

**Sino-British Collaborative Programme,**

**Oxford Brookes University & Chengdu University of Technology**

**BSc (Hons) Software Engineering Programme**

**BSc (Single Honours) Degree Project**

Programme Name: BSc (Hons) Software Engineering Programme

Module code: CHC 6096

Surname: Liu

First Name: Yue

Project Title: Ensemble learning for the classification of Alzheimer disease

Student No: 201918020206

Supervisor: Grace Ugochi. Nneji

2™ Supervisor -

(if applicable)

Date submitted: /2023

*A report submitted as part of the requirements for the degree of BSc (Hons) in Software Engineering*

*At*

*Oxford Brookes University*

**Student Conduct Regulations:**

**Ensemble learning for the classification of Alzheimer disease**

Liu Yue

201918020206

# Abstract

Alzheimer's disease is the most common type of dementia and the majority of sufferers are elderly. With the global aging process, the number of people with Alzheimer's disease is growing rapidly worldwide, and with this comes the pressure on the healthcare economy due to the increased cost of treatment. The use of magnetic resonance imaging (MRI) for the early diagnosis of Alzheimer's disease is therefore a key part of the solution to this problem. This project uses ensemble learning to combine three models which are ResNet, AlexNet, and MobileNet to avoid the potential failures of individual models and to combine the benefits of individual models. The data set of this project has a total of 33984 cross-sectional MRI images of Alzheimer's disease. The ratio of the training set, validation set, and test set is 6:2:2. The utilization of this ensemble model achieves an identification accuracy of 98.2%. The project will assist medical practitioners in applying automated systems in the identification of Alzheimer’s disease, thus saving more lives, practitioners’ time, and medical resources in medical centers.

# Keywords

Ensemble learning, ResNet, AlexNet, MobileNet, Deep learning, Alzheimer’s disease, Image recognition.

# Abbreviations

**DL** Deep Learning

**ResNet** Residual Networks

**GUI** Graphical User Interfaces

**AD** Alzheimer's disease

**MRI** Magnetic Resonance Imaging

**IDE** Integrated Development Environment

**SVM** Support Vector Machines

**AUC** Area Under Curve

**DEMNET** Dementia Network

**BAGGING** Bootstrap Aggregation

**MCI**  Mild Cognitive Impairment

**AI** Artificial Intelligence

# Glossary

**Deep learning:** Deep learning is part of a broader family of machine learning methods based on artificial neural networks with representation learning.

**Alzheimer’s Disease:** Alzheimer’s disease is a type of brain disorder that causes problems with memory, thinking and behavior. This is a gradually progressive condition.

**Ensemble learning:** Ensemble learning is a general meta-approach to machine learning that seeks better predictive performance by combining the predictions from multiple models.

**ResNet:** Residual Network is a specific type of neural network that was introduced in 2015 by Kaiming He, Xiangyu Zhang, Shaoqing Ren and Jian Sun in their paper “Deep Residual Learning for Image Recognition”.

**AlexNet:** AlexNet is a classic convolutional neural network architecture. It consists of convolutions, max pooling and dense layers as the basic building blocks.

**MobileNet:** Efficient Convolutional Neural Networks for Mobile Vision Applications. MobileNets are based on a streamlined architecture that uses depth-wise separable convolutions to build light weight deep neural networks.

**Graphical User Interface:** Graphical user interface is a digital interface in which a user interacts with graphical components such as icons, buttons and menus.

**Deep Neural network:** By using a technique called Deep Neural Networks, which mimics the behavior of the human brain, the researchers were able to train it to be more discriminating than previous methods.

**Fine-tuned:** Deep learning requires continuous training in the deep network to update the model parameters (weights) to fit the model that can achieve the expected results.

**F1-Score:** It is an index used to measure the accuracy of the binary model in statistics. It also takes into account the accuracy rate and recall rate of the classification model. The F1 score can be regarded as a harmonic average of model accuracy and recall, with a maximum value of 1 and a minimum value of 0.

**Convolutional layer:** Each Convolutional layer in the convolutional neural network is composed of several convolutional units, and the parameters of each convolutional unit are optimized by the backpropagation algorithm. The purpose of the convolution operation is to extract different features of the input. The first convolution layer may only extract some low-level features such as edges, lines and corners, etc., and more layers of networks can iteratively extract more complex features from low-level features.

**Bagging:** Bootstrap aggregation, is the ensemble learning method that is commonly used to reduce variance within a noisy dataset.

# Acknowledgment

I would like to thank my supervisor, Dr. Grace Ugochi Nneji for her professional guidance and advice. Without her long-term support in all aspects of this project and her patience and dedication, this thesis would not have reached this level of completion. I am also grateful to the project leader, Joojo Walker for his teaching and the other staff, and to Oxford Brookes University and the Chengdu University of Technology for the valuable opportunities they have provided for me. Finally, I would like to thank the community and network for the resources provided, and my family and friends who have supported me and encouraged me. Thank you to all of you for making this project a success.

**Table of Contents**

[Abstract 4](#_Toc15560)

[Keywords 4](#_Toc29063)

[Abbreviations 5](#_Toc13049)

[Glossary 6](#_Toc14663)

[Acknowledgment 8](#_Toc18785)

[Chapter 1 Introduction 11](#_Toc26422)

[1.1. Background 11](#_Toc15802)

[1.1.1. Challenges 12](#_Toc6865)

[1.2. MRI 14](#_Toc20372)

[1.3. Deep Learning 16](#_Toc17252)

[1.4. Convolutional Neural Network 17](#_Toc3868)

[1.4.1. Input Layer 17](#_Toc24113)

[1.4.2. Convolutional Layer 17](#_Toc9001)

[1.4.3. Pooling Layer 18](#_Toc8906)

[1.4.4. Fully Connected Layer 18](#_Toc18618)

[1.4.5. Activation Function 19](#_Toc30495)

[1.4.6. Loss Function 19](#_Toc8003)

[1.4.7. Evaluation 20](#_Toc4366)

[1.5. Ensemble model 21](#_Toc12409)

[1.6. Aim 21](#_Toc6528)

[1.7. Objectives 22](#_Toc22407)

[1.8. Project Overview 22](#_Toc1360)

[1.8.1. Scope 22](#_Toc2761)

[1.8.2. Audience 23](#_Toc14996)

[Chapter 2 Background Review 24](#_Toc25728)

[2.1. Single CNN Model: 24](#_Toc3328)

[2.2. Multi-model : 25](#_Toc27175)

[2.3. Ensemble Models 25](#_Toc12913)

[Chapter 3 Methodology 28](#_Toc20290)

[3.1. Dataset 28](#_Toc6386)

[3.1.1. Dataset preprocessing 28](#_Toc6118)

[3.2. Ensemble Model 29](#_Toc4528)

[3.2.1. Finetuned Residual Network (ResNet) 29](#_Toc24654)

[3.2.2. Finetuned AlexNet 32](#_Toc8372)

[3.2.3. Finetuned MobileNet 34](#_Toc31280)

[3.2.4. Ensemble Model 36](#_Toc32614)

[3.3. Performance Evaluation Metrics 38](#_Toc21942)

[3.4. evaluations 38](#_Toc4127)

[3.5. Technology(explain and table 38](#_Toc6984)

[3.6. Project Version Management 39](#_Toc13142)

[Chapter 4 Results 41](#_Toc17538)

[4.1. Hyperparameters setting 41](#_Toc13100)

[4.2. Evaluations of the project model 41](#_Toc13841)

[4.2.1. Acc 41](#_Toc24676)

[4.2.2. Loss 41](#_Toc10577)

[4.2.3. precision 41](#_Toc5129)

[4.2.4. Recall 41](#_Toc27604)

[4.2.5. F1 41](#_Toc9037)

[4.3. GUI 41](#_Toc27716)

[Chapter 5 Professional issue 41](#_Toc32620)

[5.1. Project Management 41](#_Toc1854)

[5.1.1. Activities 41](#_Toc18838)

[5.1.2. Schedule 42](#_Toc23575)

[5.1.3. Project Data Management 42](#_Toc9000)

[5.1.4. Project Deliverables 42](#_Toc28840)

[5.2. Risk Analysis 43](#_Toc1218)

[5.3. Professional Issues 43](#_Toc25745)

[Chapter 6 Conclusion 45](#_Toc31486)

[References 46](#_Toc14236)

