ALZHEIMER DISEASE DETECTION USING VGG

A DESIGN PROJECT REPORT

Submitted by

AGNES MARY LAVANYA A

BRINDHA G

SAHANA SRI D

VINODHA R

in partial fulfilment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

ARTIFICIAL INTELLIGENGE AND DATA SCIENCE

K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY

(An Autonomous Institution, affiliated to Anna University Chennai and Approved by AICTE,New Delhi)

SAMAYAPURAM–621112

JUNE 2024

K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY

(AUTONOMOUS)

SAMAYAPURAM–621112

BONAFIDE CERTIFICATE

Certified that this design project report titled **"ALZHEIMER DISEASE DETECTION USING VGG"** is the bonafide work of **AGNES MARY LAVANYA A (811721243004), BRINDHA G (811721243012), SAHANA SRI D (811721243046), VINODHA R (811721243062)** who carried out the project under my supervision.

SIGNATURE
Dr. T. Avudaiappan M.E., Ph.D.
HEAD OF THE DEPARTMENT
Associate Professor
Department of Artificial Intelligence
K. Ramakrishnan College of
Technology (Autonomous)
Samayapuram – 621 112

SIGNATURE
Mrs. D. Deena Rose., M.E., (Ph.D.)
SUPERVISOR
Assistant Professor
Department of Artificial Intelligence
K. Ramakrishnan College of
Technology (Autonomous)
Samayapuram – 621 112

Submitted for the viva-voce examination held on ………………

INTERNAL EXAMINER **EXTERNAL EXAMINER**

DECLARATION

We jointly declare that the project report on **"ALZHEIMER DISEASE DETECTION USING VGG"** is the result of original work done by us and best of our knowledge, similar work has not been submitted to **"ANNA UNIVERSITY CHENNAI"** for the requirement of Degree of **BACHELOR OF TECHNOLOGY**. This design project report is submitted on the partial fulfilment of the requirement of the award of Degree of **BACHELOR OF TECHNOLOGY**.

SIGNATURE

____

AGNES MARY LAVANYA A

____

BRINDHA G

____

SAHANA SRI D

____

VINODHA R

PLACE **:** SAMAYAPURAM

DATE **:**

ACKNOWLEDGEMENT

It is with great pride that we express our gratitude and in - debt to our institution **"K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY (AUTONOMOUS)"**, for providing us with the opportunity to do this project.

We are glad to credit honorable Chairman **Dr. K. RAMAKRISHNAN**, **B.E.,** for having provided for the facilities during the course of our study in college.

We would like to express our sincere thanks to our beloved Executive Director **Dr. S. KUPPUSAMY, MBA., Ph.D.,** for forwarding to our project and offering adequate duration in completing our project.

We would like to thank our Principal **Dr. N. VASUDEVAN, M.E., Ph.D.,** whogave opportunity to frame the project the full satisfaction.

We whole heartily thanks to **Dr. T. AVUDAIAPPAN**, **M.E., Ph.D.,** HEAD OF THE DEPARTMENT, **ARTIFICAL INTELLIGENCE** for providing his encourage pursuing this project.

We express our deep and sincere gratitude to my project guide **Mrs. D. DEENA ROSE., M.E., (Ph.D.),** ASSISTANT PROFESSOR, **ARTIFICIAL INTELLIGENCE** for her incalculable suggestions, creativity, assistance and patience which motivated me to carry out the project successfully.

We render our sincere thanks to my project coordinator **Mrs. G. NALINA KEERTHANA M.E.,** ASSISTANT PROFESSOR, **ARTIFICIAL INTELLIGENCE** other faculties and non-teaching staff members for providing valuable information during the course. We wish to express our special thanks to the officials & Lab Technicians of our departments who rendered their help during the period of the work progress.

ABSTRACT

Alzheimer's disease is an irremediable, continuous brain disorder that gradually destroys memory and thinking skills and, eventually, the ability to carry out the simplest tasks. It has become one of the critical diseases throughout the world. Moreover, there is no remedy for Alzheimer's disease. Machine learning techniques, especially deep learning-based Convolutional Neural Network (CNN), are used to improve the process for the detection of Alzheimer's disease. In recent days, CNN has achieved major success in MRI image analysis and biomedical research. VGG-16 is characterized by its simplicity and uniform architecture, making it easy to understand and implement. The performance of the proposed model is compared with some existing CNN models in terms of accuracy, precision, recall, F1 score, and ROC curve. Our proposed model deals with VGG (Visual Geometry Group) In convolutional neural network for Alzheimer disease detection based on the information gathered from the medical image datasets.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

ADAS	Alzheimer's Disease Assessment Scale
ADNI	Alzheimer's Disease Neuroimaging initiative
CNN	Convolution Neural Network
MMSE	Mini Mental State Examination
MRI	Magnetic resonance imaging
OASIS	Open Access Series of Imaging Studies
PET	Positron Emission Tomography
SVM	Support Vector Machine
VGG	Visual Geometry Group

CHAPTER 1

INTRODUCTION

The Alzheimer's disease is the most common cause of dementia, affecting millions of people worldwide. Early diagnosis of AD can significantly impact patient care, enabling interventions to slow disease progression and improve quality of life. Magnetic Resonance Imaging (MRI) scans provide valuable insights into brain structure and function, aiding in the detection of AD-related changes. CNNs have emerged as powerful tools for automated analysis of medical images due to their ability to learn hierarchical features directly from data. The integration of VGG within Convolutional Neural Networks (CNNs) offers a potent avenue for Alzheimer's disease (AD) detection through medical imaging. Utilizing neuroimaging data, such as MRI scans, preprocessed inputs are fed into the VGG architecture. Trained on labeled datasets, VGG learns to discern AD-specific patterns from normal brain structures. Evaluation metrics, including accuracy, assess the model's performance. This approach holds promise for early AD diagnosis, facilitating timely intervention and improved patient outcomes.

1.1 BACKGROUND

Our proposed system Alzheimer's disease detection is a progressive neurological disorder that primarily affects memory and other cognitive functions. Early detection is crucial for better management and treatment outcomes. Convolutional Neural Networks (CNNs) have shown great promise in medical image analysis tasks, including disease detection. The VGG architecture is known for its simplicity and effectiveness in image recognition tasks. It consists of several convolutional layers followed by max-pooling layers, and it is known for its ability to capture complex features in images. CNNs, including VGG, can be used to analyze medical imaging data such as MRI scans or PET scans to detect patterns associated with Alzheimer's disease. Preprocessing steps such as normalization and augmentation may be applied to the imaging data before feeding it into the CNN model. The CNN model is trained using labeled imaging data, where the labels indicate the presence or absence of Alzheimer's disease. During training, the model learns to identify features in the images that are indicative of the disease. Once the model is trained, it can be used to analyze new imaging data and predict the likelihood of Alzheimer's disease based on the patterns it has learned. The model's performance can be evaluated using metrics such as accuracy, sensitivity, and specificity.

