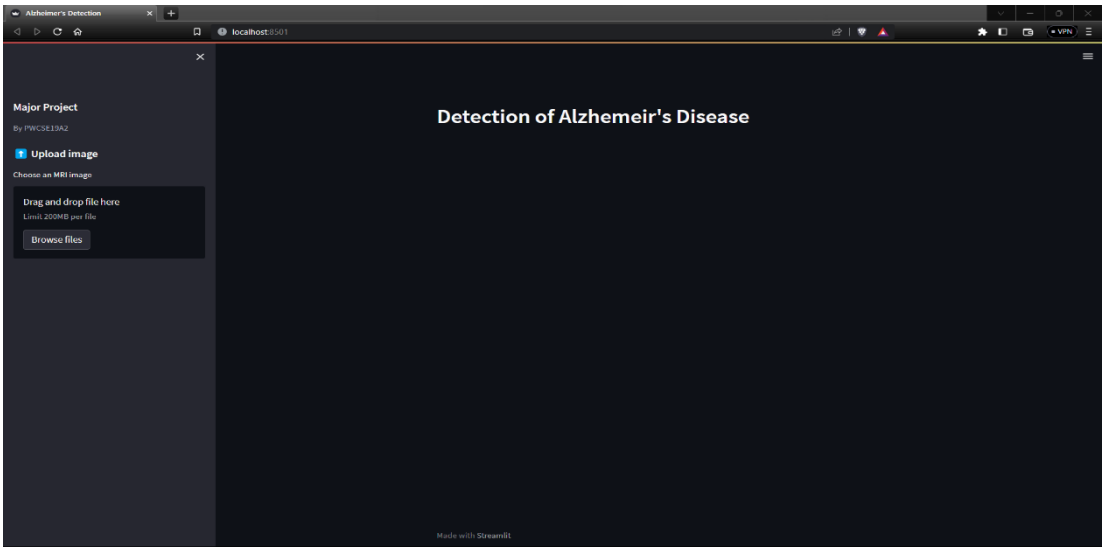


Fig 5.1 home page

5.1.1 Testcase-1: Mild Demented

We supplied mild class images from test data as input for prediction The results of the prediction were then compared to the actual labels and 2 test cases were derived.

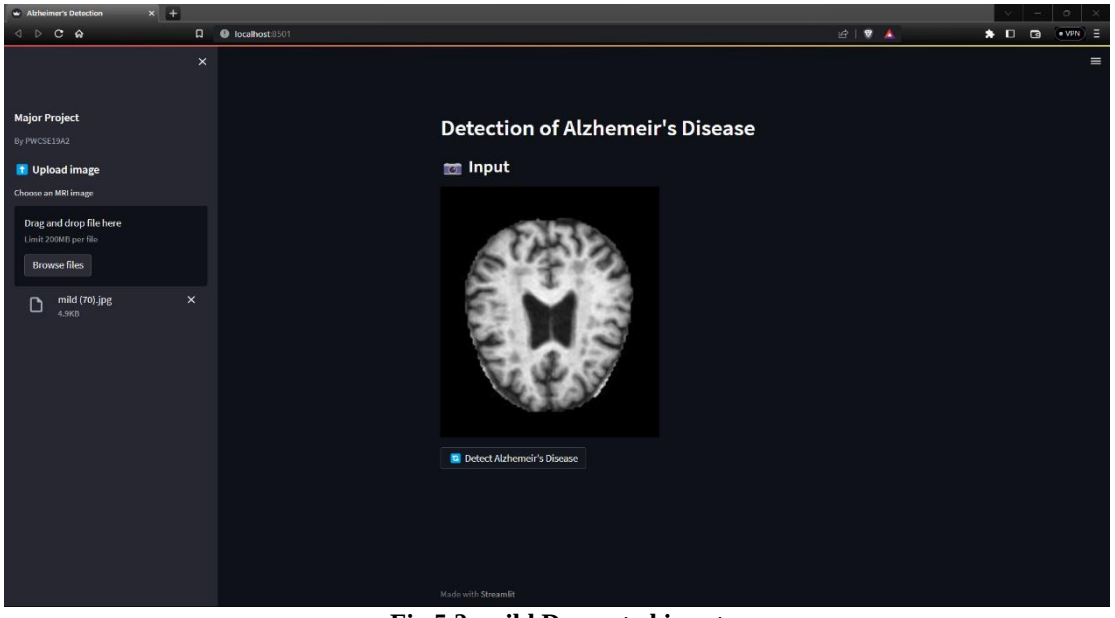


Fig 5.2: mild Demented input

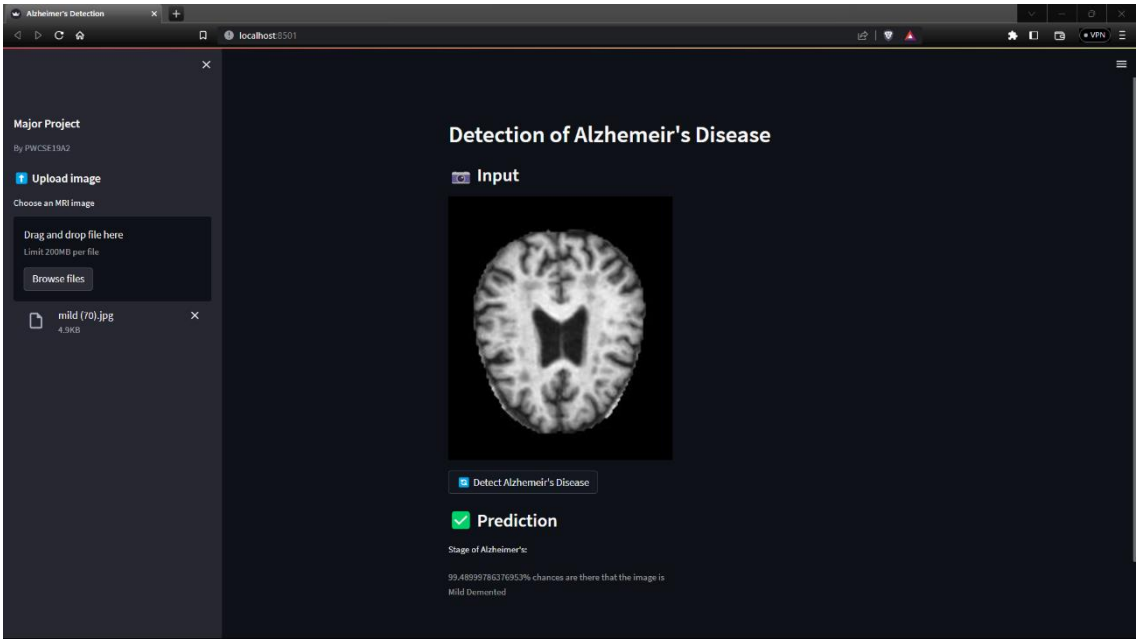


Fig5.3: mild demented output

test_id	Case Description	Expected Result	Prediction	Pass/Fail
mild_1	Prediction for Mild Demented	Mild demented	Mild demented	Pass
mild_2	Prediction for Mild Demented	Mild demented	Very mild demented	Fail
mild_3	Prediction for Mild Demented	Mild demented	Moderate demented	Fail
mild_4	Prediction for Mild Demented	Mild demented	Non-Demented	Fail

Table 5.1: Model testcases for mild Demented

5.1.2 Testcase-2: Moderate Demented

We supplied moderate class images from test data as input for prediction The results of the prediction were then compared to the actual labels and 3 test cases were derived.