# Introduction

## Background

Alzheimer's disease (AD) involves the part of the brain that controls thought, memory and language, and is a progressive neurodegenerative disease with an insidious onset, the most common type of dementia with no known cause [1]. The majority of people with Alzheimer's disease are elderly. With the global ageing process, the number of people with Alzheimer's disease is likely to increase by more than 100 million worldwide, and the increasing burden of its prevalence may exceed the capacity for manageable diagnosis and management. At the same time, AD often has a long latency and prodromal period, which will place a huge burden on patients as well as their families when they progress to an advanced stage. At the same time, the cost of treatment will increase significantly, as well as the fact that there is no curable drug for AD, which may impose a heavy burden on society in terms of medical expenses, and material resources. It is therefore of utmost importance to diagnose AD early and to slow down or stop the progression of the disease through early intervention. The earliest clinical means of diagnosing AD can only rely on the patient's performance on cognitive tests, and the early symptoms of AD are so similar to the decline in memory capacity caused by normal ageing that early diagnosis is difficult to achieve. With advances in medical imaging technology and the discovery of some biological markers associated with AD, the degree of brain atrophy can be an important basis for the diagnosis of AD, and Magnetic Resonance Imaging (MRI) technology can accurately capture changes in the volume of brain areas caused by brain atrophy, thus helping in the diagnosis of AD [2]. The use of deep learning and other methods to analyse MRI images has been widely studied and will become one of the breakthroughs in the early diagnosis of AD, particularly Convolutional Neural Networks (CNNs) are capable of learning and extracting representative features from raw input images, have shown promise in the accurate identification of Alzheimer's disease based on MRI images.[2]. A sample of AD image is shown in Figure 1

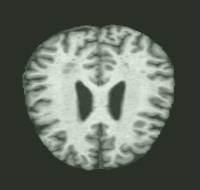


Figure 1: A sample of Alzheimer’s image

### Challenges

* Category imbalance:

It's possible that there are more examples of one type of Alzheimer's disease than the others in the distribution of the disease in medical photos, which will result in biased model training. Particularly, there are comparatively few research on the applicability of computer-aided diagnosis of AD in diagnosing various progressive stages of AD, with the majority of the literature focusing more on binary categorization or diagnosis of AD and MCI phases. This also includes dataset bias, which can happen when particular patient groups are over- or under-represented. This may result in deceptive ensemble learning models that are biased towards particular patient populations, which could limit the model's ability to be generalized to other cohorts.

* Feature extraction:

For accurate classification, it is crucial to extract useful characteristics from medical images, but this can be difficult given the complexity and high dimensionality of medical images. Brain MRI pictures, for instance, may be vulnerable to image quality fluctuation due to variances in acquisition techniques, hardware, and software utilized by various centers, which may result in variations in image resolution, contrast, and noise.

* Hyper-parameter tuning:

Ensemble learning algorithms contain multiple models with different hyper-parameters, which makes hyper-parameter tuning a challenge. Setting optimal values for hyperparameters such as learning rate, dropout rate or batch size is an iterative process that requires care to avoid over-fitting.

* Ensemble complexity:

Choosing the right base model for an ensemble can be a challenge. Different types of models may perform better for different types of Alzheimer's disease images, so selecting the most appropriate model requires domain expertise and careful evaluation. And the models used in the integration may differ in accuracy or generalisation behaviour, making the performance of the integration difficult to control. As the number of underlying models in the integration increases, the complexity of the overall system increases. while overfitting occurs when the model is too complex and fits the noise in the data. Ensuring that the ensemble is properly optimised and calibrated can be challenging.

* High cost:

Integration models require significant computational power to produce results, which may be limited by available computational resources, and ensemble learning typically requires a longer training time than individual models, as multiple models need to be trained and combined. This is particularly challenging for large and complex datasets, such as those related to Alzheimer's disease.

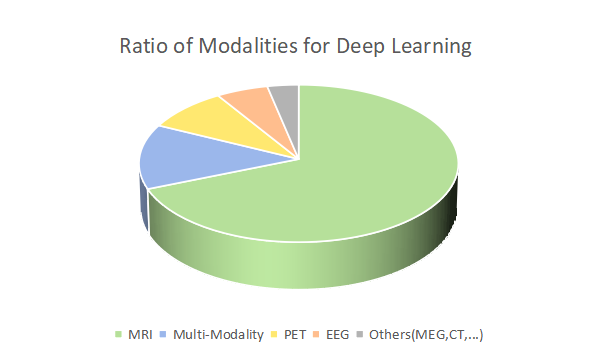
## MRI

Imaging detection is now a practical, affordable, reliable, and accurate diagnostic technique. When AD occurs, it may be identified by looking for certain observable structural abnormalities on medical images, such as brain atrophy, white matter degeneration, widening and depth of sulci, narrowing of gyrus, enlargement of ventricles, thinning of cerebral cortex, and so on. Among these, MRI technology has made several reasonably well-liked technological advancements, including functional MRI (fMRI) and PET-MRI. Due to its higher soft tissue contrast, multiplanar imaging capability, and lack of ionizing radiation compared to other imaging techniques, magnetic resonance imaging (MRI) has gained widespread acceptance as a non-invasive imaging modality for detecting AD. Table 1 provides detailed information on different imaging technologies. [4-7]

With the continuous development of science and technology, various neuroimaging data are also being generated in large quantities. Other imaging methods, such as computed tomography (CT) and PET, are also used for the diagnosis of Alzheimer's disease, but there are some limitations. Due to the use of ionizing radiation, the soft tissue contrast of CT is inferior to that of MRI, and therefore cannot be used for functional imaging. On the other hand, PET imaging provides functional and molecular data with high sensitivity, but is limited by radiation exposure and costly. Compared to these modes, MRI has fewer limitations and is a multifunctional imaging mode that can provide multiple anatomical and functional information. Figure 2 shows the ratio of MRI images in deep learning mode medical images. [8]

|  |  |  |
| --- | --- | --- |
| Categories | Mechanism | Diagnostic features |
| MRI | High-resolution structural images of the brain reflect abnormal morphological changes in brain tissue. | Cortical thinning, gray matter loss, and hippocampal atrophy in specific anatomical regions (inner and outer frontal parietal lobes and posterior cingulate cortex) of MCI/AD patients. |
| fMRI | Detect brain activity by detecting changes in oxygen consumption of brain nerve cells under resting or external stimuli. | Under task stimuli related to memory, emotion, and cognition, there was a significant decrease in the activation status of the medial frontal and parietal lobes in MCI/AD patients; Under resting state, MCI/AD patients exhibit abnormal activity in brain regions related to memory, executive function, and attention, such as the hippocampus, parahippocampal gyrus, and left inferior frontal gyrus, as well as in default network core brain regions such as the posterior cingulate gyrus, anterior cuneiform lobe, and left thalamus. |
| DTI | The Brownian motion of water molecules is traced to probe the directional condition of nerve fibres in the white matter of the brain, to detect the degree of freedom and diffusion of water molecules in the direction of diffusion, and to assess the integrity of the white matter of the brain and the functional state of the brain. | Impaired integrity of hippocampus, frontotemporal parietal lobe, cingulate gyrus, geniculate corpus callosum and hooked bundle in MCI/AD patients with abnormal diffusion of water molecules. |
| PET | Fluorodeoxyglucose PET (FDG-PET) was used to detect changes in glucose metabolism. | Decreased FDG uptake in the temporoparietal cortex in patients with cognitive impairment. Metabolic abnormalities in brain regions can be detected about 10 years earlier. |

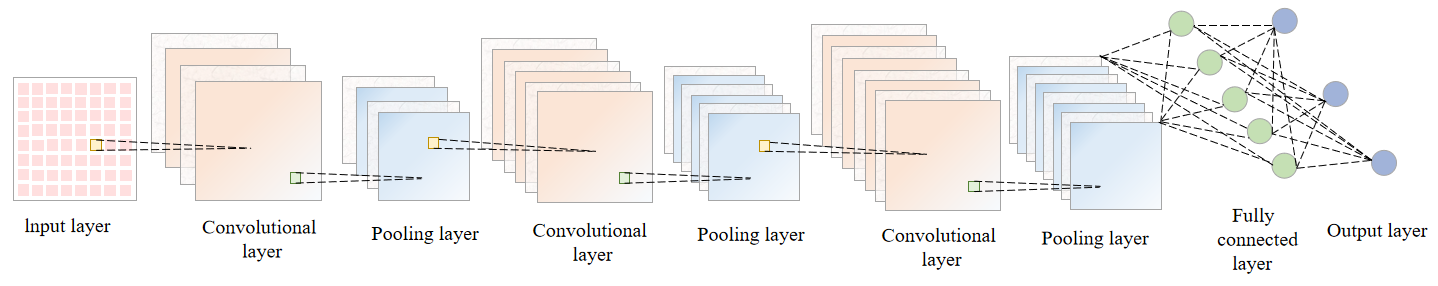
**Table 1 Mechanisms of Various Neuroimaging and AD Diagnostic Features**



**Figure 2 Ratio of Modalities for Deep Learning**

## Convolutional Neural Network

Convolutional Neural Network (CNN) is one of the most widely used neural networks in deep learning. It can learn image features from a large amount of data through a series of operations such as convolution, non-linear activation and pooling, and has achieved great success in image classification, object detection, semantic segmentation and other fields. The CNN contains many convolution and pooling layers. Each convolution layer has multiple convolution cores, also known as filters or feature detectors. Full connectivity and normalization layers are often used in the structure of CNNs [9,11,12]. The Convolutional Neural Network Architecture is shown in Figure 3.