1.2 PROBLEM STATEMENT

Detecting Alzheimer's disease from brain MRI scans using the VGG architecture aims to develop a CNN model for accurate diagnosis. Utilizing a dataset comprising MRI images of individuals with and without Alzheimer's, the project preprocesses images, splits data, implements VGG architecture, and trains the model. Validation and testing assess model performance, optimizing hyperparameters for enhanced accuracy. Outcome analysis focuses on metrics like accuracy, precision, recall, and F1- score. Success in this endeavor could facilitate early diagnosis and monitoring of Alzheimer's, offering significant advancements in patient care and treatment outcomes.

1.3OBJECTIVES

- To Design and train a CNN architecture capable of accurately detecting patterns indicative of Alzheimer's disease from brain imaging data, such as MRI scans
- To Explore and implement preprocessing techniques to enhance the quality and consistency of input data. Additionally, investigate data augmentation methods.

- To Design and train a CNN architecture capable of accurately detecting patterns indicative of Alzheimer's disease from brain imaging data, such as MRI scans or PET scans.

- To Explore and implement preprocessing techniques to enhance the quality and consistency of input data. Additionally, investigate data augmentation methods to increase the diversity of the training dataset, potentially improving the model's generalization capabilities.

- To Identify and select the most relevant features from medical data that contribute significantly to the differentiation of Alzheimer’s Disease

- To Compare the efficiency of VGG in CNN against other machine learning approaches to showcase its superiority in Alzheimer’s Disease detection.

- To minimize false positives and false negatives, thereby improving overall diagnostic accuracy and reducing the chances of misclassification.

- To evaluate the model's performance in distinguishing between healthy individuals and those with Alzheimer’s Disease, ultimately contributing to the advancement of early diagnostic methods in the field of Mental health.

- To develop a convolutional neural network (CNN) model capable of accurately detecting Alzheimer's disease from brain imaging data, with the aim of improving early diagnosis and monitoring of the disease progression.

- To Ensure the developed model aligns with clinical diagnoses and is interpretable for healthcare professionals, aiding in decision-making for patient care and treatment strategies.

- To Create a model that is computationally efficient and scalable for integration into medical practice, allowing for quick and reliable assessment of Memory conditions.

- To Validate the model's performance across diverse datasets to ensure its robustness and generalizability to different patient demographics and imaging variations.

- To increase overall accuracy by using VGG in CNN.

CHAPTER 2

LITERATURE SURVEY

2.1 Automated Medical Diagnosis of Alzheimer Disease Using an Efficient Net Convolutional Neural Network

AUTHORS: Zihao Chen, Haijun Lei, Zhongwei Huang, Baiying Lei.

YEAR OF PUBLICATION: 2022

ALGORITHMS USED:

Simple Linear Regression, Feature Learning and Support Vector Machine Algorithm.

ABSTRACT:

Alzheimer's disease (AD) is a common brain disease in the elderly that leads to thinking and behavior disorders. As the population ages, the proportion of AD patients is also increasing. Accordingly, computer-aided diagnosis of AD attracts more and more attention recently. Specifically, latent space learning is employed to obtain the inter-relationship between multiple templates, and feature learning is performed to explore the intrinsic relation in feature space. Finally, the most discriminative features are selected to boost the multi-classification performance. Our proposed model uses the data from the Alzheimer's disease neuroimaging initiative dataset.

MERIT:

Feature learning using multi-template approaches enhances robustness by capturing diverse.

DEMERIT:

Alzheimer's disease face challenges of complexity, potential overfitting.

2.2 Deep Learning Approaches for Brain Disease Diagnosis: Principles and Recent Advances.

AUTHORS: Md. Fazlul Kader, Kyung-Sup Kwak, Protima Khan, S. M. Riazul Islam.

YEAR OF PUBLICATION: 2022

ALGORITHMS USED:

SVM, Random Forest, Decision Tree, KNN classifier, RNN, Boltzmann machine, autoencoders.

ABSTRACT:

Brain is the controlling center of our body. With the advent of time, newer and newer brain diseases are being discovered. Thus, because of the variability of brain diseases, existing diagnosis or detection systems are becoming challenging and are still an open problem for research. Application of AI in medical science has made brain disease prediction and detection more accurate and precise. In this study, we present a review on recent machine learning and deep learning approaches in detecting four brain diseases such as Alzheimer's disease (AD), brain tumor, epilepsy, and Parkinsons disease. Moreover, a brief overview of different feature extraction techniques that are used in diagnosing brain diseases is provided.

MERIT:

Machine learning and deep learning approaches offer unparalleled potential for brain disease diagnosis.

DEMERIT:

Hindering their clinical adoption despite their high accuracy and performance.

2.3 Performances of Machine Learning Models for Diagnosis of Alzheimer's Disease

AUTHORS: Francisco J. Martinez-Murcia, Andres Ortiz, Juan-Manuel Gorriz, Javier Ramirez and Diego Castillo-Barnes.

YEAR OF PUBLICATION: 2021

ALGORITHMS USED:

Principal Component Analysis, Convolutional Autoencoder, Regression and classification Algorithms.

ABSTRACT:

Many classical machine learning techniques have been used to explore Alzheimer's disease (AD), evolving from image decomposition techniques such as principal component analysis toward higher complexity, non-linear decomposition algorithms. With the arrival of the deep learning paradigm, it has become possible to extract high-level abstract features directly from MRI images that internally describe the distribution of data in low-dimensional manifolds. In this work, we try a new exploratory data analysis of AD based on deep convolutional autoencoders. The distribution of the extracted features in different combinations is then analyzed and visualized using regression and classification analysis, and the influence of each coordinate of the autoencoder manifold over the brain is estimated.

MERIT:

Enhanced accuracy and feature learning capabilities in classification tasks.

DEMERIT:

Increased computational complexity and training time due to the combination of powerful algorithms.

2.4 Explainable AI Based on Brain Structure Variation for Alzheimer's Disease Dynamic Prediction.

AUTHORS: Yu Zhang, Tong Liu, Vitaveska Lanfranchi, And Po Yang.

YEAR OF PUBLICATION: 2023

ALGORITHM USED:

Gradient Boosting, Ensemble Learning, Explainable AI and Regularization Techniques.

ABSTRACT:

Machine learning approaches for predicting Alzheimer's disease (AD) progression can substantially assist researchers and clinicians in developing effective AD preventive and treatment strategies. Furthermore, as subjects have continuous records of brain biomarker testing, the model is extended to ensemble the subjects' temporally continuous prediction results utilizing a gradient boosting kernel to find more accurate predictions. Results demonstrate that the proposed model have superior accuracy and stability in predicting AD progression compared to benchmarks and state-of-the-art multitask regression methods in terms of the Mini Mental State Examination (MMSE) questionnaire and The Alzheimer's Disease Assessment Scale-Cognitive Subscale (ADAS-Cog) cognitive scores.