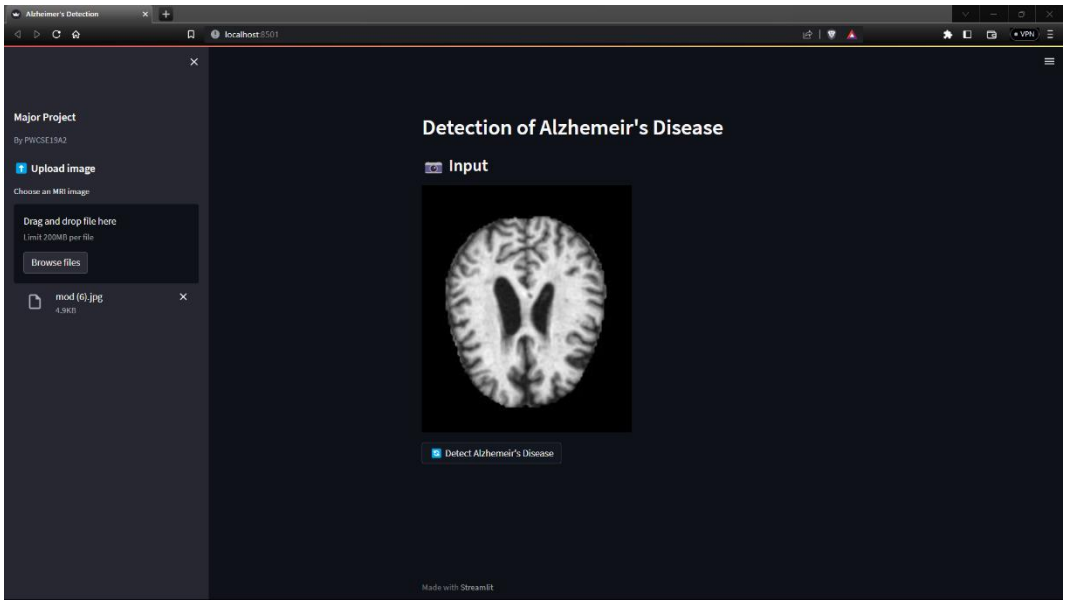


Fig 5.4: moderate Demented input

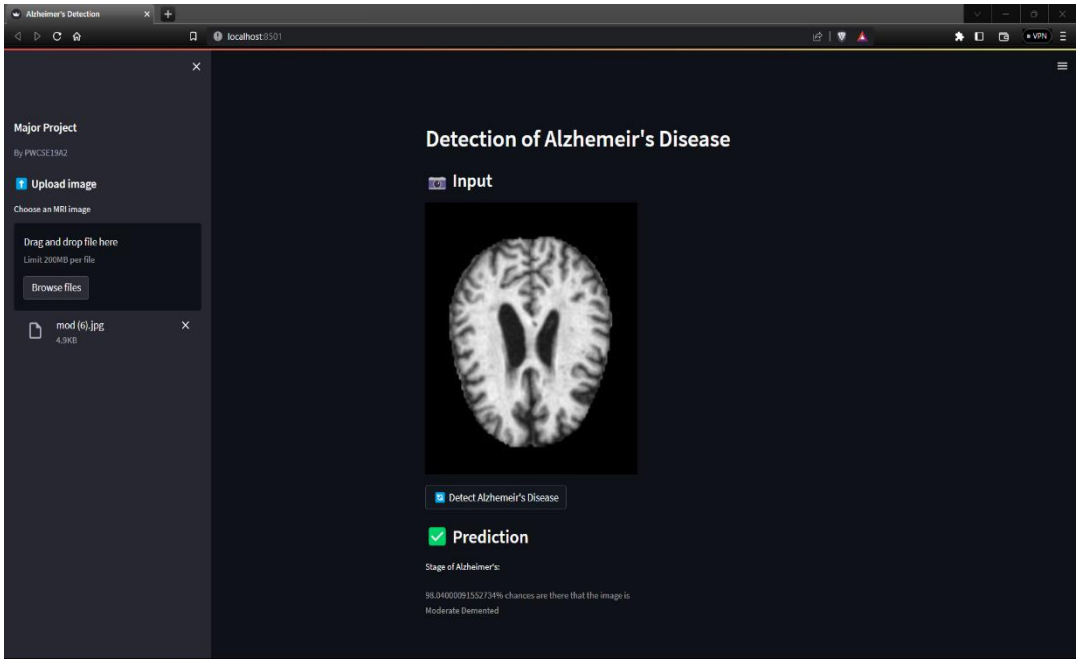


Fig 5.5: moderate Demented output

test_id	Case Description	Expected Result	Prediction	Pass/Fail
moderate_1	Prediction for Moderate Demented	Moderate demented	Moderate demented	Pass
moderate_2	Prediction for Moderate Demented	Moderate demented	Very mild demented	Fail
moderate_3	Prediction for Moderate Demented	Moderate demented	Mild demented	Fail
Moderate_4	Prediction for Moderate Demented	Moderate demented	Non-Demented	Fail

Table 5.2: Model testcases for Moderate Demented

5.1.3 Testcase-3: non-Demented

We supplied Non-Demented class images from test data as input for prediction The results of the prediction were then compared to the actual labels and 3 test cases were derived.

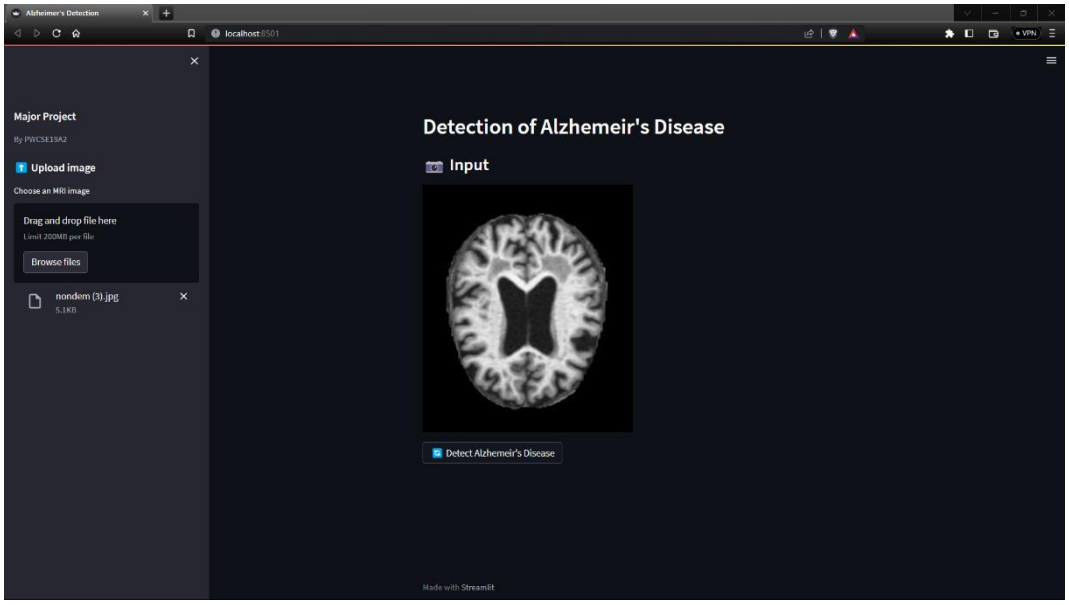


Fig 5.6: Non-Demented input

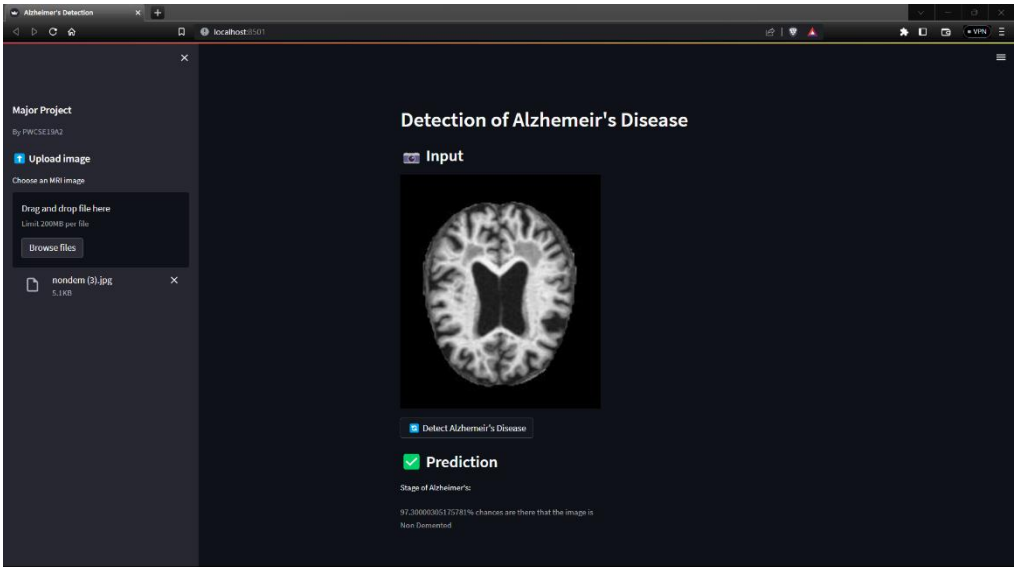


Fig5.7: Non-Demented output

test_id	Case Description	Expected Result	Prediction	Pass/Fail
nondem_1	Prediction for Non-Demented	Non-demented	Non-demented	Pass
nondem_2	Prediction for Non-Demented	Non-demented	Very mild demented	Fail
nondem_3	Prediction for Non-Demented	Non-demented	Mild demented	Fail
nondem_4	Prediction for Non-Demented	Non-demented	Moderate demented	Fail

Table 5.3: testcases for nondemented

5.1.4 Testcase-4: Very Mild Demented

We supplied Very mild Demented class images from test data as input for prediction The results of the prediction were then compared to the actual labels and 3 test cases were derived.