**Figure 3 : The Convolutional Neural Network Architecture**



### Input Layer

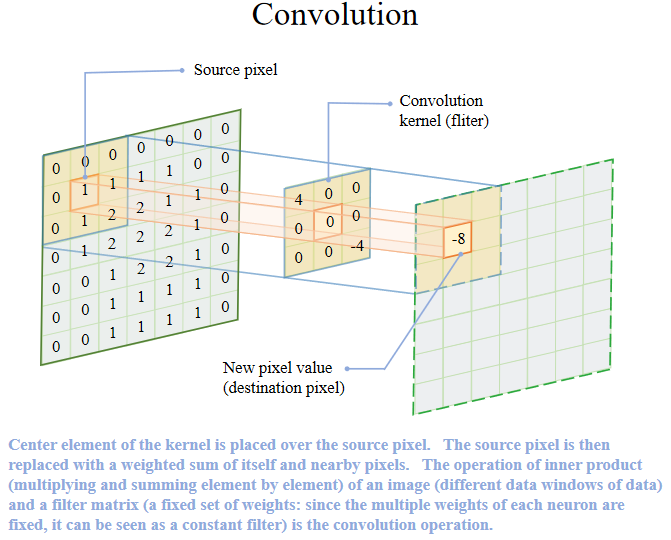
The Input layer is the first layer of the neural network and is where the data enters the network. This layer's primary job is to collect input data from the outside world, convert it into a format that the network can use, and then transfer it to the following layer for processing. In a CNN processing an image, the input layer generally represents a matrix of pixels of a picture. A picture can be represented by a three-dimensional matrix. The length and width of the 3D matrix represent the size of the image, while the depth of the 3D matrix represents the colour channel of the image. For example, a black and white image has a depth of 1, while in RGB colour mode the image has a depth of 3. [10,13]

### Convolutional Layer

The core of a convolutional neural network is the convolutional layer, and the core part of the convolutional layer is the convolutional operation. Each convolution kernel performs a convolution operation on the input of the previous layer and outputs the corresponding feature map. The convolution kernel is equivalent to a learnable filter that extracts local features of the graph over different image regions at different locations and sizes. The Convolutional Layer Architecture is shown in Figure 4.

In this operation, the following parameters are present:

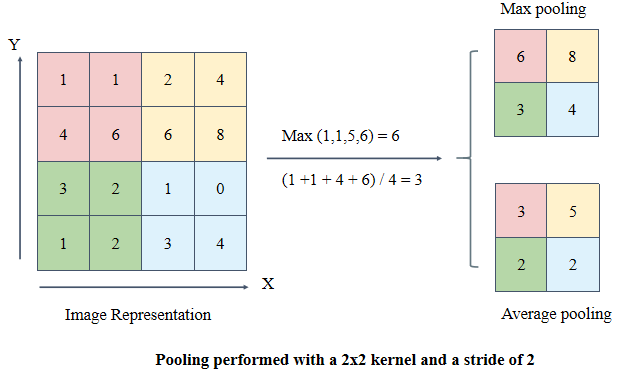
* Depth: the number of neurons, which determines the thickness of the depth of the output. It also represents the number of filters.
* Step stride: determines how many steps to slide to the edge.
* Zero-padding: the outer edge is supplemented by a number of circles of zeros to facilitate sliding from the initial position to the end position in steps.



**Figure 4 : The Convolutional Layer Architecture**

### Pooling Layer

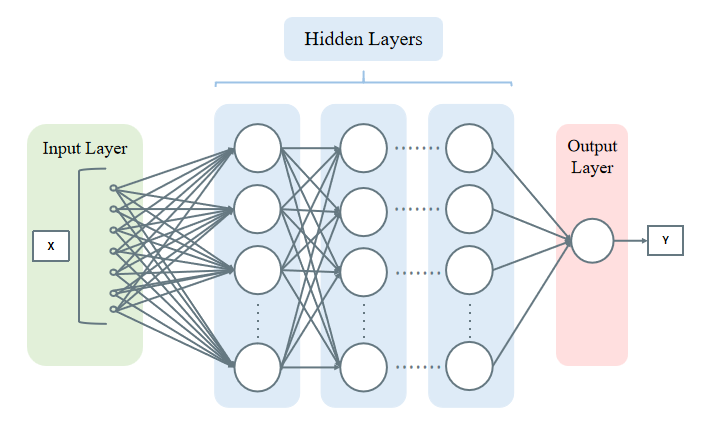
In addition to the convolutional layer, the pooling layer in the CNN also has an important role to play. Pooling mainly performs a downsampling operation, i.e. the size of the image feature map is reduced in order to reduce the computational effort and number of parameters of the subsequent classifier. The pooling operation takes the overall statistical features of the neighbouring regions of the input matrix as the output of that position, mainly Average Pooling, Max Pooling, etc. By pooling, noise and irrelevant information in the feature map can be effectively reduced and, more importantly, pooling preserves the translation invariance of the image. Hyperparameters of the pooling layer: pooling window and pooling step size. The pooling operation can also be thought of as a convolution operation [10,13,14]. The Pooling Layer Architecture is shown in figure 5



**Figure 5 : The Pooling Layer Architecture**

### Fully Connected Layer

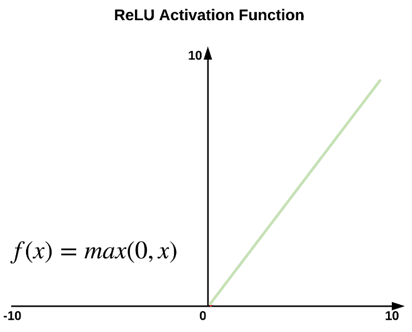
The final classification outcome is often provided by 1 to 2 fully connected layers at the conclusion of the CNN after numerous rounds of convolutional layer and pooling layer processing. The most fundamental layer in neural networks is known as the Fully linked Layer, sometimes referred to as the Dense Layer since every node is linked to every other node in the preceding layer. A fully connected layer's primary job is to output the input to the following layer by performing linear transformation and nonlinear activation on the characteristics of the previous layer's input. Different sizes of fully connected layers can be selected in model design depending on the task's needs. It is subject to overfitting and needs extensive parameter training. The whole connection layer also consumes a significant amount of computational power and storage space when processing huge amounts of picture data [9,10,15]. The Fully Connected Layer Architecture is shown in Figure 6.



**Figure 6 : The Fully Connected Layer Architecture**

### Activation Function

Activation functions are an integral part of Convolutional Neural Networks (CNNs) and are used to introduce non-linearity to the output of neural network models. To put it simply, activation functions control a neuron's output by converting its input signal to an output signal. In essence, it is a mathematical function that adjusts a neuron's output to establish a threshold for activation. The importance of activation functions in CNNs is shown by the fact that without them, a neural network would be unable to recognize and learn from linearly inseparable patterns in the input data. Some of the commonly used activation functions are Sigmoid, Tanh, ReLU, Leaky ReLU, and Softmax. Sigmoid and Tanh have smooth curves that make them suitable for binary classification tasks, while ReLU and its variants are better suited for deep learning architectures due to their computational efficiency [10.16]. The ReLU function image is shown in Figure 7.



**Figure 7 : The ReLU activation function**

### Loss Function

A essential part of a Convolutional Neural Network (CNN), the Loss Function updates model parameters to reduce prediction error, which is a critical step in the optimization process. Simply described, it is a mathematical function that calculates the discrepancy between output that was expected and output that was actually produced. The aim of the Loss Function is to find the optimal set of weights that can help the CNN model make accurate predictions. Some of the common Loss Functions are: Mean Squared Error (MSE), Binary Cross-Entropy, Categorical ross-Entropy, Hinge Loss. [10.17]

### Evaluation

Model evaluation is to evaluate the performance of a trained model, which is an indispensable part of the model development process. It helps to find out how well the best models for expressing data and selected models will perform in the future. Confusion matrix, also known as error matrix, is a visual tool in supervised learning. It is mainly used to compare the classification result of a model with the real information of an instance. It is an N\*N matrix. N is the number of classifications. Each row in the matrix represents the prediction category of an instance, and each column represents the real category of an instance. Many of the indexes used in the evaluation of a model of machine learning come from the operation of the result of the confusion matrix. Common classification model evaluation indicators are Precision, Recall, F-score, Accuracy, and so on. Table 2 shows the Confusion Matrix. Table 3 shows the Explanation of Model Evaluation Indicators.

|  |  |  |
| --- | --- | --- |
|  | **Positive** | **Negative** |
| **True** | True Positive | True Negative |
| **False** | False Positive | False Negative |

**Table 2 the Confusion Matrix**

|  |  |  |
| --- | --- | --- |
| Indicators | Formulas | Meaning |
| Accuracy |  | The proportion of all correctly judged results to total observations in the classification model |
| Precision |  | The proportion of model prediction pairs among all results where model observation is positive |
| True positive rate, TPR, Sensitivity, Recall |  | The proportion of model prediction pairs among all results where the true value is positive |
| Specificity |  | The proportion of model prediction pairs among all results where the true value is negative |
| F1-Score |  | F1 is defined as harmonic mean based on the harmonic mean of the accuracy and recall rates. |

**Table 3 the Explanation of Model Evaluation Indicators**

## Ensemble model

In convolution neural network (CNN), Ensemble Learning refers to a technology that combines predictions from multiple models into a single prediction by building and combining multiple learners. Its motivation is to improve the accuracy, stability and generalization of the model. Even if a weak classifier has a wrong prediction, other weak classifiers can correct the error. Applications of Ensemble Learning in CNN include Bagging, Boosting, and Stacking. Bagging trains multiple CNN models by randomly resampling the training data and weights their predictions. Boosting trains the CNN model repeatedly, focusing on the samples from the previous round of model prediction errors, and weighting the predictions from multiple models. Stacking, on the other hand, trains several CNN models with different structures and uses their middle features to train higher-level classifiers to improve the generalization performance and prediction accuracy of the models [18]. The model combines the output probabilities of three CNN models on average: ResNet, AlexNet, and MobileNet. These three models have different architectures and pre trained weights, providing diversity and complementarity in feature extraction. Use weighted sum to average the output probability, where the weights are determined by the validation accuracy of each model.