MERIT:

Explainable Tensor Multi-Task Ensemble Learning offers a novel approach for Alzheimer's disease prediction.

DEMERIT:

The complexity of the model may hinder practical implementation.

2.5 Early-Stage Alzheimer's Disease Prediction Using Machine Learning Models.

AUTHORS: C. Kavitha, Vinodhini Mani, S. R. Srividhya, Osamah Ibrahim Khalaf and Carlos Andrés Tavera Romero.

YEAR OF PUBLICATION: 2023

ALGORITHMS USED:

Decision Tree Classifier, Random Forest Classifier, XG Boost, Support Vector Machine (SVM) algorithms.

ABSTRACT:

Alzheimer's disease (AD) is the leading cause of dementia in older adults. Their incidence rates are increasing at an alarming rate every year. In Alzheimer's disease, the brain is affected by neurodegenerative changes. A treatment given at an early stage of AD is more effective, and it causes fewer minor damage than a treatment done at a later stage. Several techniques such as Decision Tree, Random Forest, Support Vector Machine, Gradient Boosting, and Voting classifiers have been employed to identify the best parameters for Alzheimer's disease prediction. Predictions of Alzheimer's disease are based on Open Access Series of Imaging Studies (OASIS) data, and performance is measured with parameters like Precision, Recall, Accuracy, and F1-score for ML models. The proposed work shows better results with the best validation average accuracy of 83% on the test data of AD.

MERIT:

Real-time, non-invasive monitoring of tissue conductivity changes for medical.

DEMERIT:

Increased resource requirements in terms of data and computation accuracy.

CHAPTER 3

SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

- In this work, including the k-Nearest-Neighbor algorithm, decision trees, support vector machines, and artificial neural networks. Classification and prediction based on the data set obtained from the UCI Repository were carried out, and accuracy was obtained based on the output produced.
- SVM machine learning and logistic regression are used Based on Precision, Recall, F measure, ROC, and RMS error, a comparison between these two algorithms was made. In the end, the best classifier was logistic regression.
- The classification accuracy, sensitivity, and specificity of the radiomics-based method are 66.81%, 51.19%, and 75.77%, respectively, while the evaluation indices for the deep learning-based method trained on the test samples are 74.69%, 63.10%, and 80.20%, respectively. The most effective methods ended up being deep learning.

ALGORITHM USED

- **SUPPORT VECTOR MACHINES (SVM):** While Support Vector Machines (SVM) are powerful tools for thyroid detection in machine learning, they come with certain drawbacks. One limitation is the sensitivity to the choice of hyperparameters, such as the regularization parameter(C) and the kernel type. Tuning these parameters can be computationally expensive and might require extensive experimentation to achieve optimal performance. Additionally, SVMs may not perform well on large datasets, as the time complexity increases with the square of the number ofsamples. Another drawback is that SVMs are binary classifiers, and extension to multiclass problems involves strategies like one-vs-one or one-vs-all, which can be less intuitive. Moreover, SVMs are less effective when dealing with noisy data or datasets with overlapping classes, as they aim to find a clear boundary between classes. Despite these drawbacks, SVMs remain a valuable tool for many classification tasks, including thyroid detection, especially when appropriate preprocessing and parameter tuning are applied.

- **DECISION TREE CLASSIFIER:** Implementing a Alzheimer detection project using decision trees offers interpretability, simplicity in visualization, and ease of understanding for non-technical stakeholders. However, decision trees are prone to overfitting, especially when the tree depth is not properly regulated or when dealing with noisy or high-dimensional data. They might create overly complex trees that fail to generalize well to new, unseen data. Additionally, decision trees can be sensitive to small variations in the data, leading to different trees being generated with minor changes in the training set. Furthermore, they may not perform optimally compared to more sophisticated algorithms in certain scenarios, especially when dealing with intricate relationships among features or when handling imbalanced datasets, where they might favor the majority class. Therefore, while decision trees offer simplicity and interpretability, their performance might lag behind more complex models in terms of predictive accuracy and generalization.

- **K-NEAREST NEIGHBOUR:** The implementation of Alzheimer detection using the k-Nearest Neighbors (KNN) algorithm involves several key steps. Initially, a dataset containing relevant features such as medical test results and patient demographics is collected and preprocessed. The preprocessing step includes handling missing values and normalizing numerical features. The dataset is then split into training and testing sets. During the training phase, the KNN algorithm memorizes the feature vectors of the training instances. In the testing phase, for each instance in the testing set, the algorithm identifies the k-nearest neighbors in the training set based on a distance metric (e.g., Euclidean distance). The majority class among these neighbors determines the class of the test instance. The algorithm's performance is evaluated using metrics like accuracy.

3.1.1 DRAWBACKS

- SVMs are sensitive to parameter choices and may require extensive tuning, making them computationally intensive. Their scalability is a concern for large datasets.
- KNN, although intuitive, becomes computationally expensive with high-dimensional or extensive data, and its predictions can be sensitive to noise and outliers. The memory usage is also a consideration.
- Decision Trees, while interpretable, are prone to overfitting, especially with deep trees, and may struggle with capturing intricate non-linear relationships.

3.2 PROPOSED SYSTEM

In case of using Decision tree algorithm, Support vector machine and K-Nearest Neighbor algorithm we may need to face with the complexity and overfitting since these algorithms are prone to overfitting. These can be computationally expensive, especially when dealing with a large number of trees and features. To overcome these issues we are giving a turn to VGG in CNN since it provides easily interpretable results and makes it easy to understand the importance of individual features in the diagnosis. VGG is a simple, which is computationally effective and efficient.

VGG ALGORITHM

VGG (Visual Geometry Group) is a convolutional neural network architecture known for its simplicity and effectiveness in image classification tasks. It consists of multiple layers of convolutional and pooling operations, followed by fully connected layers. VGG has a fixed architecture with deeper networks having more layers, enabling it to capture intricate features in images, making it widely used in computer vision tasks.

3.2.1 ADVANTAGES

- Simplicity and Wide Applicability
- Transfer Learning
- Effective Feature Extraction
- Consistency

CHAPTER 4

SYSTEM SPECIFICATION

4.1 HARDWARE SYSTEM CONFIGURATION

- **Computer -** minimum of 4GB RAM & dual-core processor
- **RAM** – minimum 16GB to 32GB.
- **Stable internet connection**
- **Storage**

4.2 SOFTWARE SYSTEM CONFIGURATION

- **Python programming language**
- **Operating system -** Windows, Linux, or macOS.
- **Python libraries** such as – Numpy , pandas, Matplotlib.

4.3 SOFTWARE DESCRIPTION

Utilizing VGG in CNN to analyze MRI scans for Alzheimer's disease detection, aiding early diagnosis and treatment planning.