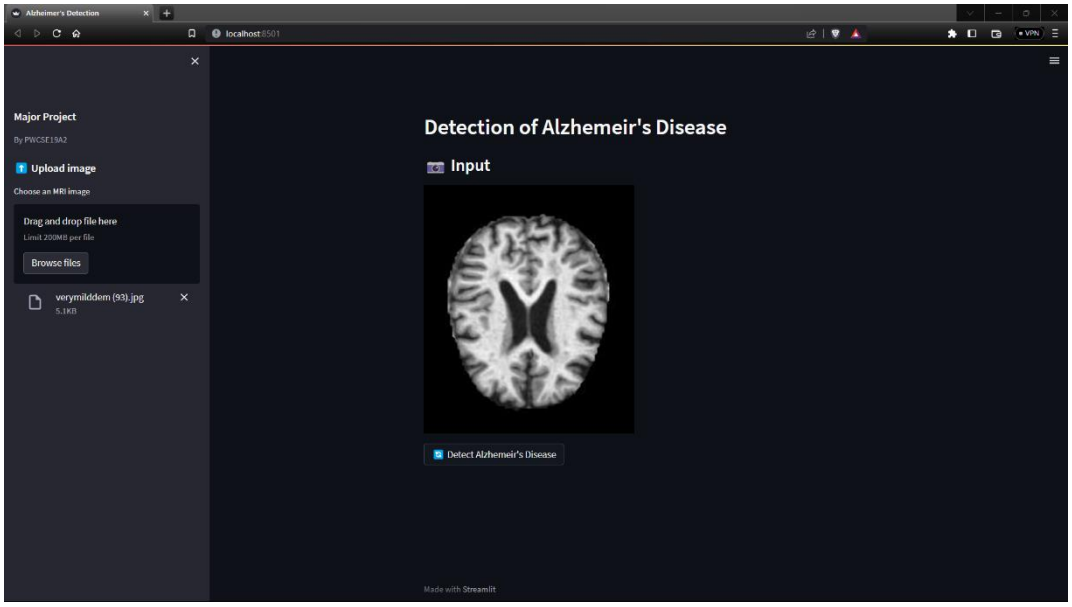


Fig 5.8: Very Mild Demented input

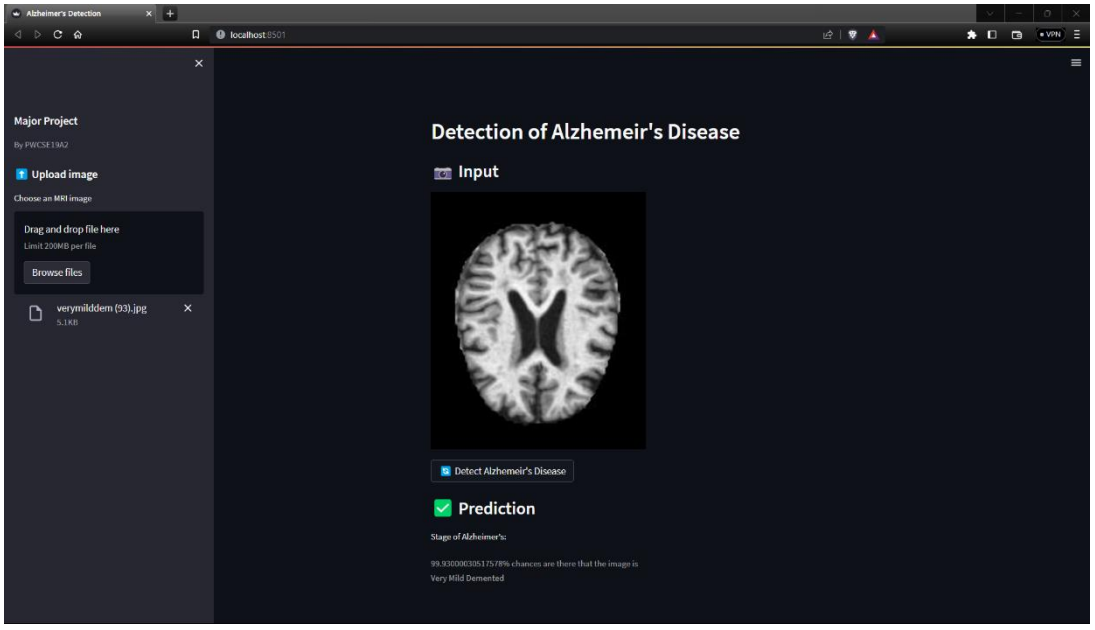


Fig 5.9: Very Mild Demented output

test_id	Case Description	Expected Result	Prediction	Pass/Fail
verymild_1	Prediction for Very Mild Demented	Very Mild demented	Very Mild demented	Pass
verymild_2	Prediction for Very Mild Demented	Very Mild demented	Moderate demented	Fail
verymild_3	Prediction for Very Mild Demented	Very Mild demented	non-demented	Fail
verymild_4	Prediction for Very Mild Demented	Very Mild demented	Mild Demented	Fail

Table 5.4: testcases for very mild demented

CHAPTER 6

RESULT AND ANALYSIS

We evaluated various performance metrics like accuracy, precision, recall and F1 score. To determine the correct predictions for our model we used a confusion matrix. Finally, we compare the accuracy of our model with other Deep Learning Models. Several metrics and techniques were used to identify overfitting and parameter tuning issues after models were created. Performance evaluations are multiclass and described by the confusion matrix. A learning model was developed to detect the stage of Alzheimer's disease which can be Very MildDemented, MildDemented, ModerateDemented and Non-Demented from collected Alzheimer's MRI data. A novel Deep Learning classifier was developed and validated to predict and test on random data. The following evaluation measures were calculated using these components: precision, recall, accuracy, and F-score. Based on this study, recall (sensitivity) is the proportion of people accurately identified as having Alzheimer's. Alzheimer's diagnosis precision is the rate of people correctly classified as not having the disease. Alternatively, F1 represents the weighted average of recall and precision, while accuracy represents the proportion of people correctly classified. According to the results, the patient receives a report that tells him or her what stage of Alzheimer's disease he or she is currently in. It is very critical to detect the stages because the stages are based on the responses of patients. In addition, knowing the stage helps doctors better understand how the disease affects them.

This research used these environments, tools, and libraries to conduct its experiments and analyses:

A) Environment Used: Python 3

B) Tensorflow, Scikit-learn libraries for machine learning

C) Keras API for model building and evaluation

6.1 Graphs

6.1.1 Model loss

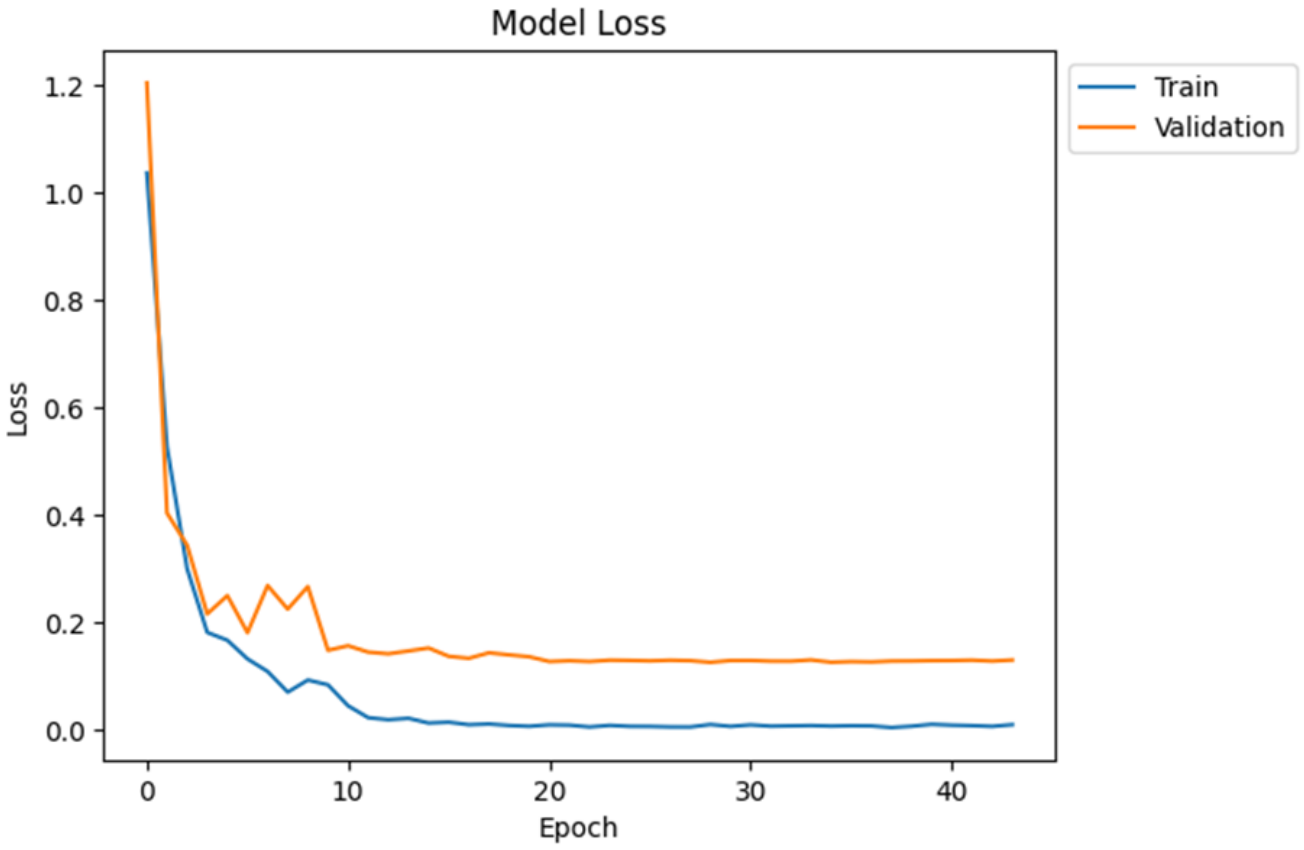


Fig 6.1: Model Loss

The following graph shows the model loss per epoch for the first epoch model loss is highest that is 1.0369 which gradually decreases from 1.0369 to 0.0105 in 44 epochs and for validation highest loss is from 1.2 that gradually decreases from 1.2 to 0.2 in 44 epochs. Model loss is stable from epoch 15 to 44. The graph shows loss for both Training and Validation.