## Aim

This project aims to use an ensemble model to efficiently classify Alzheimer's disease MRI image into four categories, so as to take early measures to reduce mortality and reduce cost of medical resources.

## Objectives

1. Get to know what ensemble learning is.
2. Researches on more articles related AD
3. Analyzing the performance of each single model
4. Analyzing the performance of the ensemble model.
5. To preprocess the MRI dataset and prepare the data for training.
6. To train three CNN models (ResNet, AlexNet, and MobileNet) on the preprocessed data.
7. Model tweaking and fitting.
8. To develop an ensemble model by combining the predictions of the individual models.
9. Evaluating the performance of the model by using different performance evaluation metrics such as accuracy, recall, precision, F1-score.
10. To evaluate the performance of the individual models and compare with the ensemble model.
11. To develop a graphical user interface that can be used to identify Alzheimer's disease from MRI images.
12. Final presentation of this project to the targeted audience.

## Project Overview

### Scope

Magnetic resonance imaging is an important biological tissue imaging technique. Compared with other imaging techniques, MRI imaging has many advantages in terms of non-invasiveness and high spatial resolution, and is now widely used in medical imaging, especially for exploring complex and fine brain structures and functions. As a result, an increasing number of researchers are using deep learning methods to analyse magnetic resonance imaging (MRI) for the early diagnosis of brain diseases when no symptoms are apparent in the early stages. It is well known that Convolutional Neural Networks (CNN)'s fast feature extraction capability makes its pattern recognition in image data analysis very effective [3]. But CNN model application could face one or more challenges which could be overfitting, be underfitting, the gradient disappears, etc. Thus, the purpose of this project is to take the merit of single channel CNN models and concatenate them for a better performance which is refer to as Ensemble learning for the classification of Alzheimer disease.

The significant of this study include:

* Enables early screening, identification and disease risk warning of patients at high risk of Alzheimer's disease in the elderly population.
* Helps researchers to conduct effective research into Alzheimer's disease, leading to a better understanding of the disease's pathology and progression.
* Contributes to the development of novel drugs and measures to slow the progression of the disease.
* Reduces mortality in people with Alzheimer's disease.
* Reduces the probability of human-induced misdiagnosis or missed diagnosis.
* Helps to reduce the workload of doctors and improve efficiency.
* May provide a method as well as a reference for further research into early diagnosis and prevention techniques for related brain diseases.

### Audience

* Alzheimer's patients and their families.
* Medical professionals, such as neurologists and radiologists.
* Hospital.
* Medical Imaging Researcher.
* Medical magnetic resonance imaging manufacturer.
* Researchers and scholars interested in using deep learning techniques for medical diagnosis and image recognition.

# Background Review

Convolutional neural network has been utilized for analyzing and predicting medical images for physician and radiologists during diagnostics decision like in the case of alzheimer’s disease using single CNN model and ensemble models.

## Single CNN Model:

(Maqsood et. al., 2019) study transfer learning assisted classification and detection of alzheimer's disease stages using 3d MRI scans. Deep learning techniques were applied throughout the investigation to measure brain activity images and MRI diagnoses to avoid Alzheimer's disease, ultimately achieving a classification accuracy of 92.85% in a single CNN model like AlexNet [19]. (Kazemi et. al., 2018) study a deep learning pipeline to classify different stages of alzheimer's disease from f-MRI data. Convolutional neuronal network architecture AlexNet was applied to fMRI datasets to classify different stages of the disease [20]. (Lee et. al., 2019) propose a novel framework for structural magnetic resonance image (sMRI) classification of Alzheimer’s disease (AD) with data combination, outlier removal, and entropy-based data selection using AlexNet [21]. An Alzheimer’s stage detection system is proposed based on deep features using a pre-trained AlexNet model, by transferring the initial layers from pre-trained AlexNet model and extract the deep features from the Convolutional Neural Network (CNN) (Nawaz et. al., 2020) [22]. Early diagnosis of the disease, by detection of the preliminary stage, called Mild Cognitive Impairment (MCI), remains a challenging issue. In this respect (Miled et. al., 2020) introduce, a powerful classification architecture that implements the pre-trained network AlexNet to automatically extract the most prominent features from Magnetic Resonance Imaging (MRI) images in order to detect the Alzheimer's disease at the MCI stage [23]. (Li et. al., 2022) propose two improved ResNet algorithms that introduced the Contextual Transformer (CoT) module, group convolution, and Channel Shuffle mechanism into the traditional ResNet residual blocks [24]. (Liu et al., 2021) used an Attention Based 3D ResNet to detect the process of Alzheimer's disease (AD). The research results indicate that this model can highly accurately detect the AD process, with a classification accuracy of 96.7%, which is much higher than other existing methods [25]. Fulton et al employed a ResNet50 model in diagnosing three classes and achieved 98.99% [26]. (Lu et al., 2019) used the MobileNet model in deep learning technology to classify the early diagnosis of Alzheimer's disease. Through research and training on a large amount of data, they ultimately achieved a high classification accuracy of 94.8. They verified on the training set that the MobileNet model can still achieve high classification accuracy even with low computational resource requirements, and also demonstrated that compared to other deep learning models, the MobileNet model has the advantages of faster training speed and smaller computational resource requirements [27]. Rehman et al proposed hybrid classical quantum network for automatic detection of AD and ResnNet34 for feature extraction, achieving the highest test accuracy of 97.2% [28].

## Multi-model :

(Fedorov et. al., 2019) investigate the use of variants of DIM in a setting of progression to Alzheimer's disease in comparison with supervised AlexNet and ResNet inspired convolutional neural networks [29]. (Al-Adhaileh, 2021) study diagnosis and classification of alzheimer's disease by using a convolution neural network algorithm. Two deep neural network techniques, AlexNet and Restnet50, were applied for the classification and recognition of AD [30]. (Acharya et. al., 2021) aim to classify MRI of Alzheimer disease patients into multiple class by using VGG16, ResNet -50 and AlexNet as transfer learning models along with convolution neural networks [31].Since the convolutional layer of the general convolutional neural network (CNN) cannot satisfactorily extract long-distance correlation in the feature space, a deep residual network (ResNet) model, based on spatial transformer networks (STN) and the non-local attention mechanism, is proposed for the early diagnosis of AD (Sun et. al., 2021) [32]. Alanazi et al compared the hybrid algorithm between machine learning and deep learning in deep learning, and found that the accuracy, sensitivity, specificity and AUC values of the AlexNet+SVM model were 94.8%, 93%, 97.75% and 99.7% respectively, which was better [33].