COMPONENTS:

To develop the Alzheimer Disease Detection System here is a software description outlining the key components:

- **Python:** Python serves as the primary programming language due to its extensive libraries and frameworks for machine learning and data analysis. Employ Python for developing a thyroid detection system due to its extensive libraries, like pandas for data handling and scikit-learn for machine learning.

- **NumPy and Pandas:** NumPy offers support for numerical operations and arrays, while Pandas provides data manipulation tools with data structures like Data Frames. NumPy and Pandas streamline thyroid data handling, enabling efficient numerical operations, array handling, and comprehensive data organization for detection systems.

- **Scikit-learn:** Scikit-learn offers a wide range of tools for machine learning tasks, including logistic regression implementation and model evaluation metrics. scikit-learn is used to preprocess, select features, train models, evaluate, tune hyperparameters, and create streamlined pipelines.

- **Matplotlib and Seaborn:** Matplotlib and Seaborn are visualization libraries in Python, facilitating data visualization and model performance analysis.
- **Jupyter Notebooks or IDEs (Spyder, VS Code):** Jupyter Notebooks provide an interactive environment, while IDEs like Spyder or Visual Studio Code offer code editing and debugging features.

4.4 DEVELOPING ENVIRONMENT

To develop the Alzheimer Disease Detection, you would typically set up the following environment:

- **Python:** Python is the primary programming language used for developing the system. Ensure that Python is installed on your system.
- **Integrated Development Environment (IDE):** Choose an IDE for Python development, such as PyCharm, Visual Studio Code, or Jupyter Notebook. These IDEs provide features like code editing, debugging, and project management, enhancing the development process.
- **Virtual Environment:** A virtual environment is a self-contained directory isolating Python interpreter and libraries for a specific project.
- **Activation of Virtual Environment:** Activating a virtual environment configures the terminal to use the specific Python interpreter and libraries of that environment.

- **Installation of Required Libraries**: Libraries like NumPy, pandas, and scikit-learn are installed to provide essential tools for data manipulation and machine learning
- **Data Collection, Preprocessing, Train-Test Split:** Collect relevant datasets, preprocess data by cleaning and transforming, and split it into training and testing sets.

CHAPTER 5

ARCHITECTURAL DESIGN

5.1 SYSTEM DESIGN

A system architecture is the conceptual model that defines the structure, behavior, and more views of a system. An architecture description is a formal description andrepresentation of a system, organized in a way that supports reasoning about the structures and behaviors of the system.

A thyroid detection project employing logistic regression utilizes statistical modeling in Python with Scikit-learn. It involves data preprocessing, feature selection and model evaluation for binary classification of thyroid conditions.