6.1.2 Model Accuracy

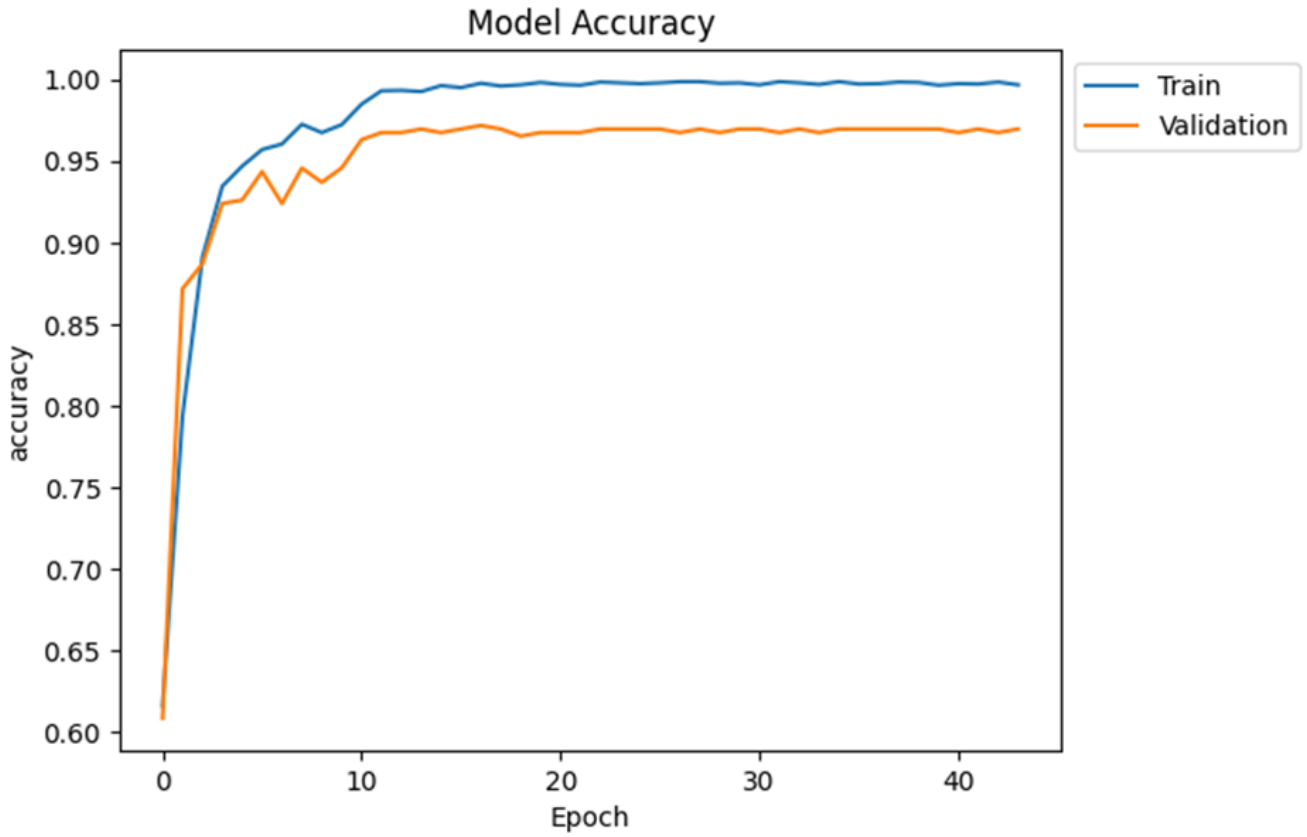


Fig 6.2 Model accuracy

The following graph shows the model Accuracy per epoch for the first epoch model Accuracy is lowest that is 0.6164 which gradually increases from 0.6164 to 0.9966 in 44 epochs and for validation initial Accuracy is 0.60 that gradually increases from 0.60 to 0.97 in 44 epochs. Model accuracy is stable from epoch 15 to 44. The graph shows accuracy for both Training and Validation.

6.2 Confusion matrix

A confusion matrix is a metric tool for evaluating the results of a classification machine learning model. It is a square matrix whose dimensions depend on the number of classes you have in your model. This confusion matrix give description for each class prediction for data. X-axis of confusion matrix represent predicted class labels and Y-axis represent True class labels.

6.2.1 Train Data confusion Matrix

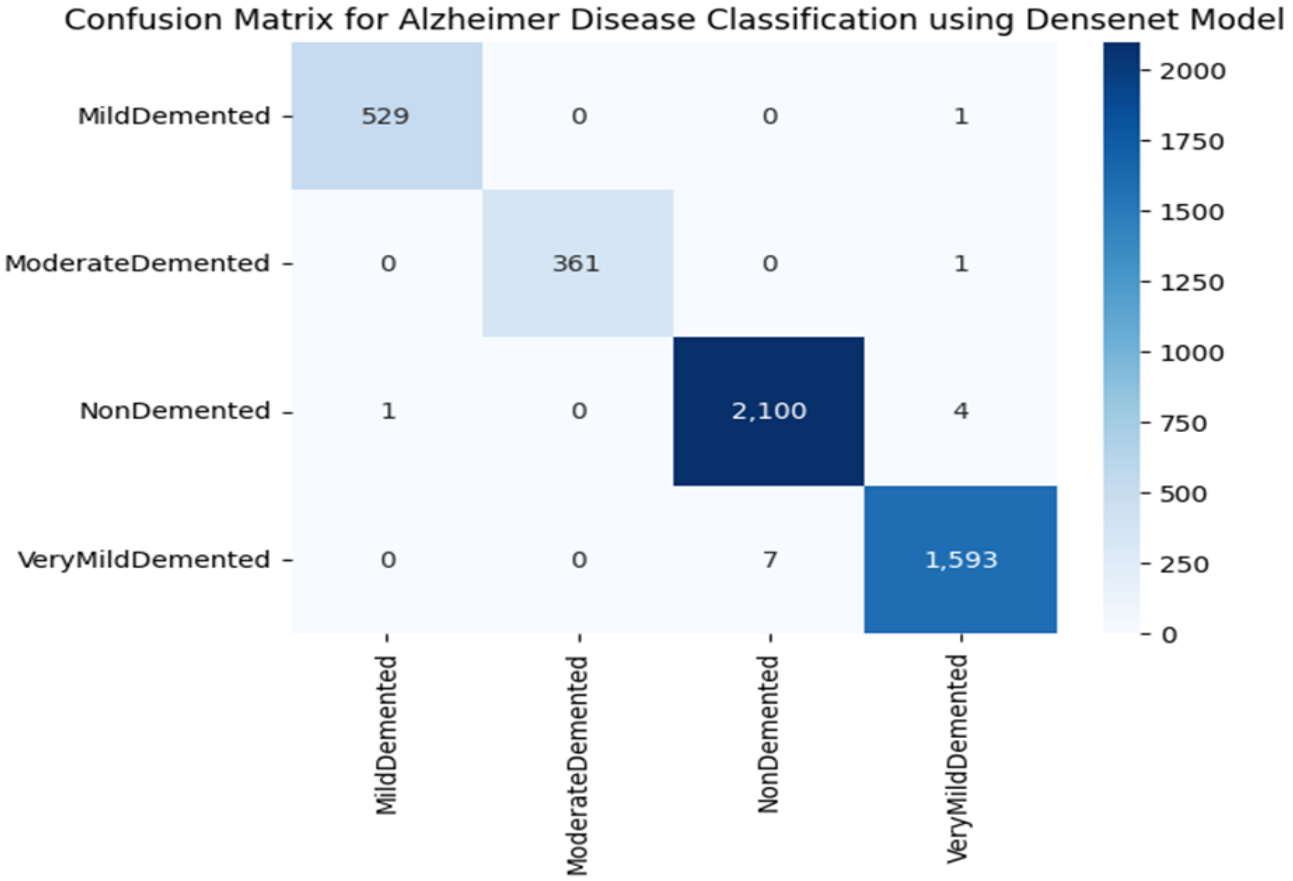


Fig 6.3: Train Confusion matrix

In training data, Mild Demented images were 530 in which 529 images were correctly predicted and 1 image is predicted as Very Mild Demented, Moderate Demented images were 362 in which 361 images were correctly predicted and 1 image is predicted as Very Mild Demented, Non Demented images were 2105 in which 2100 images were correctly predicted and 1 image is predicted as Mild Demented and 4 images as Very Mild Demented, Very Mild Demented images were 1600 in which 1593 images were correctly predicted and 7 image is predicted as Non Demented.