## Ensemble Models

(An et. al., 2020) present a deep ensemble learning framework that aims to harness deep learning algorithms to integrate multisource data and tap the 'wisdom of experts' [1]. (Nanni et. al., 2020) evaluate the potential of ensemble transfer-learning techniques, pretrained on generic images and then transferred to structural brain MRI, for the early diagnosis and prognosis of AD, with respect to a fusion of conventional-ML approaches based on Support Vector Machine directly applied to structural brain MRI [34]. (Zhang et. al., 2021) study diagnosis of alzheimer's disease with ensemble learning classifier and 3d convolutional neural network [35]. Based on MRI data, a method combining a 3D convolutional neural network and ensemble learning is proposed to improve the diagnosis accuracy. There are three main objectives: i) to present a fully automated deep-ensemble approach for dementia-level classification from brain images, ii) to compare different deep learning architectures to obtain the most suitable one for the task, and (iii) evaluate the robustness of the proposed strategy in a deep learning framework to detect Alzheimer's disease and recognise different levels of dementia (Loddo et. al., 2021) [36]. (Li et. al., 2022) study ensemble of convolutional neural networks and multilayer perceptron for the diagnosis of mild cognitive impairment and alzheimer's disease [37]. To capture the anatomical changes in the brain caused by AD/MCI, deep learning-based MRI image analysis methods have been proposed in recent years. An integrative mulitresolutional ensemble deep learning-based framework is proposed to achieve better predictive performance for the diagnosis of Alzheimer disease (Razzak et. al., 2022) [38]. A summary of the researches based on AD can be seen in Table 4.

|  |  |  |  |
| --- | --- | --- | --- |
| **Authors** | **CNN Type** | **Proposed Model/Technique** | **Performance Metrics** |
| Maqsood et. al.,[19] | Single model | Fine-tuned AlexNet | Accuracy = 98.25% |
| Kazemi et. al., [20] | Single model | Fine-tuned AlexNet | Accuracy = 97.64% |
| Lee et. al., [21] | Single model | Fine-tuned AlexNet | Accuracy = 98.74% |
| Nawaz et. al., [22] | Single model | Fine-tuned AlexNet | Accuracy = 99.21% |
| Miled et. al., [23] | Single model | AlexNet | Accuracy = 96.83% |
| Li et. al., [24] | Single model | Fine-tuned ResNet | Accuracy = 97.50% |
| Liu et. al., [25] | Single model | 3D ResNet | Accuracy = 96.7% |
| Fulton et. al., [26] | Single model | ResNet50 | Accuracy = 98.99% |
| Lu et al., [27] | Single model | MobileNet | Accuracy = 94.8% |
| Rehman et. al., [28] | Single model | Hybrid classical quantum network /ResnNet34 | Accuracy = 97.2% |
| Fedorov et. al., [29] | Multi-model | Deep InfoMax + SVM | Accuracy = 86% |
| Al-Adhaileh [30] | Multi-model | ResNet -50  AlexNet | Accuracy = 58.7%  Accuracy = 94.53% |
| Acharya et. al., [31] | Multi-model | VGG16  ResNet -50  Modified AlexNet | Accuracy = 85.07%  Accuracy = 75.25%  Accuracy = 95.70% |
| Sun et. al., [32] | Multi-model | ResNet + spatial transformer networks (STN) | Accuracy = 97.1% |
| Alanazi et. al., [33] | Multi-model | AlexNet+SVM hybrid models  ResNet-50+SVM hybrid models | Accuracy = 94.8%  Accuracy = 93.3% |
| An et. al., [1] | Ensemble model | Inception V + ResNet50 + VGG19 | Accuracy = 85.2% |
| Nanni et. al., [34] | Ensemble model | AlexNet + GoogleNet + ResNet + Inception-v3 + SVM | Accuracy = 93.1% |
| Zhang et. al., [35] | Ensemble model | Ensemble learning classifiers + 3D CNN | Accuracy = 95.2% |
| Loddo et. al., [36] | Ensemble model | Fine-tuned AlexNet + Fine-tuned ResNet-101 + Fine-tuned Inception-ResNet-v2 | Accuracy = 98.67% |
| Li et. al., [37] | Ensemble model | VGG16 + Inception v3 + ResNet50 + DenseNet121 + MLP | Accuracy = 98.61% |
| Razzak et. al., [38]. | Ensemble model | 2D-CNN + 3D-CNN + Inception-v3 + DenseNet169 + ResNet50 | Accuracy = 97.9% |

**Table 4: Summarize the other related work for Alzheimer disease classification.**

# Methodology

This section will explain the approach carried out which include, the dataset and its preprocessing, the model and the performance evaluation metrics.

## Dataset

This dataset is gotten from Kaggle dataset and it contains 33984 cross-sectional MRI images of the brain with Alzheimer's disease. 8960 MRI images contains mild dementia, 6464 MRI images is moderate dementia, 9600 MRI images is non-dementia, and 8960 MRI images is the very mild dementia. The split ratio for the training set, validation set, and test set is close to 6:2:2. The four types of images of Alzheimer's disease are shown in Table 5.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Image | 00be34e4-c61c-45a8-8ee8-48e29a8adce0 | 3c8e9142-69fc-4d7a-8c3b-852bdd9b20e7 | 00b8529f-23c6-415c-96cc-a4e6ca6ed6ac | 0bc79d8a-5243-4041-acc4-70b27b990519 |
| Type | MildDetermined | ModerateDetermined | NonDetermined | VeryMildDetermined |

**Table 5 :The four types of images of Alzheimer's disease**

### Dataset preprocessing

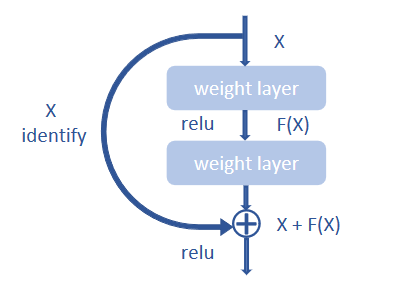
In this phase, data preprocessing was performed on the data so it could fit for the model. This project implements preprocessing of image datasets, including data reading, resizing, and normalization.

Start by defining the file path train\_ Picture and test\_ Picture points to the folder paths of the training set and the test set, respectively, with the parameter imges\_set Size=64 is used to uniformly scale the input pictures to the same size. At the same time, four categories of images were set: non-dementia, mild dementia, moderate dementia and very mild dementia. Next, due to the different pixels of each image, it is necessary to use the CV library to reset all image sizes to a uniform size, using the cv2.imread() function to read the pictures in each category folder, and use the cv2.resize() function to resize the pictures to the specified size imges\_ Size. Then to make training more stable, using image = image/255.0 and img\_ To\_ The array() function divides the pixel value by 255.0 so that standardizes the data to a range of [0,1], and converts it to a numpy array format. Finally, the np.random.shuffle() function is used to randomly rearrange the training set to ensure better generalization performance of the model in the learning process..

## Ensemble Model

### Finetuned Residual Network (ResNet)

In the figure 8, x is weighted by the first layer, and then F (x)+x is obtained after the nonlinear variation of the Relu function and the second layer weighting. This is a linear stack with two layers constituting a residual learning module. The network composed of residual modules is called ResNet. The difference between ResNet and ordinary networks is the introduction of jump connections, which can help the information from the previous residual block flow unimpeded into the next residual block. The problem of gradient disappearance and degradation caused by too deep a network is avoided.



**Figure 8: Residual block Architecture**

The residual network(ResNet) is constructed from Residual Building Blocks, it does not increase the complexity of the network while increasing the depth of the network, and the effect is far better than other networks such as VGG and Google Net. With the increase of the number of floors, this advantage becomes more and more obvious. The quick connection of ResNet makes the network easier to optimize. The internal residual block uses a skip connection, which alleviates the problem of gradient disappearance caused by increasing depth in the deep neural network.

The first step in fine-tuning ResNet is to load a dataset from a specified path, which contains brain scans of patients with varying degrees of dementia. Then adjust the size of the dataset to a fixed size, in this case 64 x 64 pixels, and normalize the pixel values by dividing each pixel value by 255. This normalization process helps to improve the performance of the model by ensuring that the input values are within a certain range. The classification labels for images have been redefined, including four categories: 'Non Determined', 'MildDetermined', 'ModerateDetermined', and 'VeryMildDetermined'.

Next, the dataset is divided into a training set and a validation set. The training set is used to transfer patterns from the data to the model, and the validation set is used to evaluate the performance of the model on new, invisible data. Load the training dataset, use the imread method in the CV2 library to read the picture, and zoom it to the specified size. The shuffle method was also used to shuffle the datasets in order to improve the model's effectiveness. Finally, the tags were processed with one-hot encoding (to\_categorical).

Define ResNet Block and ResNet models. First, define the ResNet Block, the Residual Block, which is the most basic unit in ResNet. Residual blocks allow the model to learn the residual characteristics of the input and then add them back to the original input. This helps the model learn the underlying patterns in the data while preserving the original information. The residual block consists of two three-layer convolution operations, each with a core size of 3 x 3, which introduces residual information by adding jumps across a small number of layers. A shortcut connection is introduced to solve the gradient diffusion problem in deep neural networks. That is, the if structure is used to determine whether a downsampling operation is required, and if so, a convolution of step 2 is added to the main path for downsampling and a 1x1 convolution on the shortcut to accommodate the change in channel count; otherwise, the two paths are joined directly. Finally, the two paths are added together using the add function and the output is processed by the activation function.