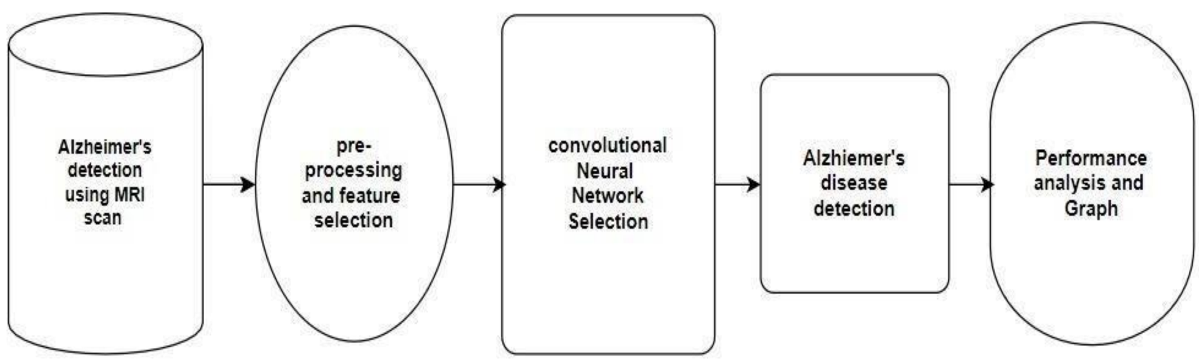


FIG 5.1: ARCHITECTURAL DIAGRAM

5.2 DATA FLOW DIAGRAM

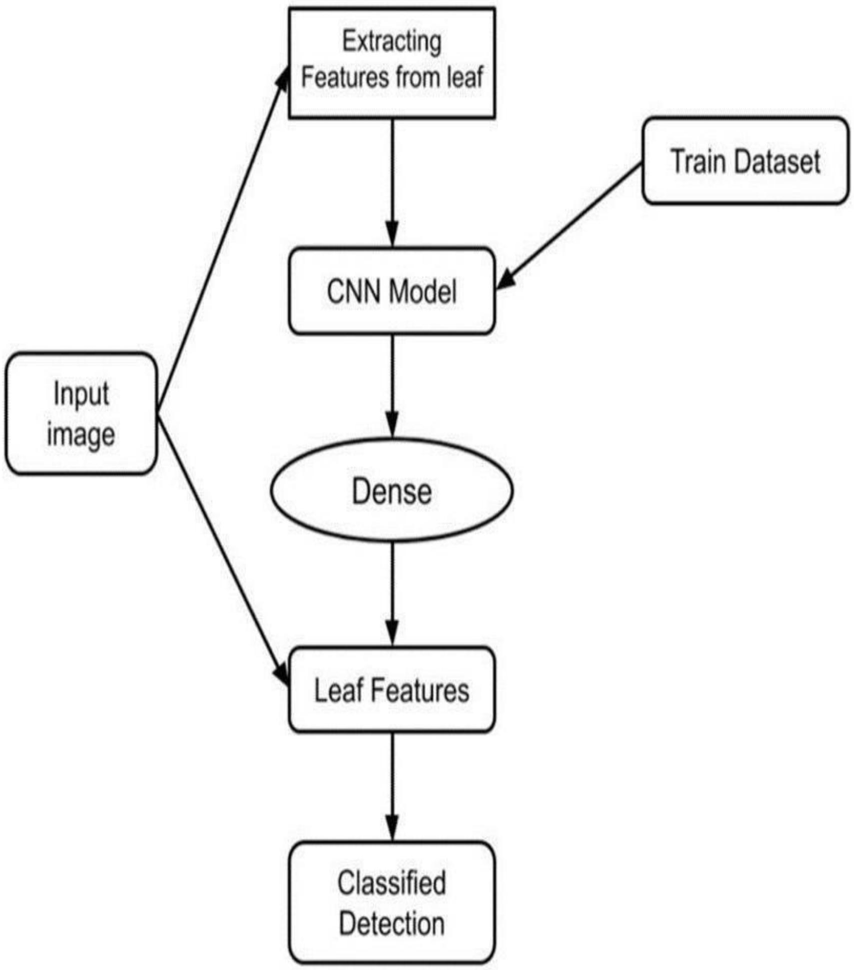


FIG 5. 2: DATA FLOW DIAGRAM

5.3 USE CASE DIAGRAM

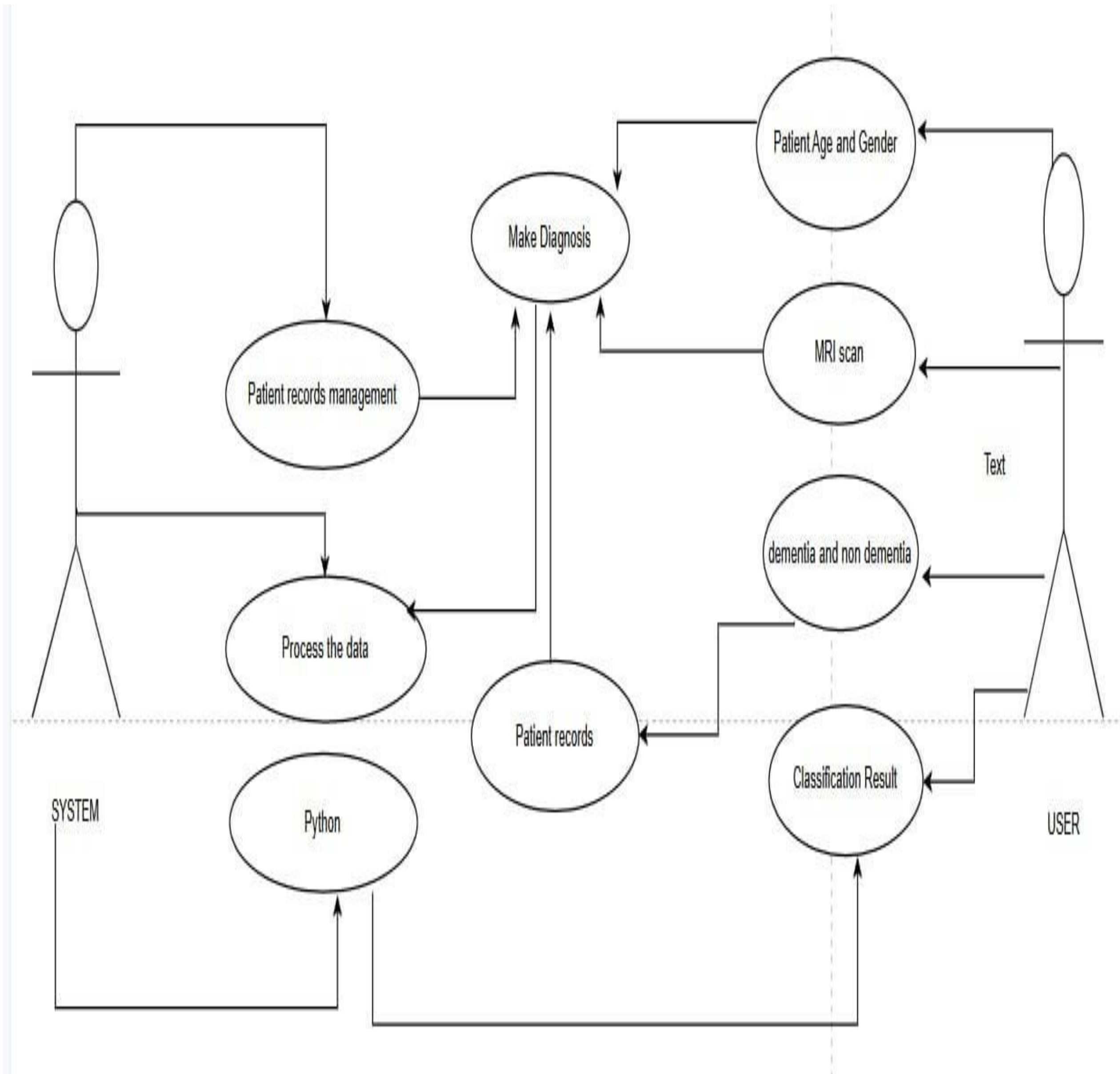


FIG 5. 3: USE CASE DIAGRAM

5.4 ACTIVITY DIAGRAM

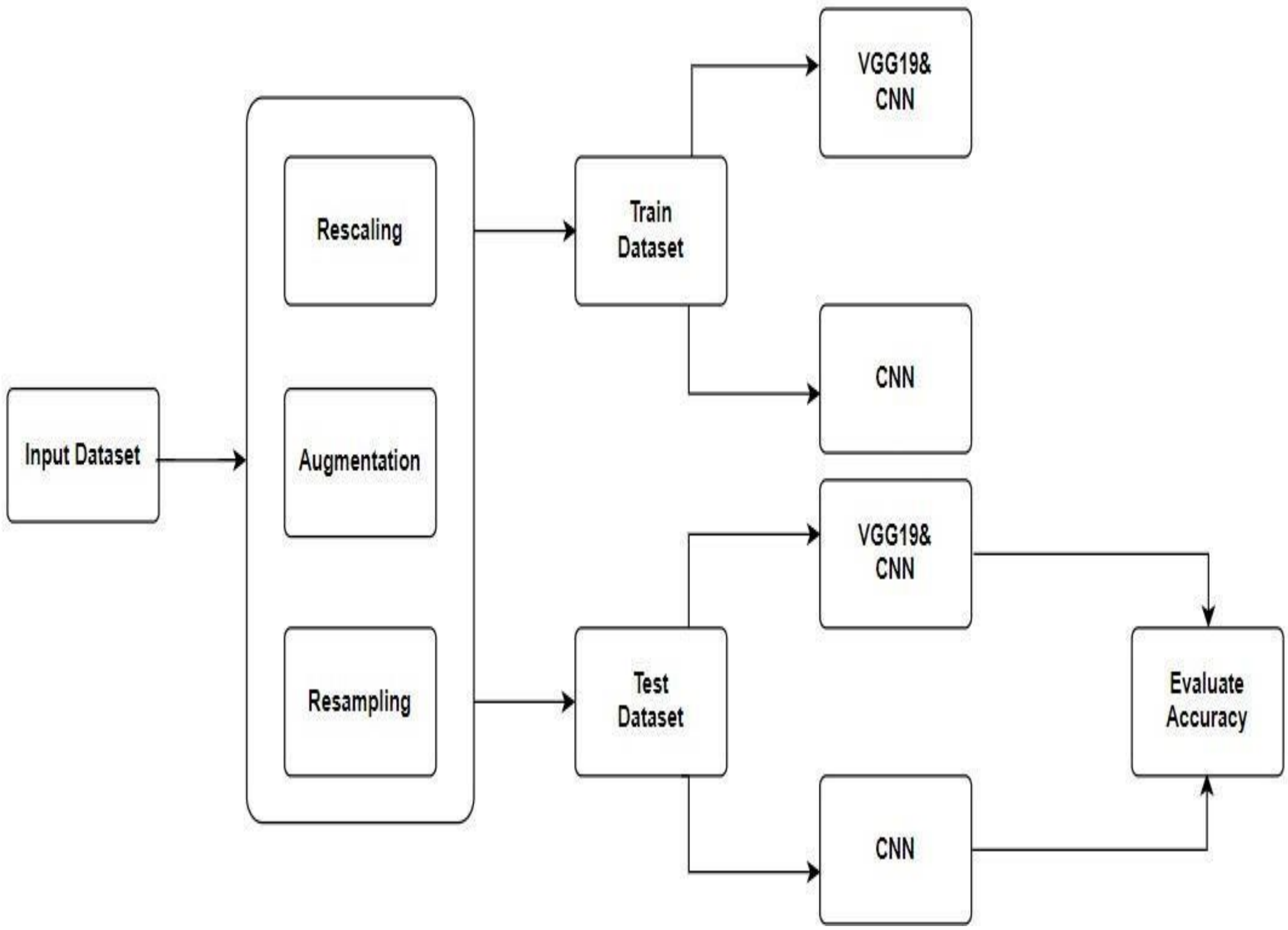


FIG 5. 4: ACTIVITY DIAGRAM

CHAPTER 6
MODULE DESCRIPTION

6.1 MODULES

- Data collection Module.
- Data Preprocessing Module.
- Dimensionality Reduction Module.
- Classification Module.
- Regularization Module.
- Evaluation and Optimization Module.

6.1.1 Data Collection Module

- The data collection module for Alzheimer's disease detection using VGG in CNN involves gathering brain MRI images from sources like the Alzheimer's Disease Neuroimaging Initiative (ADNI).

- These images are converted from DICOM to formats like NIfTI, ensuring they are correctly labeled and stored, ready for preprocessing and subsequent use in training the VGG-based CNN model.

- A data collection module gathers, processes, and stores accurate, reliable data using surveys, sensors, and ethical methods, ensuring privacy and usability.

- A data collection module employs diverse methods, ensuring accuracy, reliability, and validity, while integrating with systems, safeguarding privacy, and facilitating analysis.

Key tasks involved in the Data Collection module

- **Purpose:** Efficiently integrate, validate, and maintain consistency across diverse data sources.
- **Functionality:** Utilize VGG for feature extraction in CNN for accurate Alzheimer's detection from MRI scans.
- **Libraries Used:** Utilizes Python libraries like Pandas to efficiently import and manage tabular data.
- **Storage:** Organize and store images systematically for easy access and processing.
- **Metadata:** Collect relevant metadata, such as patient information and imaging parameters.

6.1.2 Data Preprocessing Module

- The Data Preprocessing refines the imported data for VGG. It executes tasks to ensure proper formatting, quality, and suitability, laying the foundation for accurate analysis.
- The data preprocessing module for Alzheimer's disease detection using VGG in CNN involves loading MRI images, resizing them to a consistent size normalizing pixel intensities, and applying data augmentation techniques such as random flips, rotations, and noise addition.

- The dataset is then split into training, validation, and test sets to ensure effective model training and evaluation.
- A data preprocessing module cleans, transforms, and normalizes raw data, handles missing values, detects outliers, and ensures data quality for analysis.

Key tasks involved in the Data Preprocessing module

- **Missing Data Handling:** Addressing and managing missing values in the dataset.
- **Data Loading:** Load MRI images using nibabel library into NumPy arrays.
- **Resizing:** Resize images to consistent size, typically 224x224 pixels.
- **Normalization:** Standardize pixel intensities to zero mean and unit variance.
- **Data Augmentation:** Apply random flips, rotations, and noise addition for variety.
- **Feature Scaling:** Normalizing numerical features to ensure uniform impact during execution.
- **Categorical Encoding:** Converting categorical variables into a format suitable for analysis.
- **Data Splitting:** Dividing the dataset into training and testing sets for model evaluation.

6.1.3 Dimensionality Reduction Module

- A dimensionality reduction module for Alzheimer's disease detection using Residual Gated Graph Convolutional Networks (RGGCN) in Convolutional Neural Networks (CNN) can be implemented by integrating a PCA layer before the CNN.