6.2.2 Validation data confusion matrix

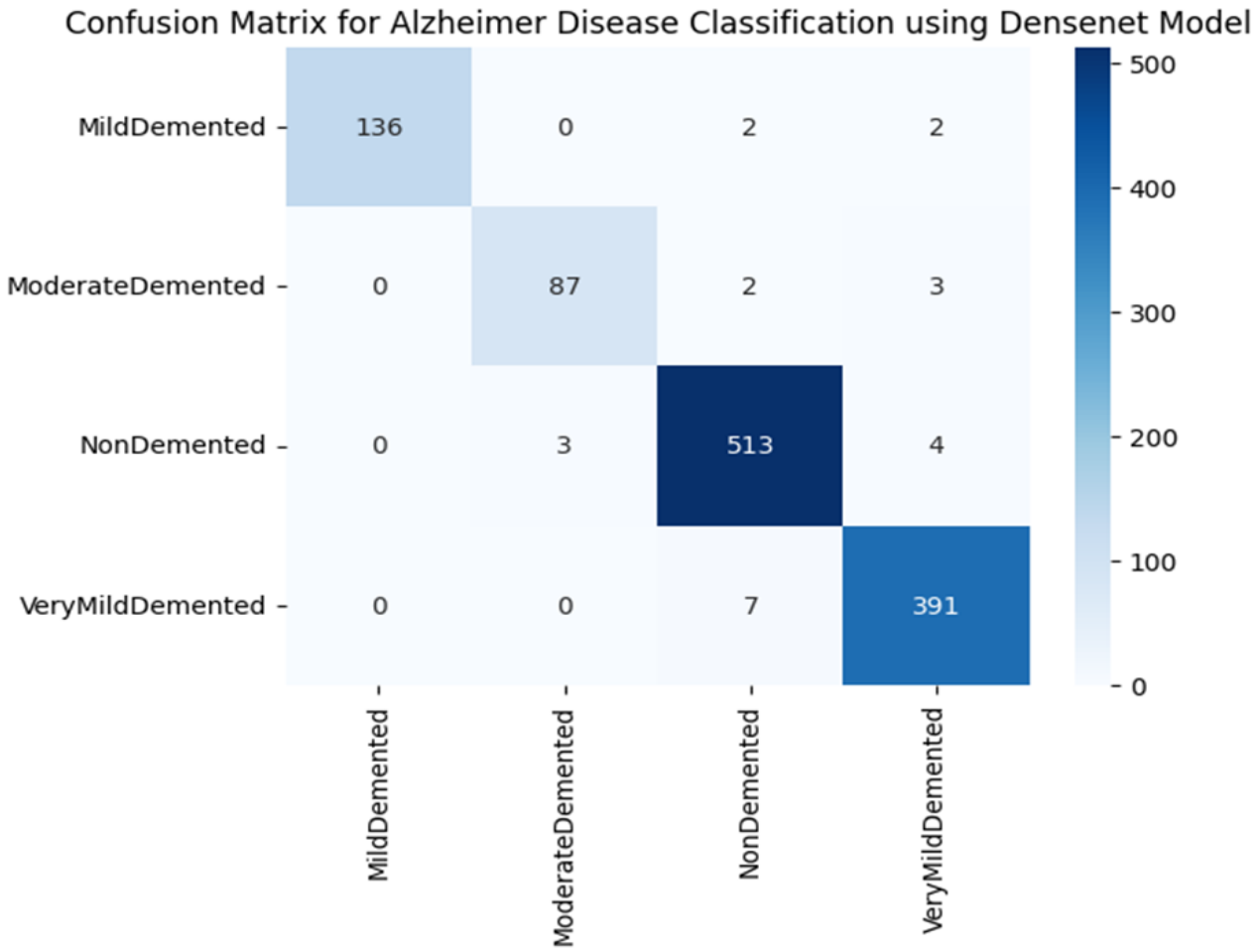


Fig 6.4: Validation Confusion matrix

In validation data, Mild Demented images were 140 in which 139 images were correctly predicted and 2 images were predicted as Non Demented and 2 were predicted as Very Mild Demented, Moderate Demented images were 92 in which 87 images were correctly predicted and 2 images is predicted as Non Demented and 3 were predicted as Very Mild Demented, Non Demented images were 520 in which 513 images were correctly predicted and 3 images were predicted as Moderate Demented and 4 were predicted as Very Mild Demented, Very Mild Demented images were 398 in which 391 images were correctly predicted and 7 image is predicted as Non Demented

6.2.3 Test Data Confusion Matrix

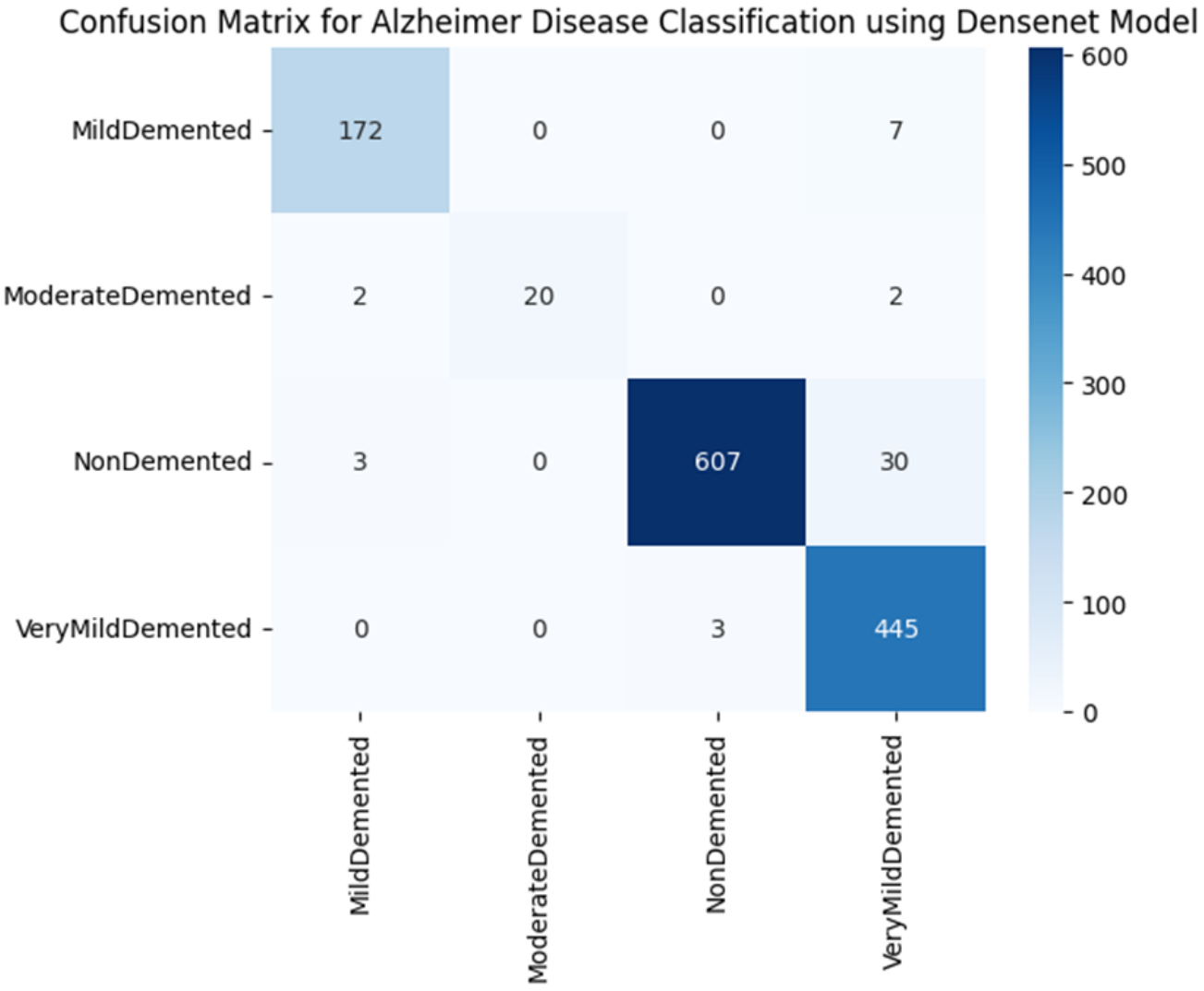


Fig 6.5 Confusion matrix for test data

For Test data, Mild Demented images were 179 in which 172 images were correctly predicted and 7 images were predicted as Very Mild Demented, Moderate Demented images were 24 in which 20 images were correctly predicted and 2 images were predicted as Mild Demented and 2 were predicted as Very Mild Demented, Non Demented images were 640 in which 607 images were correctly predicted and 3 images were predicted as Mild Demented and 30 were predicted as Very Mild Demented, Very Mild Demented images were 448 in which 445 images were correctly predicted and 3 image were predicted as Non Demented.

6.3 Classification Report

A classification report is a performance evaluation metric in machine learning. It is used to show the precision, recall, F1 Score, and support of your trained classification model. We used classification report to find precision, recall and f1score for our model on training, validation, test data.

6.3.1 Train data classification report

	Precision	Recall	F1score	support
Mild Demented	1.00	1.00	1.00	530
Moderate Demented	1.00	1.00	1.00	362
Non Demented	0.99	0.99	0.99	2105
Very Mild Demented	0.99	0.99	0.99	1600
accuracy			0.99	4597
Macro avg	0.99	0.99	0.99	4597
weighted avg	0.99	0.99	0.99	4597

Table 6.1: Categorical classification on train data

The classification report defines Mild Demented gives 1.00 precision, 1.00 recall, 1.00 f1score with support of 530 images, Moderate Demented gives 1.00 precision, 1.00 recall, 1.00 f1score with support of 362 images, non-Demented gives 0.99 precision, 0.99 recall, 0.99 f1score with support of 2105 images, Very Mild Demented gives 0.99 precision, 0.99 recall, 0.99 f1score with support of 1600 images. The model on train data evaluation gives 99.70 % accuracy with support of 4597.