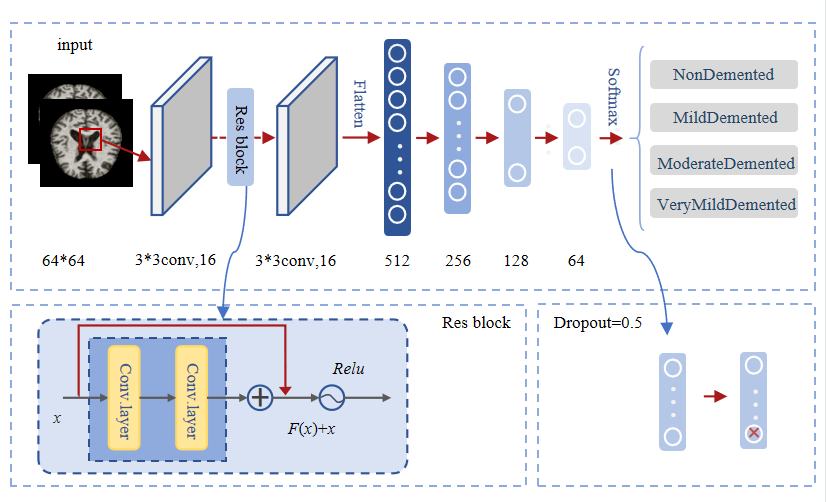
Next the ResNet model is defined, which first extracts the initial features by a conv2d operation that convolves the layer with 16 filters and a convolution kernel of size 3 x 3. Then using the residual block method, the first block is constructed with an output channel count of 16 and maxpooling2d is applied for feature extraction and compression (pooling=True). After the block, the conv2d and relu activation functions are applied once more. The 2D feature map is then compressed by flatten into a 1D tensor for later processing. The final package of fully connected layers is responsible for classifying the input into one of four categories - non-dementia, mild dementia, moderate dementia and mild dementia. The dense layer has 512, 256 and 128, 64 nodes respectively, followed by the discard layer, which is used to prevent overfitting by randomly discarding nodes during training.

Then, define the model input-output and optimizer, in which the picture size is (64,64) and is a RGB color picture, so the input shape is (64,64,3). Compile the ResNet model using the Adam optimizer and the Classification Cross-Entropy Loss Function. The model uses a training set of 128 batches for 20 cycles. The validation segment of 0.2 is used to validate the performance of the model after each epoch. The performance of the model is evaluated using accuracy indicators, which measure the percentage of correct predictions made by the model.After training, the model is evaluated on the test set to measure its performance on invisible data. Test losses and accuracy will be printed to the console. Visualize the performance of the model using two graphs, one for training and verifying accuracy, and the other for training and verifying losses.

After training, the model is evaluated on the test set to measure its performance on invisible data. Visualize the performance of the model using two diagrams, one for training and validating accuracy and the other for training and validating losses.

Finally, the model is saved to a file for future use. This process fine-tunes the ResNet architecture for specific tasks, including loading data, defining the ResNet architecture, compiling models, training models, evaluation models, and saving models.

The Finetuned ResNet Architecture used in this project is shown in Figure 9



**Figure 9: Finetuned ResNet Architecture**

### Finetuned AlexNet

AlexNet has an eight layer structure. The first five layers are convolutional neural networks, and the sixth to eighth layers are traditional neural networks. It uses the ReLU activation function to prevent the gradient from disappearing and the Dropout to prevent over fitting. The whole network can be seen as the input layer is operated by convolutional layers, followed by a series of fully connected layers, and finally the output layer is used to obtain the prediction result.

The fine-tuned alexnet model structure, including the role of each layer, parameter settings, and so on, is described in detail below.

First, load the necessary libraries and set environment variables. At the beginning of the code, use the Import command to load the required libraries. Where os.environ ['CUDA\_VISIBLE\_DEVICES'] = "0" specifies the number of the GPU, indicating that the GPU with the number 0 is used for training and reasoning. At the same time, tensorflow, numpy, opencv-python libraries are imported for subsequent data processing and model building.

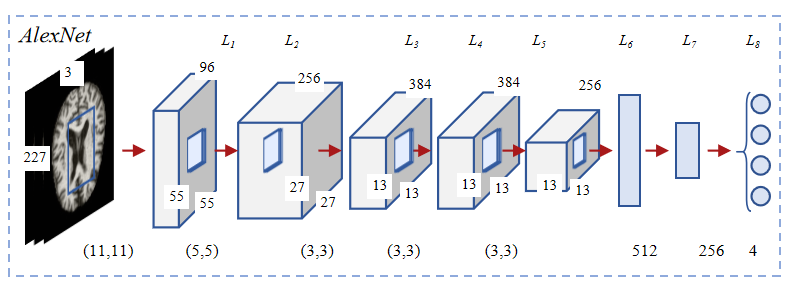
The second step is to load the dataset. The dementia symptom dataset used in the code consists of four categories, NonDemented, MildDemented, ModerateDemented, and VeryMildDemented. The images in the these category are all grayscale images with varying sizes. In the code, all images are resized to the same size of 64\*64 by iterating through each category of images, which is the input size required by the AlexNet model, and stored in train\_ Img\_ Final list. At the same time, each image's corresponding label (i.e., category) is stored in train\_ Img\_ Label list. After the traversal is complete, convert the two lists into numpy arrays. Then use the shuffle function to scramble the data to avoid fitting. Finally, labels are coded one-hot, which converts each label into a vector of the number of categories (4 here), where the corresponding label position for the image is set to 1 and the others to 0.

The third step is model design. First, the input shape of the model is the size of the picture 64x64x3, and 3 represents the number of channels for the color image, since the image is converted to RGB format using OpenCV before training. After the input layer, there are five convolution layers, each with a different filter size and number of filters, each with a step of 1 and a "same" padding with an activation function of relu. The first layer uses four convolution cores, each 11 in size × 11. Four feature maps are output after convolution. The second layer uses eight convolution cores, each size being 5 × 5. Eight feature maps are output after convolution. Layers 3 and 5 use eight convolution cores of 3 size each × 3. Eight feature maps are output after convolution. Layer 4 uses 16 convolution cores of 3 size each × 3. After convolution, 16 feature maps are output. The sixth layer, Flatten, expands the output of the previous layer into a one-dimensional vector with 19200 features. Next, two full-junction layers were performed, and ReLU activation was used. The first full-junction layer contained 512 neurons and the second 256 neurons. A shedding layer with a shedding rate of 0.5 is added after each compact layer to reduce over-fitting. Finally, a fully connected layer of the softmax activation function is added to output the probability of four classifications.

The fourth step is model compilation and training. At model compilation time, the loss function is configured as cross-entropy function, the optimizer is Adam optimizer, and the evaluation index is accuracy. Set the batch size to 64, train 20 cycles, and evaluate the model using a 0.2 cross-validation scale. During the training process, using history to record the loss and accuracy value of each iteration, it show a clearer training process.

Finally, the model is evaluated and saved. By recording historical loss and accuracy values during training, the training and validation accuracy and loss comparison diagrams of the model are drawn, and the model is saved in the current catalog for future reuse.

The Finetuned AlexNet Architecture used in this project is shown in Figure 10.



**Figure 10: Finetuned AlexNet Architecture**

### Finetuned MobileNet

The MobileNet structure uses depthwith separable convolution to replace the standard convolution operation, and calls these two structures repeatedly to reduce the amount of model parameters and increase the amount of model calculation. Each layer is followed by a batchnorm and a ReLU nonlinear layer. Finally, the Flatten layer and the full connection layer are used to classify the images. Point convolution and deep convolution structure are the core of MobileNet, which makes MobileNet more efficient and more suitable for mobile devices. Point convolution is mainly responsible for integrating the information in the feature map, while depth convolution is responsible for extracting features.

The focus of this architecture is to provide a lightweight and efficient model that can run well on mobile and embedded devices with limited computing resources. In the original architecture, proposed using deep separable convolutions to decompose standard convolutions into deep convolutions and point by point convolutions. This significantly reduces the number of parameters and computational costs of the model.

Firstly, import the necessary modules and set environment variables to train using GPU. Then, the code defines some parameters, such as image size, random seed for data shuffling, and the class that the model needs to classify.

Then use OpenCV to load training and testing data from their respective directories. The size of the image is adjusted to a specified image size of 64x64 pixels, and normalized by dividing each pixel value by 255.0. Then convert the image into an array and attach it to the training and testing image array. In addition, corresponding labels are created based on the category of the image.

Before training the model, use random seed to mix and wash the training and test data to ensure that the data is randomly divided during training and testing. The label also uses Keras' to '\_ The categorical() function is converted to categorical data.

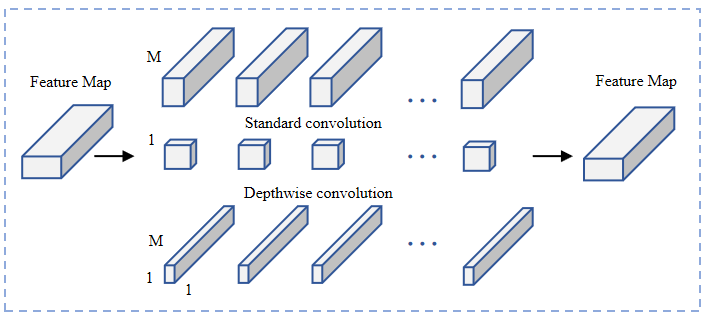
Then, several functions were defined to build the MobileNet architecture. depth\_ point\_ Conv2d () and pointwise\_ The conf() function is used to implement deep separable convolution, which is the main building block of MobileNet.