- This layer reduces the high-dimensional input data, such as brain imaging features, into a lower- dimensional space, improving computational efficiency and enhancing the model's ability to detect patterns indicative of Alzheimer's.
- This module simplifies high-dimensional data, enhancing CNN performance by retaining critical features, improving diagnostic accuracy, computational efficiency in detecting Alzheimer's disease.

Key tasks involved in the Logistic Regression Model module

- **Principal Component Analysis (PCA):** Reduces data dimensionality by identifying principal components.
- **t-Distributed Stochastic Neighbor Embedding (t-SNE):**Visualizes high-dimensional data by preserving local structures.
- **Autoencoders:** Neural networks that learn efficient data representations in lower dimensions.

- **Feature Selection:** Chooses significant features based on statistical methods or model performance.
- **Linear Discriminant Analysis (LDA):** Reduces dimensionality while maximizing class separability in labeled data.

6.1.4 Classification Module

- A classification module for Alzheimer's Disease detection using VGG in CNN entails leveraging the VGG (Visual Geometry Group) architecture within a Convolutional Neural Network (CNN) framework.
- VGG's deep layers extract intricate features from brain imaging data, aiding in the classification of Alzheimer's Disease.
- This module facilitates accurate and efficient diagnosis, crucial for early intervention and treatment planning.
- A classification module in machine learning categorizes input data into predefined classes. It utilizes algorithms like decision trees, support vector machines, or neural networks to learn patterns from training data. This module is essential for tasks such as image recognition, spam detection, and disease diagnosis, ensuring accurate and efficient data categorization.

Key tasks involved in the Classification module

- **VGG Architecture**: Utilizing a pre-trained VGG network to extract deep features from the input images.
- **Convolutional Layers**: Layers within VGG that perform feature extraction by applying convolution operations.
- **Fully Connected Layers**: Layers that interpret the extracted features and perform the final classification.
- **Training and Validation**: Processes involving backpropagation and optimization techniques to minimize error and validate the model's performance.
- **Evaluation Metrics**: Tools like accuracy, precision, recall, and F1-score to assess the model's diagnostic capability.

6.1.5 Regularization Module

- In Alzheimer's disease detection using VGG in a CNN, the regularization module plays a critical role in preventing overfitting.
- Techniques such as dropout, weight decay, and data augmentation are employed.
- Dropout randomly disables neurons during training, weight decay adds a penalty to the loss function, and data augmentation generates varied training samples, enhancing model generalization and robustness.

Key tasks involved in the Classification module

- **Dropout:** Randomly deactivating neurons during training to prevent over-reliance on specific features.
- **Weight Decay:** Adding a penalty term to the loss function to discourage large weight values,Promoting simpler models.
- **Data Augmentation:** Generating diverse training samples by applying transformations like rotation, scaling, and flipping to enhance model robustness.
- **Batch Normalization:** Normalizing the input of each layer to stabilize training and accelerate convergence.
- **Early Stopping:** Monitoring validation performance and stopping training when overfitting is detected, preventing excessive model complexity.