6.3.2 Validation data classification report

	Precision	Recall	F1score	support
Mild Demented	1.00	0.97	0.99	140
Moderate Demented	0.97	0.95	0.96	92
Non Demented	0.98	0.99	0.98	520
Very Mild Demented	0.98	0.98	0.98	398
accuracy			0.98	1150
Macro avg	0.98	0.97	0.98	1150
weighted avg	0.98	0.98	0.98	1150

Table 6.2: Categorical classification report on validation data

The classification report defines, Mild Demented gives 1.00 precision, 0.97 recall, 0.99 f1score with support of 140 images, Moderate Demented gives 0.97 precision, 0.95 recall, 0.96 f1score with support of 92 images, Non-Demented gives 0.98 precision, 0.99 recall, 0.98 f1score with support of 520 images, Very Mild Demented gives 0.98 precision, 0.98 recall, 0.98 f1score with support of 398 images. The model on validation data evaluation gives 98.00 % accuracy with support of 1150.

6.3.3 Test data classification report

	Precision	Recall	F1score	support
Mild Demented	0.97	0.96	0.97	179
Moderate Demented	1.00	0.83	0.91	24
Non Demented	1.00	0.95	0.97	640
Very Mild Demented	0.92	0.99	0.95	448
accuracy			0.96	1291
Macro avg	0.97	0.93	0.95	1291
weighted avg	0.97	0.96	0.96	1291

Table 6.3: Categorical classification report on test data

The classification report defines, Mild Demented gives 0.97 precision, 0.96 recall, 0.97 f1score with support of 179 images, Moderate Demented gives 1.00 precision, 0.83 recall, 0.91 f1score with support of 24 images, Non Demented gives 1.00 precision, 0.95 recall, 0.97 f1score with support of 640 images, Very Mild Demented gives 0.92 precision, 0.99 recall, 0.95 f1score with support of 448 images. The model on validation data evaluation gives 96.36 % accuracy with support of 1291.

6.4 Model comparison

We have compared our DenseNet-169 model with other Deep learning classifiers by supplying our augmented data to compare training and test accuracy, precision and recall for different Deep learning classifiers.

6.4.1 Training Data with different Deep learning Classifiers

Model	Accuracy	Precision	Recall
DenseNet169	99.70%	99.70%	99.70%
DenseNet 121	94.63%	95.25%	93.91%
ResNet50	98.83%	98.50%	97.50%
ResNet101	99.30%	99.25%	99.25%
VGG16	99.15%	99.00%	99.00%
VGG19	98.74%	98.74%	98.75%
MobileNet	97.98%	97.75%	98.50%
MobileNetV2	97.78%	97.50%	97.60%

Table 6.4: Models Comparison on train data

In the above table comparison is done for training accuracy, precision and recall for different Deep Learning Models in which our model gives highest accuracy of 99.70% then second highest accuracy is given by ResNet 101 model with 99.30% and lowest accuracy is given by MobileNetV2 which is 97.78%. DenseNet-169 model gives highest precision and recall values that is 99.70% and MobileNetV2 model gives lowest precision and recall values at 97.50% and 97.60%.

6.4.2 Test Data with different Deep learning Classifiers

Model	Accuracy	Precision	Recall
DenseNet169	96.36%	96.43%	96.36%
DenseNet 121	80.79%	89.00%	72.00%
ResNet50	89.46%	89.52%	89.38%
ResNet101	93.41%	93.91%	93.18%
VGG16	89.38%	93.25%	85.75%
VGG19	92.25%	94.50%	92.75%
MobileNet	78.68%	82.75%	73.50%
MobileNetV2	79.46%	84.00%	77.50%

Table 6.5: Models Comparison on test data

In the above table comparison is done for test accuracy, precision and recall for different Deep Learning Models in which our model gives highest accuracy of 96.36% then second highest accuracy is given by ResNet 101 model with 93.41% and lowest accuracy is given by MobileNet which is 78.68%. DenseNet-169 model gives highest precision and recall values that is 96.43%, 96.36% and MobileNetV2 model gives lowest precision and recall values at 82.75% and 73.50%.

CONCLUSION

Alzheimer's disease is a serious public health problem. Rather than delivering a cure, it is more necessary to lower risk, provide early intervention, and precisely detect symptoms. As seen in the literature review, many efforts have been made to detect Alzheimer's Disease by various machine learning algorithms and micro-simulation methods; however, it remains a difficult task to identify relevant attributes that can detect Alzheimer's very early, which, if detected earlier, can help people with Alzheimer's live a healthy life, as well as their family members. In the study reported here, we suggested using a deep learning model to diagnose Alzheimer's stages. The classification model is evaluated using Kaggle datasets. The experimental findings reveal that the deep learning model has an accuracy value of 96.36%. Given the effectiveness of our most comprehensive multi-diagnostic classifiers, they may be clinically relevant if further prospective RCT-controlled investigations succeed.

FUTURE SCOPE

The future work will focus on improving the accuracy of the model and getting large data for Alzheimer's. This is specially for the moderately demented class to train the model more efficiently. While there are only three callbacks defined and used, there might be other callbacks that can be used to enhance the efficiency and accuracy of the system. We can use custom callbacks to improve the model's efficiency. Our data was executed on a variety of algorithms, out of which DenseNet-169 was rated the highest of all of the algorithms that were run on our data. There might be other deep learning algorithms better than our model but have not been discovered. Using Ensemble learning, you can combine multiple algorithms to achieve the best prediction by combining them together. It is also critical to focus on extraction and analysis of novel features that will help detect Alzheimer's Disease, and on eliminating redundant and irrelevant features from existing feature sets to improve detection accuracy.

REFERENCES

1. Dan Pan, An Zeng, Longfei Jia,Yin Huang,Tory Frizzell and Xiaowei Song (2020) Early Detection of Alzheimer's Disease Using Magnetic Resonance Imaging: A Novel Approach Combining Convolutional Neural Networks and Ensemble Learning.
2. C.Kavitha, Vinodini Mani, S.R Sri Vidhya, Osmah Ibrahim, Carlos Andres (2022) Early-Stage Alzheimer's Disease Prediction Using Machine Learning Models.Early-Stage Alzheimer's Disease Prediction Using Machine Learning Models.
3. Roobaea Alrobaea, Seifeddine Mechti, Mariem Haoues, Saeed Rubaiee(2021) Alzheimer's Disease Early Detection Using Machine Learning Techniques.
4. Weiming lin, Tong Tong, Qinquan Gao, Di Guo, Xiaofeng Du, Yonngui Yang, Gang Guo, Min Xiao, Min Du, Xiaobo Qu (2018) Convolutional Neural Networks-Based MRI Image Analysis for the Alzheimer's Disease Prediction From Mild Cognitive Impairment.
5. Vasco Sá Diogo, Hugo Alexandre Ferreira & Diana Prata (2022) Early diagnosis of Alzheimer's disease using machine learning: a multi-diagnostic, generalizable approach.
6. Carol Y cheung, An Ran Ran, Shujun Wang, Victor TT Chan, Kaiser Sham, Saima hilal (2022) A deep learning model for detection of Alzheimer's disease based on retinal photographs: a retrospective, multicentre case-control study.
7. Taeho jo, Kwangsik Nho, Andrew J.Saykin (2019) Deep Learning in Alzheimer's Disease: Diagnostic Classification and Prognostic Prediction Using Neuroimaging Data.
8. Muhammed Raees, Vinu Thomas (2021) Automated detection of Alzheimer's Disease using Deep Learning in MRI.
9. Sheng Liu, Arjun V. Masukar, Henry Rusinek, Jingyun Chen, Ben Zhang, Weicheng Zhu (2022) Generalizable deep learning model for early Alzheimer's disease detection from structural MRIs.
10. Hiroki Fuse,Kota Oishi, Norihide Maikusa, Tadanori Fukami (2018) Detection of Alzheimer's Disease with Shape Analysis of MRI Images.
11. Hiroki Fuse,Kota Oishi, Norihide Maikusa, Tadanori Fukami (2018) Classification of Patients with Alzheimer's Disease and Healthy Subjects from MRI Brain Images Using the Existence Probability of Tissue Types.
12. Deepti Pachauri, Chris Hinrichs, Moo K. Chung, Sterling C. Johnson, and Vikas Singh (2011) Topology-Based Kernels With Application to Inference Problems in Alzheimer's Disease.

13. Gabriel Lizarraga, Niovi Rojas, Mercedes Cabrerizo, Malek Adjouadi (2016) A Web Platform for Data Acquisition and Analysis for Alzheimer’s Disease.