The mobilenet() function is used to define the mobilenet architecture. This function adopts an input tensor and applies a series of convolutions and depth separable convolution layers. This function also applies BatchNormalization and ReLU activation function after each convolution layer to stabilize training and introduce nonlinearity. Then, the application depth of this function can separate several iterations of the convolution and end with an average pooling layer, followed by several fully connected layers with ReLU activation and regularization discarding.

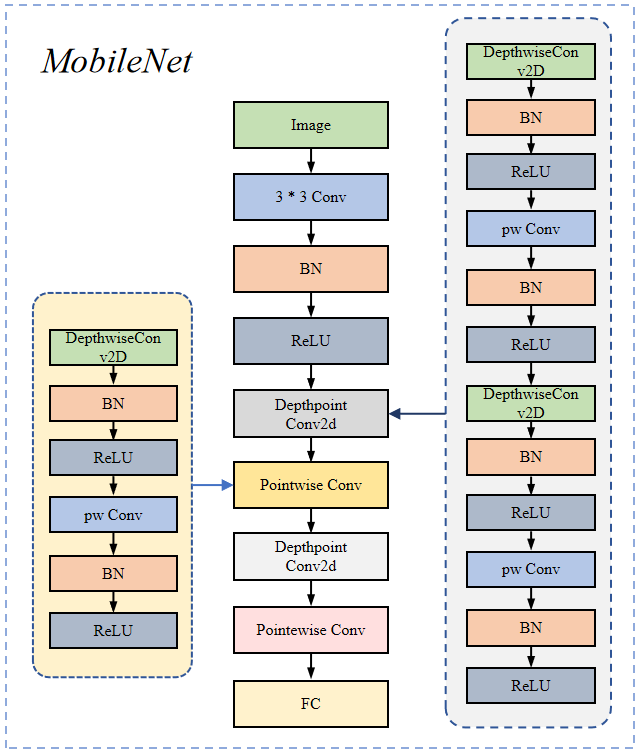
Once the model architecture is defined, the code uses random gradient descent (SGD) as the optimizer, the learning rate is 0.002, and the classification cross entropy is used as the loss function to compile the model. Then use the fit () function to train the model, which takes training data and labels as inputs and trains the model for a specified number of periods and batch sizes.

After training, the code draws the training and validation accuracy and losses for each epoch, and saves the drawing to disk. The model is also saved to disk using Keras' save() function.

This process uses deep separable convolution to reduce the number of model parameters and computational costs. The SGD with low learning rate and classification cross entropy loss is used to train the model, and the training data is mixed before each epoch. Finally, evaluate the model on a separate test dataset and print the test loss and accuracy to the console. The Depthwise Separable Convolution Architecture used in this project is shown in Figure 11. The Finetuned MobileNet Architecture used in this project is shown in Figure 12. The MobileNet network structure in this article is shown in table 6.



**Figure 11: Depthwise Separable Convolution Architecture**



**Figure 12: Finetuned MobileNet Architecture**

|  |  |  |  |
| --- | --- | --- | --- |
| Input layer | Convolution kernel | strides | Output layer |
| Input | 3×3×3×32 | 2 | Conv2d\_0 |
| Conv2d\_0 | 32×3×3dw | 1 | Conv2d\_1\_depthwise |
| Conv2d\_1\_depthwise | 32×1×1×64 | 1 | Conv2d\_1\_pointwise |
| Conv2d\_1\_pointwise | 64×3×3dw | 2 | Conv2d\_2\_depthwise |
| Conv2d\_2\_depthwise | 64×1×1×128 | 1 | Conv2d\_2\_pointwise |
| Conv2d\_2\_pointwise | 128×3×3dw | 1 | Conv2d\_3\_depthwise |
| Conv2d\_3\_depthwise | 128×1×1×18 | 1 | Conv2d\_3\_pointwise |
| Conv2d\_3\_pointwise | 128×3×3dw | 1 | Conv2d\_4\_depthwise |
| Conv2d\_4\_depthwise | 128×1×1×256 | 1 | Conv2d\_4\_pointwise |
| Conv2d\_4\_pointwise | 256×3×3dw | 2 | Conv2d\_5\_depthwise |
| Conv2d\_5\_depthwise | 256×1×1×256 | 1 | Conv2d\_5\_pointwise |
| Conv2d\_5\_pointwise | 256×3×3dw | 1 | Conv2d\_6\_depthwise |
| Conv2d\_6\_depthwise | 256×1×1×512 | 1 | Conv2d\_6\_pointwise |
| Conv2d\_6\_pointwise | 512×1×1×512 | 1 | Flatten |
| Flatten | --(512) | 1 | Dense1 |
| Dense1 | --(512) | 1 | Dense2 |
| Dense2 | --(256) | 1 | Dense3 |
| Dense3 | Softmax | 1 | Output |

**Table 6:The MobileNet network structure**

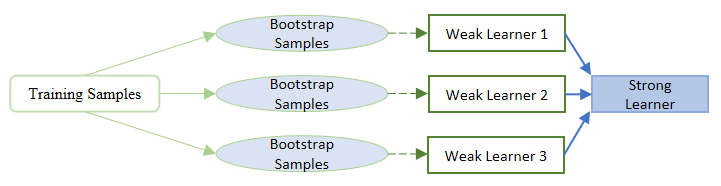
### Ensemble Model

For a given training sample S, M training samples were drawn from the training sample S in each round using Booststraping, and n rounds were conducted to obtain a total of n sets of samples. After obtaining the sample sets, one prediction model is obtained each time using one sample set, for n sample sets, a total of n prediction models can be obtained. The classification results are then obtained by applying voting to the previously obtained n models.

* Algorithm

Bagging (Bootstrap Aggregation): To obtain an integration with strong generalisation performance, the ensemble learning in the integration should be as independent of each other as possible, or try to make the base learners as undifferentiated as possible. Given a training dataset, one possible approach is to sample the training samples to produce a thousand different subsets, and then train a base learner from each subset of the data. This is expected to result in a large variation in the base learners due to the different training data, however, in order to obtain better integration, it is also desirable that individual learning is not too poor, and if each subset sampled is completely different, then each learner uses only a small fraction of the training data, not even enough to learn effectively, neither of which is clearly guaranteed to produce a better base learner. Therefore, to solve this problem, consider using sampled subsets that are overlapping with each other.

The ensemble Architecture is shown in figure 13.



**Figure 13: The Ensemble Architecture**

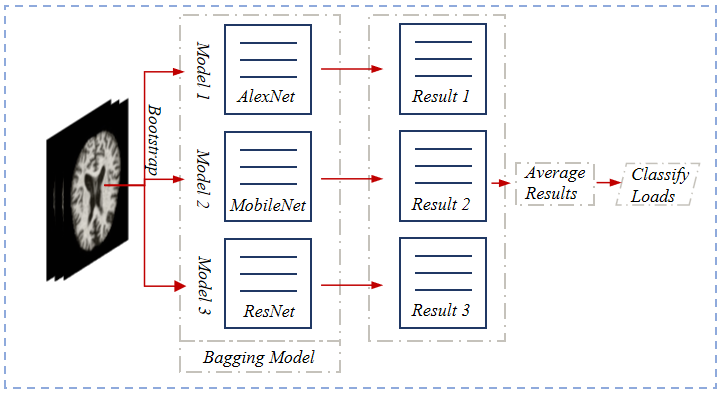
This project sets the file path as an input parameter with the goal of classifying image files into one of four categories using three different pre-training models (modelr, modela, and modelm). The output of each model is combined using a voting scheme, which selects the category with the most votes and returns a string representing the predicted classification.

This list will be used to accumulate votes for each possible category. After reading the image file on file\_path and resizing it to 64 x 64 pixels, the resulting image is converted to a three-dimensional array with an added axis (ndarray). Corresponds to the sample size.

Then, three pre-trained machine learning models, modelr, modela, and modelm (ResNet, AlexNet, MobileNet), are used to predict the probability that a given image will belong to each of the four predefined classes. The three models have been fine-tuned and trained. Each model produces probability distributions of four categories, which sum to 1. The argmax() function is applied to each potential class probability distribution to retrieve the index with the maximum value corresponding to the most likely prediction for that particular model, then overwrites the existing variables pred1, pred2, and pred3 with the integer index of the prediction class label. The predicted class indexes and their respective class labels are then printed sequentially through a print statement. Suppose Classes is a dictionary that maps the class index to the class name.

Next, the code updates the list of votes. Each predicted class index receives one voting unit, so its corresponding value in the list is increased by one. The variable pred is initially set to the predicted value of the second model, pred2. However, if more than one class has the same maximum number of votes, the variable pred gets the index of the class with the most votes in the voting list.

Finally, the function returns a string that indicates the classification (symbol) of the prediction using the class name corresponding to the prediction class index (pred) found in the previous step. This value is displayed in the final graphical user interface element, completing the provision of predictive classifications based on classifiers that integrate multiple pre-trained models. The project architecture is shown in figure 14.