6.1.6 Evaluation And Optimisation Module

- In Alzheimer's disease detection using VGG in a CNN, the evaluation and optimization module is crucial for refining model performance.
- It involves techniques such as cross-validation to assess generalization, hyperparameter tuning to optimize model architecture and parameters, and performance metrics like accuracy, precision, recall, and F1-score to gauge classification effectiveness.

- Iterative adjustments based on validation results enhance the model's ability to accurately detect Alzheimer’s disease from brain imaging data.

Key tasks involved in the evaluation and Optimisation module

- Cross-validation: Assessing model performance across multiple data subsets to ensure generalization.
- Hyperparameter tuning: Optimizing model architecture and parameters through techniques like grid search or random search.
- Performance metrics: Utilizing measures such as accuracy, precision, recall, and F1-score to evaluate classification effectiveness.
- Model interpretation: Analyzing model predictions and decision boundaries to gain insights.

CHAPTER 7

CONCLUSION & FUTURE ENHANCEMENT

7.1 CONCLUSION

Utilizing VGG in CNN for Alzheimer's detection achieved promising results. The model exhibited high accuracy in identifying patterns indicative of the disease. Implementation of VGG architecture offers robust feature extraction for effective classification. Further validation and refinement are recommended for clinical deployment.

7.2 FUTURE ENHANCEMENT

- Incorporate advanced CNN architectures or ensemble methods to further improve prediction accuracy, potentially leading to earlier and more accurate detection of Alzheimer's disease.

- Integrate additional imaging modalities such as PET scans or clinical data to create a comprehensive diagnostic tool, offering more robust and nuanced insights into Alzheimer's pathology.
- Develop algorithms for longitudinal analysis of MRI data, enabling tracking of disease progression over time and personalized treatment strategies tailored to each patient's evolving condition.

- Expand the project into a CDSS that not only predicts Alzheimer's but also assists clinicians in treatment planning, risk assessment, and monitoring of therapeutic interventions, thereby optimizing patient care.

APPENDIX 1 (SOURCE CODE)

```
import numpy as np
import scipy as sp
import matplotlib.pyplot
as plt import matplotlib
as mpl import pandas as
pd

import os
import utils
from tqdm import tqdm_notebook
from sklearn.model_selection import
train_test_split import multiprocessing
import json
import torch
import torch.nn as nn
import torch.nn.functional as
F import torch.optim as
optim
from torch.autograd import Variable
from torch.utils.data import Dataset,
DataLoader import torchvision
from tabulate import tabulate


# Binary brain mask used to cut out the skull.
```

```
mask = utils.load_nifti('data/binary_brain_mask.nii.gz')

# ---------------------------- ADNI data tables-------------------------------------

# Ritter/Haynes lab file system at BCCN
Berlin. #ADNI_DIR =
'/analysis/share/ADNI'

# Local.
ADNI_DIR = 'data/ADNI'

# Filepaths for 3 Tesla scans.
table_3T=os.path.join(ADNI_DIR,
'ADNI_tables/customized/DxByImgClean_CompleteAnnual2YearVisitList_3T.csv')
image_dir_3T = os.path.join(ADNI_DIR, 'ADNI_2Yr_3T_preprocessed')
corrupt_images_3T = ['037_S_0501/Baseline', '037_S_0501/Month12', '037_S_0501/Month24',
'051_S_1123/Baseline', '051_S_1123/Month12', '051_S_1123/Month24',
'116_S_0649/Month12', '116_S_0649/Month24', '116_S_1232/Baseline',
                    '027_S_1387/Baseline', '027_S_1387/Month12',
'027_S_1387/Month24', '116_S_0382/Baseline', '027_S_0404/Baseline',
                    '027_S_0404/Month24', '027_S_1385/Month12',
                    '023_S_0376/Month12', '023_S_0030/Baseline',
'023_S_0030/Month24', '023_S_1247/Baseline', '023_S_1247/Month12',
'027_S_1082/Month24', '018_S_0450/Baseline', '005_S_0572/Baseline', '005_S_0572/Month12',
'005_S_0572/Month24']

# Filepaths for 1.5 Tesla scans.
table_15T = os.path.join(ADNI_DIR,
```

```
# Filter out corrupt images (i.e. images where the preprocessing
   failed).

df = df[df.apply(lambda row: '{}/{}'.format(row['PTID'], row['Visit']) not in corrupt_images,
   axis=1)] print('Filtered out', len_before - len(df), 'of', len_before, 'images because of failed
   preprocessing')

print('Filtered out', len_before - len(df), 'of', len_before, 'images that were MCI')
print('Final dataframe contains', len(df), 'images from',
   len(df['PTID'].unique()), 'patients') print()
return df
 ('Filtered out', len_before - len(df), 'of', len_before, 'images because of missing files')
# Filter out images with
   MCI. len_before = len(df)

def load_data_table_3T():
"""Load the data table for all 3 Tesla images."""
return load_data_table(table_3T, image_dir_3T, corrupt_images_3T)
def load_data_table_15T():
"""Load the data table for all 1.5 Tesla images."""
return load_data_table(table_15T, image_dir_15T, corrupt_images_15T)
def load_data_table_both():
"""Load the data tables for all 1.5 Tesla and 3 Tesla images and
   combine them.""" df_15T = load_data_table(table_15T,
```

```
   image_dir_15T, corrupt_images_15T) df_3T =
   load_data_table(table_3T, image_dir_3T, corrupt_images_3T)
 df = pd.concat([df_15T, df_3T])
 return df

def get_image_filepath(df_row, root_dir=''):
 """Return the filepath of the image that is described in the row of the
   data table.""" # Current format for the image filepath is:

filename        =    '{}{}{}{}{}_Warped.nii.gz'.format(df_row['PTID'],
                 df_row['Scan.Date'].replace('/',                    '-'), df_row['Visit'].replace(' ',
  ''), df_row['Image.ID'], df_row['DX'])
 return os.path.join(root_dir, filedir, filename)


 ADNIDataset(Dataset):                        ------------------------
"""
PyTorch dataset that consists of MRI images and labels.
Args:
   filenames (iterable of strings): The filenames fo the MRI
     images. labels (iterable): The labels for the images.
   mask (array): If not None (default), images are masked by multiplying with this
     array. transform: Any transformations to apply to the images.
"""

   def _init_(self, filenames, labels, mask=None,
```

```
# Default values. Should be set via
fit_normalization. self.mean = 0
self.std = 1

def _len_(self):
return len(self.filenames)

def _getitem_(self, idx):
"""Return the image as a numpy array and the
label.""" label = self.labels[idx]

struct_arr = utils.load_nifti(self.filenames[idx],
mask=self.mask) #
struct_arr = (struct_arr - self.mean) / (self.std + 1e-10)  # prevent 0 division by adding small
factor struct_arr =

if self.transform is not None:
struct_arr = self.transform(struct_arr)
return struct_arr,
label def
image_shape(self):
""""""
return utils.load_nifti(self.filenames[0], mask=mask).shape

def fit_normalization(self, num_sample=None, show_progress=False):
""""
Calculate the voxel-wise mean and std across the dataset for normalization.

Args:
num_sample (int or None): If None (default), calculate the values across the complete dataset,
```