14. Qi Zhou, Mohammed Goryawala, Mercedes Cabrerizo, Warren Barker, David Loewenstein, Ranjan Duara and Malek Adjouadi (2014) Multivariate Analysis of Structural MRI and PET (FDG and 18FAV-45) for Alzheimer’s Disease and Its Prodromal Stages.
15. Mohamed Mahyoub, Martin Randles, Thar Baker, Po Yang (2018) Effective Use of Data Science Toward Early Prediction of Alzheimer's Disease.
16. Fan Yang, Yuxia Li, Ying Han, Jiehui Jiang (2020) Use of multilayer network modularity and spatiotemporal network switching rate to explore changes of functional brain networks in Alzheimer's disease.
17. H. M. Tarek Ullah, Zishan Ahmed Onik, Riashat Islam, Dr. Dip Nandi (2018) Alzheimer’s Disease And Dementia Detection From 3D Brain MRI Data Using Deep Convolutional Neural Networks.
18. Chenyu Zhang (2020) Genetic Basis of Alzheimer’s Disease and Its Possible Treatments Based on Big Data.
19. Sivakani. R and Gufran Ahmad Ansari (2020) Machine Learning Framework for Implementing Alzheimer’s Disease.
20. Guangyu He, An Ping, Xi Wang, Yufei Zhu (2020) Alzheimer's Disease Diagnosis Model Based on Three-dimensional Full Convolutional DenseNet.
21. Jonathan Young, Marc Modat, Manuel J Cardoso, John Ashburner and Sebastien Ourselin (2012) Classification Of Alzheimer’s Disease Patients And Controls With Gaussian Processes.
22. Binglin Wang, Wei Li, Wenliang Fan, Xi Chen, Dongrui Wu (2019) Alzheimer's Disease Brain Network Classification Using Improved Transfer Feature Learning with Joint Distribution Adaptation.
23. Sarvesh Dubey (2022) Alzheimer's Dataset (4 class of Images).
24. Ron Brookmeyer, Sarah Gray, Claudia Kawas (2000) Projections of Alzheimer's Disease in the United States and the Public Health Impact of Delaying Disease Onset.
25. Julie Zissimopoulos, Eileen Crimmins, Patricia st.Clair (2014) The Value of Delaying Alzheimer’s Disease Onset.
26. Samreen Wani (2022) Alzheimer's-related deaths in India increased 5 times in 30 years.

APPENDIX A

Software and Hardware Requirements

Software Requirements

1. Operating System:	Windows, Mac OS
2. Technology:	Python, notebook, streamlit
3. IDE:	VS Code, Google Colab
4. Python Version:	Python 3.11

Hardware Requirements

1. Hardware:	AMD Ryzen 5 4.2 Ghz
2. RAM:	8 GB to 12 GB
3. GPU:	Nvidia GTX 1650
4. Hard Disk:	100 GB

APPENDIX - B

Technology Used:

Python

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built-in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library is available in source or binary form without charge for all major platforms, and can be freely distributed. Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test- debug cycle makes this simple approach very effective. The language's core philosophy is summarized in the document The Zen of Python (PEP 20), which includes aphorisms such as:

- Beautiful is better than ugly.
- Explicit is better than implicit.
- Simple is better than complex.
- Complex is better than complicated.
- Readability counts.

Rather than having all of its functionality built into its core, Python was designed to be highly extensible (with modules). This compact modularity has made it particularly popular as a means of adding programmable interfaces to existing applications. Van Rossum's vision of a small core language with a large standard library and easily extensible interpreter stemmed from his frustrations with ABC, which espoused the opposite approach. Python strives for a simpler, less-cluttered syntax and grammar while giving developers a choice in their coding methodology. In contrast to Perl's "there is more than one way to do it" motto, Python embraces a "there should be one— and preferably only one —obvious way to do it" design philosophy. Alex Martelli, a Fellow at the Python Software Foundation and Python book author, writes that "To describe something as 'clever' is not considered a compliment in the Python culture. “Python’s developers strive to avoid premature optimization, and reject patches to non-critical parts ofthe CPython reference implementation that would offer marginal increases in speed at the cost of clarity. When speed is important, a Python programmer can move time-critical functions to extension modules written in languages such as C, or use PyPy, a just-in-time compiler. Cython is also available, which translates a Python script into C and makes direct C-level API calls into the Python interpreter. Python’s developers aim for the language to be fun to use. This is reflected in its name—a tribute to the British comedy group Monty Python—and in occasionally playful approaches to tutorials and reference materials, such as examples that refer to spam and eggs (a reference to a Monty Python sketch) instead of the standard foo and bar. A common neologism in the Python community is pythonic, which can have a wide range of meanings related to program style. To say that code is pythonic is to say that it uses Python idioms well, that itis natural or shows fluency in the language, that it conforms with Python's minimalist philosophy and emphasis on readability. In contrast, code that is difficult to understand or reads like a rough transcription from another programming language is called unpythonic.

Visual Studio

Microsoft Visual Studio is an integrated development environment (IDE) from Microsoft. It is used to develop computer programs, as well as websites, web apps, web services and mobile apps. Visual Studio includes a code editor supporting IntelliSense (the code completion component) as well as code refactoring. Visual Studio does not support any programming language, solution or tool intrinsically; instead, it allows the plugging of functionality coded as a VSPackage. When installed, the functionality is available as a Service. The IDE provides three services: SVs Solution, which provides the ability to enumerate projects and solutions; SVs UIShell, which provides windowing and UI functionality (including tabs, toolbars, and tool windows); and SVsShell, which deals with registration of VSPackages. In addition, the IDE is also responsible for coordinating and enabling communication between services. All editors, designers, project types and other tools are implemented as VSPackages. Visual Studio uses COM to access the VSPackages. The Visual Studio SDK also includes the Managed Package Framework (MPF), which is a set of managed wrappers around the COM-interfaces that allow the Packages to be written in any CLI compliant language. However, MPF does not provide all the functionality exposed by the Visual Studio COM interfaces. The services can then be consumed for creation of other packages, which add functionality to the Visual Studio IDE. Support for programming languages is added by using a specific VSPackage called a Language Service. A language service defines various interfaces which the VSPackage implementation can implement to add support for various functionalities. Functionalities that can be added this way include syntax coloring, statement completion, brace matching, parameter information tooltips, member lists, and error markers for background compilation. If the interface is implemented, the functionality will be available for the language. Language services are implemented on a per-language basis. The implementations can reuse code from the parser or the compiler for the language. Language services can be implemented either in native code or managed code. For native code, either the native COM interfaces or the Babel Framework (part of Visual Studio SDK) can be used. For managed code, the MPF includes wrappers for writing managed language services. Visual Studio does not include any source control support built in but it defines two alternative ways for source control systems to integrate with the IDE.A Source Control Package can provide its own customized user interface. In contrast, a source control plugin using the MSSCCI (Microsoft Source Code Control Interface) provides a set of functions that are used to implement various source control functionality, with a standard Visual Studio user interface.

MSSCCI was first used to integrate Visual SourceSafe with Visual Studio 6.0 but was later opened up via the Visual Studio SDK. Visual Studio .NET 2002 used MSSCCI 1.1, and Visual Studio .NET 2003 used MSSCCI 1.2. Visual Studio 2005, 2008, and 2010 use MSSCCI Version 1.3, which adds support for rename and delete propagation, as well as asynchronous opening.

Visual Studio supports running multiple instances of the environment (each with its own set of VSPackages). The instances use different registry hives (see MSDN's definition of the term "registry hive" in the sense used here) to store their configuration state and are differentiated by their AppId (Application ID). The instances are launched by an AppId-specific .exe that selects the AppId, sets theroot hive, and launches the IDE. VSPackages registered for one AppId are integrated with other VSPackages for that AppId. The various product editions of Visual Studio are created using the different AppIds. The Visual Studio Express edition products are installed with their own AppIds, but the Standard, Professional, and Team Suite products share the same AppId. Consequently, one can install the Express editions side-by-side with other editions, unlike the other editions which update thesame installation. The professional edition includes a superset of the VSPackages in the standard edition, and the team suite includes a superset of the VSPackages in both other editions. The AppId system is leveraged by the Visual Studio Shell in Visual Studio 2008.

Visual Studio Code includes built-in support for IntelliSense code completion, rich semantic code understanding and navigation, and code refactoring. And when the coding gets tough, the tough get debugging.

Colab:

Colab, short for Google Collaboratory, is a powerful cloud-based platform that has gained popularity among data scientists, researchers, and developers. Developed by Google, Colab offers a convenient environment for creating, sharing, and running code in a Jupyter notebook format. One of the major advantages of Colab is its integration with Google Drive, allowing users to easily store and access notebooks and datasets. Colab provides a hassle-free setup as it comes preinstalled with popular libraries and frameworks for machine learning and data analysis, including TensorFlow, PyTorch, and pandas. Another standout feature of Colab is its free access to powerful GPU and TPU resources, enabling users to train and run complex models with enhanced speed and efficiency. Additionally, Colab encourages

collaboration through real-time editing and sharing options, allowing multiple users to work on the same notebook simultaneously. With its user-friendly interface, extensive library support, and cloud-based capabilities, Colab has become an invaluable tool for individuals and teams working on various data-related tasks. Furthermore, Colab offers a range of productivity-enhancing features. It provides a vast selection of pre-installed code snippets and examples, making it easier for users to explore new techniques and learn from existing implementations. Colab also supports the use of Markdown cells, enabling users to create interactive and informative documentation alongside their code. The platform seamlessly integrates with other Google services, such as Google Sheets, Big Query, and Google Cloud Storage, enabling users to access and process large datasets directly within their notebooks. Another noteworthy feature of Colab is its ability to run notebooks on Google Cloud Platform (GCP), allowing users to leverage the scalability and robustness of cloud infrastructure. Colab even supports the execution of shell commands, granting users flexibility in managing their environments and dependencies. With its versatility, collaborative nature, and integration with powerful tools, Colab has become a go-to platform for data analysis, machine learning, and research projects, empowering users to focus on their work rather than infrastructure setup and management.