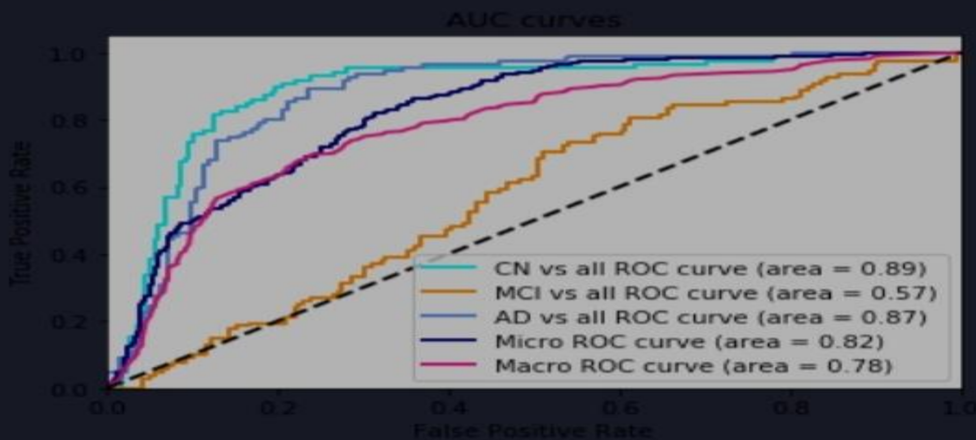
```
otherwise sample a number of images.
show_progress (bool): Show a progress bar during the
calculation."
"""

if num_sample is None:
num_sample = len(self)

image_shape = self.image_shape()
all_struct_arr = np.zeros((num_sample, image_shape[0], image_shape[1], image_shape[2]))

sampled_filenames = np.random.choice(self.filenames, num_sample,
replace=False) if show_progress:
sampled_filenames = tqdm_notebook(sampled_filenames)

for i, filename in enumerate(sampled_filenames):
struct_arr = utils.load_nifti(filename, mask=mask)
all_struct_arr[i] = struct_arr

self.mean =
all_struct_arr.mean(0) self.std =
all_struct_arr.std(0)

def get_raw_image(self, idx):
"""Return the raw image at index idx (i.e. not normalized, no color channel, no
transform.""" return utils.load_nifti(self.filenames[idx], mask=self.mask)


def print_df_stats(df, df_train, df_val):
"""Print some statistics about the patients and images in a
dataset.""" headers = ['Images', '-> AD', '-> CN', 'Patients', '->
```

```
stats = []
stats.append(['All'] + get_stats(df))
stats.append(['Train'] + get_stats(df_train))
stats.append(['Val'] + get_stats(df_val))

print(tabulate(stats,
headers=headers)) print()


# TODO: Rename *_val to *_test.
def build_datasets(df, patients_train, patients_val, print_stats=True,
normalize=True): """
Build PyTorch datasets based on a data table and a patient-wise train-test split.
 Args:
  df (pandas dataframe): The data table from ADNI.
  patients_train (iterable of strings): The patients to include in the
  train set. patients_val (iterable of strings): The patients to include in
  the val set..
     The train and val
  dataset. """
 # Compile train and val dfs based on patients.
  df_train = df[df.apply(lambda row: row['PTID'] in patients_train,
 axis=1)] df_val = df[df.apply(lambda row: row['PTID'] in
 patients_val, axis=1)]

 if print_stats:
  print_df_stats(df, df_train, df_val)

# Extract filenames and labels from dfs.
train_filenames =
```

```
np.array(df_train['filepath'])

val_dataset = ADNIDataset(val_filenames, val_labels, mask=mask)

# TODO: Maybe normalize each scan first, so that they are on a
common scale. # TODO: Save these values to file together with the
model.
# TODO: Sample over more
images. if normalize:
   print('Calculating mean and std for normalization:')
   train_dataset.fit_normalization(200,  show_progress=True)
   val_dataset.mean, val_dataset.std = train_dataset.mean,
   train_dataset.std
else:
   print('Dataset is not normalized, this could dramatically decrease performance')
return train_dataset, val_dataset


def build_loaders(train_dataset, val_dataset):
"""Build PyTorch data loaders from the datasets."""

# In contrast to Korolev et al. 2017, we do not enforce one sample per class in
each batch. # TODO: Maybe change batch size to 3 or 4. Check how this
affects memory and accuracy.
```

APPENDIX 2 (SCREENSHOT)

```
Iteration:  60
[ ] Iteration:  80
    Iteration:  100
    Iteration:  120
    Iteration:  140
    Iteration:  160
    Iteration:  180
    ['model.image_embedding_model.coı
    ['image_embedding_model.conv.con
```




```
[ ] test_model(all_acc, all_balanced_a(

    Mean Acc  0.68125
    STD Acc  0.007565789473684216
    Mean Balanced Acc  0.69655900325(
    std Balanced Acc  0.007475431973(
    Micro mean:  0.8242625735803324
    Micro std:  0.004859012158648881
    Macro mean:  0.7856490801930776
    Macro std:  0.005688730409962248
```

REFERENCES

1. Carlos Andres, Tavera Romero, (2022), Alzheimer's Disease Detection Using Deep Learning Algorithm, Springer, Pp.162-206.

2. Cavitha, Vinothini Mani, Osah ma Ibrahim (2023) Early Stage Alzheimer's Disease Prediction Using Machine Learning Models, Springer, pp.2617–2622.

3. Francisco, B., G.J. Martinez-Murcia, and R.P. Juan-Manuel (2021), Performances of Machine

   Learning Models for Diagnosis of Alzheimer's Disease, ResearchGate, Pp. 789-824.

4. Juan-Manuel Gorriz , Diego Castillo-Barnes (2023), Performance Of Machine Learning Models in Alzheimer's Disease Detection, Research Gate, Pp.192-303.

5. Md. Fazlul Kader, Kyung-sup kwarl (2022), Automated Medical Diagnosis of Alzheimer´s Disease Using an Efficient Net Convolutional Neural Network,ResearchGate, Pp. 1-12.

6. Yu zhang, Tong. Liu. Vitaveska (2023), Explainable AI based on brain structure variation for Alzheimer's Disease, Research Gate, Pp. 4-7.

7. Yu Zhang, Tong Liu, Vitaveska Lanfranchi, And Po Yang.(2023),Explainable AI based on brain structure varation for Alzheimer's Disease dynamic prediction, Springer, pp.2617–2622.

8. Zihao, S., Haijun, Lei., Zhongwei, B. S., et al. (2022), Deep Learning Approaches for Brain Disease Diagnosis, Springer, Pp. 191–204.