TensorFlow:

TensorFlow is an open-source machine learning framework developed by Google Brain. It has gained immense popularity among researchers, data scientists, and developers due to its powerful capabilities and ease of use. TensorFlow is designed to build and deploy machine learning models efficiently, enabling users to create complex neural networks and deep learning architectures. Its core strength lies in its ability to handle large-scale datasets and perform computations on GPUs and TPUs, accelerating the training and inference processes. TensorFlow offers a high-level API called Keras, which simplifies the model development process by providing intuitive abstractions and a wide range of pre-built layers and models. Additionally, TensorFlow's extensive ecosystem provides libraries and tools for various tasks, including computer vision, natural language processing, and reinforcement learning. It also offers TensorFlow Serving for deploying models in production, TensorFlow.js for running models in web browsers, and TensorFlow Lite for mobile and embedded devices. With its versatility, scalability, and community support, TensorFlow has become a go-to framework for building cutting-edge machine learning solutions and pushing the boundaries of artificial intelligence. TensorFlow is renowned for its flexibility and versatility, allowing users to implement a wide range of machine learning algorithms and architectures. It provides a symbolic programming framework where users define and manipulate complex computational graphs. This graph-based approach allows for efficient computation and optimization, making TensorFlow suitable for both research and production scenarios. TensorFlow's eager execution mode further simplifies the development process by enabling users to execute operations dynamically, akin to traditional imperative programming. It supports automatic differentiation, facilitating the training of neural networks through gradient-based optimization algorithms. TensorFlow's extensive library ecosystem, including TensorFlow Probability, TensorFlow Datasets, and TensorFlow Hub, offers additional tools and resources to support various machine learning tasks. Moreover, TensorFlow's integration with other popular frameworks and libraries, such as Keras and scikit-learn, allows for seamless interoperability and transfer of models. TensorFlow's commitment to community-driven development has fostered a vibrant ecosystem, with numerous contributions from researchers and practitioners worldwide. Whether it's building state-of-the-art deep learning models, developing custom algorithms, or deploying machine learning solutions at scale, TensorFlow continues to empower users to push the boundaries of artificial intelligence and tackle complex real-world challenges.

Keras:

Keras is a user-friendly, open-source deep learning framework written in Python. It is built on top of TensorFlow and provides a high-level API that simplifies the process of developing and training neural networks. With its focus on ease of use and intuitive syntax, Keras has become a popular choice among beginners and experienced researchers alike. Keras abstracts away the complexities of low-level operations and provides a set of powerful building blocks for creating neural networks. It offers a wide range of pre-built layers, activation functions, and loss functions, making it easy to assemble and customize models. Keras also supports both sequential and functional model architectures, allowing users to create simple linear pipelines or more complex graph-like networks. It provides a straightforward interface for model compilation, training, and evaluation, with convenient methods for handling data preprocessing and augmentation. Additionally, Keras seamlessly integrates with TensorFlow, enabling users to leverage the extensive capabilities of TensorFlow while enjoying the simplicity and readability of the Keras API. Keras's versatility, ease of use, and strong community support have established it as a go-to tool for rapid prototyping, experimentation, and production deployment of deep learning models**.** Furthermore, Keras offers a wide range of features and functionalities that enhance the deep learning workflow. It provides support for various types of networks, including convolutional neural networks (CNNs), recurrent neural networks (RNNs), and transformer models, enabling users to tackle diverse tasks such as image classification, natural language processing, and sequence generation. Keras also includes built-in support for model saving and loading, allowing users to easily store trained models and resume training or make predictions without the need to retrain from scratch. It offers convenient callbacks that enable users to monitor training progress, perform early stopping, and save the best model weights during training. Keras seamlessly integrates with popular Python libraries for data manipulation and visualization, such as NumPy and Matplotlib, providing a comprehensive ecosystem for deep learning projects. Additionally, Keras has a large and active community, which means there are abundant resources, tutorials, and pre-trained models available, making it easier for users to get started and leverage existing knowledge. Overall, Keras's combination of simplicity, flexibility, and strong integration with TensorFlow has solidified its position as a top choice for developing and deploying deep learning models efficiently.

Streamlit:

Streamlit is an open-source Python library that allows you to create interactive web applications for data science and machine learning projects. It simplifies the process of building and deploying web applications by providing a straightforward and intuitive interface.

With Streamlit, you can create web applications directly from Python scripts. It provides a set of easy-to-use commands for displaying data, creating interactive widgets, and updating the content dynamically. You can use Streamlit to build dashboards, visualizations, data exploration tools, and more.

Google Drive:

Google Drive is a cloud storage and file synchronization service provided by Google. It allows you to store files, documents, photos, videos, and more in the cloud, and access them from any device with an internet connection. Google Drive provides a convenient way to store, share, and collaborate on files.

Here are some key features and functionalities of Google Drive:

1. File Storage: You can upload and store files of various types, including documents, spreadsheets, presentations, images, videos, and more. Google Drive offers 15 GB of free storage space, and you can purchase additional storage if needed.

2. File Synchronization: Google Drive provides a desktop application called Google Drive for PC and Mac that allows you to synchronize files between your computer and the cloud. Any changes made to files in your local Google Drive folder will be automatically synced to the cloud, and vice versa.

3. Sharing and Collaboration: You can easily share files and folders with others using Google Drive. You can control the level of access (viewing, commenting, or editing) for each person you share with. Collaboration features enable multiple users to work on the same document simultaneously, making it easy to collaborate on projects in real-time.

4. File Organization: Google Drive allows you to create folders and subfolders to organize your files and keep them structured. You can also add tags and metadata to files to make searching and locating them easier.

5. Integration with Google Workspace: Google Drive integrates seamlessly with other Google Workspace apps such as Google Docs, Sheets, and Slides. This integration enables you to create and edit documents directly in Google Drive without the need for additional software.

6. Advanced Search: Google Drive provides a powerful search functionality that allows you to search for files using keywords, file types, owners, modification dates, and more. It can help you quickly find the files you need, even if you have a large number of files stored.

7. Security and Privacy: Google Drive employs robust security measures to protect your files and data. It uses encryption to secure files during transmission and storage. You can also set access permissions for each file or folder to control who can view, edit, or share them.

To use Google Drive, you need a Google Account. You can access Google Drive through a web browser by visiting drive.google.com or by downloading the Google Drive app on your computer or mobile device.

Kaggle:

Kaggle is an online platform that hosts data science competitions and provides a wide range of datasets for learning and practice. Kaggle datasets are a collection of publicly available datasets uploaded by users, including researchers, data scientists, and enthusiasts. These datasets cover various domains and are often used for exploratory data analysis, model training, and testing.

To access Kaggle datasets, you need to create a Kaggle account and accept the competition rules and terms of service. Here's a step-by-step guide on how to access and download datasets from Kaggle:

1. Create a Kaggle account: Go to the Kaggle website (www.kaggle.com) and sign up for an account using your email address or by linking your Google account.

2. Accept the competition rules: Some datasets on Kaggle may be associated with ongoing or past competitions. To access these datasets, you need to join the competition and accept its rules and terms.

3. Explore the datasets: Once you have logged in to your Kaggle account, you can browse through the datasets available on the platform. You can search for specific datasets using keywords, filter them by category or popularity, and sort them based on different criteria.

4. Select a dataset: Choose a dataset that interests you by clicking on its name or thumbnail. This will take you to the dataset's page, where you can find detailed information about the dataset, such as the description, number of files, size, and any associated tasks or competitions.

5. Download the dataset: On the dataset's page, you will find a "Download" button or a link to download the dataset files. Click on the button or link to start the download process. The dataset may be downloaded as a single file or as a compressed archive (e.g., a ZIP file).

6. Use the dataset: Once the dataset is downloaded, you can extract the files if necessary and start working with the data. You can use various tools and programming languages (e.g., Python, R, or SQL) to explore, analyze, and visualize the dataset according to your specific goals or tasks.

It's important to note that some datasets may have specific licensing restrictions or usage terms. Make sure to review the dataset's documentation or associated competition rules to understand any limitations or requirements for using the data.

Kaggle also provides a command-line tool called Kaggle CLI, which allows you to interact with Kaggle and download datasets directly from the command line. You can find more information about the Kaggle CLI in the Kaggle documentation.

Remember to always respect the data usage terms, provide appropriate attribution if required, and be mindful of any privacy or confidentiality considerations when working with datasets from Kaggle or any other source.

APPENDIX-C

LIST OF ABBREVIATIONS

AD: Alzheimer’s disease

ADNI: Alzheimer’s Disease Neuroimaging Initiative

BAC: Balanced accuracy

ML: Machine learning

MRI: Magnetic resonance imaging

OASIS: Open Access Series of Imaging Studies

OD: Odds ratio

PET: Positron emission tomography

PPV: Positive predict value

RCT: Randomized controlled trial

AUC: Area under the receiver operating characteristic curve

ROI:Region of interest

RBG: Red Blue Green

OpenCV: Open-Source Computer Vision

DFD: Data Flow Diagram

UML: unified modeling language

Std: standard deviation

VS code: Visual Studio Code

IDE: integrated development environment

CNN: Convolutional neural network

CL: Convolutinal layer

PL: Pooling Layer

FCL: Fully Connected Layer

ReLU: Rectified Linear Unit

BN: Batch normalization

RESOURCES