**Figure 14: Architecture of the project**

### Performance Evaluation Metrics

This phase will discuss the evaluation metrics used to checkmate the performance of the model which include accuracy and loss, precision, recall, f1-score.

Common evaluation indicators include accuracy, precision, recall\_ score, f1-score, roc\_ auc\_ Score et al. With precision\_ score, recall\_ score and others as an example, there is an important parameter 'average'. The average parameter defines the calculation method for this indicator. In binary classification, the average parameter defaults to binary. In multi classification, optional parameters include micro, macro, weighted, and samples. The parameter 'weighted' is used in this project. The project confusion matrixe is shown in table 7.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| The four classifications | | Predicted condition | | | |
| MildDetermined | ModerateDetermined | NonDetermined | VeryMildDetermined |
| True condition | MildDetermined | T1P1 | F1P2 | F1P3 | F1P4 |
| ModerateDetermined | F2P1 | T2P2 | F2P3 | F2P4 |
| NonDetermined | F3P1 | F3P2 | T3P3 | F3P4 |
| VeryMildDetermined | F4P1 | F4P2 | F4P3 | T4P4 |

**Table 7 Confusion Matrixe**

R1=

R2=

R3=

R4=

Accuracy1=

Accuracy2=

Accuracy3=

Accuracy4=

Recall1=

Recall2=

Recall3=

Recall4=

Precision1=

Precision2=

Precision3=

Precision4=

**Accuracy =**

**Recall =**

**Precision =**

**F1 score =**

### Hyperparameters

|  |  |
| --- | --- |
| parameters | value |
| **AlexNet** | |
| Image\_size | 64 |
| Random\_seed | 10 |
| Drop out | 0.5 |
| learning rate | 0.0001 |
| Batch size | 64 |
| Epochs | 20 |
| Optimizer | Adam |
| verbose | 2 |
| Validation split | 0.2 |
| Loss function | Categorical\_cross entropy |
| Activation function for hidden layers | ReLU |
| Number of filters in the Conv2D layer | 4; 8; 8; 16; 8 |
| input\_shape | (64,64,3) |
| Convolution kernel step size | (1,1) |
| Convolution Kernel size | First layer: 11x11 |
| Second layer : 5x5 |
| Third to fifth layer: 3x3 |
| Number of neurons in the Dense layer | 512; 256 |
| Activation function for the output layer | softmax |
| **MobileNet** | |
| Image\_size | 64 |
| Random\_seed | 10 |
| Drop out | 0.3; 0.5 |
| learning rate | 0.002 |
| Batch size | 32 |
| Epochs | 30 |
| Optimizer | SGD |
| verbose | 2 |
| Validation split | 0.2 |
| Loss function | Categorical\_cross entropy |
| Activation function for hidden layers | ReLU |
| Number of filters in the Conv2D layer | 32 |
| input\_shape | (64,64,3) |
| Convolution kernel size in DepthwiseConv2D | (3,3) |
| Number of convolutional cores used in the Conv2D layer | [64,128] |
| kernel size | (3, 3) |
| Step size used in DepthwiseConv2D | First layer: 1 |
| Second layer: 2 |
| pointwise\_conv :1 |
| Number of neurons in the Dense layer | 512; 256; 4 |
| Activation function for the output layer | softmax |
| **ResNet** | |
| Image\_size | 64 |
| Random\_seed | 10 |
| Drop out | 0.5 |
| learning rate | 0.0001 |
| Batch size | 128 |
| Epochs | 20 |
| Optimizer | Adam |
| verbose | 2 |
| Validation split | 0.2 |
| Loss function | Categorical\_cross entropy |
| Activation function for hidden layers | ReLU |
| Number of filters in the Conv2D layer | 3; 3 |
| input\_shape | (64,64,3) |
| Size of the pooling window in the pooling layer | (2,2) |
| Dropout rate | 0.25; 0.5 |
| Number of convolution kernels in the residual\_block function | 16 |
| Number of neurons in the Dense layer | 512; 256; 128; 64 |
| Activation function for the output layer | softmax |

**Table 8 : Hyperparameters**

### Technology

|  |  |
| --- | --- |
| Framework | Tensorflow |
| IDE | Vscode |
| Language | Python 3.8 |
| Central processing Unit (CPU) | Intel(R) Core (TM) i7-9750H CPU @ 2.60GHz 2.59 GHz |
| Graphic Processing Unit (GPU) | NVIDIA GeForce GTX 1650 |

**Table 9 : Technology**

### Project Version Management

Since the project will keep updating status before completion, in order to ensure the complete process of the project, a series of project version management is created, with Baidu Netdisk as the main management platform, in the whole project development process to maintain the same real-time automatic update status as the local, to avoid the loss of file data in the project. The contents are as follows:

* Schedule

Project activity sheet, Gantt diagram.

* Weekly progress

Weekly reports for the two semesters, including weekly progress records, difficulties encountered, preparations for the next week and teachers' comments.

Weekly records of completed stages, such as weekly learning progress, study notes, drawings etc.

* Literature

The literature used in the project and the learning materials in the initial phase, such as literature or code, etc.

* Code

Datasets and their source web pages.

Project code, such as model code, ensemble code, drawing code, and Graphics User Interface code.

Code test results, such as evaluation data and drawings of evaluation data, as well as raw code and adjustment code.

* Report

Project Proposal Report and its drafts.

Project Process Report and its drafts.

Project Final Report , PPT and poster and their drafts

# Results

* 1. Hyperparameters setting

In machine learning, the learning rate is the tuning parameter in an optimisation algorithm that determines the step size in each iteration that determines whether and when the objective function converges to a local minimum. A suitable learning rate allows the objective function to converge to a local minimum in a suitable amount of time. An epoch refers to the process of feeding all the data into the network to complete a forward and backward transfer, which is simply equivalent to training once with all the samples in the training set. dropout is a strategy proposed to prevent overfitting during the training process of a neural network, aiming to randomly delete neurons in the network according to a certain probability during the training process, which can reduce the complex co-adaptation relationship between neurons. Batch is a portion of the data that is fed into the network for training, and Batch Size is the number of training samples in each batch, the size of which affects the training speed and model optimization.

* **AlexNet Hyperparameters setting:**

For alexnet, the four hyperparameters Drop out and Learning rate and Batch size and Epochs are mainly tuned for. AlexNet Hyperparameters setting is shown in table 10.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Test | Drop out  (81 row) | Learning rate | batch\_size | Epochs |
| Learning rate test | Test1 | 0.5 | **0.01** | 64 | 20 |
| Test2 | 0.5 | **0.003** | 64 | 20 |
| Test3 | 0.5 | **0.001** | 64 | 20 |
| Test4 | 0.5 | **0.0001** | 64 | 20 |
| Test5 | 0.5 | **0.00001** | 64 | 20 |
| Dropout test | Test4 | **0.5** | 0.0001 | 64 | 20 |
| Test6 | **0.4** | 0.0001 | 64 | 20 |
| Test7 | **0.3** | 0.0001 | 64 | 20 |
| Test8 | **0.2** | 0.0001 | 64 | 20 |
| batch\_size test | Test4 | 0.5 | 0.0001 | **64** | 20 |
| Test9 | 0.5 | 0.0001 | **32** | 20 |
| Epochs test | Test10 | 0.5 | 0.0001 | 64 | **15** |
| Test4 | 0.5 | 0.0001 | 64 | **20** |
| Test11 | 0.5 | 0.0001 | 64 | **25** |

**Table 10: AlexNet Hyperparameters setting**

* **ResNet Hyperparameters setting:**

With regard to the adjustment of ResNet, the adjustment of a Number of neurons in the Dense layer is added on the basis of the four super parameters of Drop out and Learning rate and Batch size and Epochs. ResNet Hyperparameters setting is shown in table 11.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Test | Number of neurons in the Dense layer | Drop out  (123 row) | Learning rate | batch\_size | Epochs |
| Number of neurons in the Dense layer test | Test12 | **512; 256; 128** | 0.5 | 0.0001 | 128 | 20 |
| Test13 | **512; 256; 128; 64** | 0.5 | 0.0001 | 128 | 20 |
| Drop out  (123 row) test | Test13 | 512; 256; 128; 64 | **0.5** | 0.0001 | 128 | 20 |
| Test14 | 512; 256; 128; 64 | **0.4** | 0.0001 | 128 | 20 |
| Test15 | 512; 256; 128; 64 | **0.3** | 0.0001 | 128 | 20 |
| Test16 | 512; 256; 128; 64 | **0.2** | 0.0001 | 128 | 20 |
| Learning rate test | Test17 | 512; 256; 128; 64 | 0.5 | **0.001** | 128 | 20 |
| Test13 | 512; 256; 128; 64 | 0.5 | **0.0001** | 128 | 20 |
| Test18 | 512; 256; 128; 64 | 0.5 | **0.00001** | 128 | 20 |
| batch\_size test | Test13 | 512; 256; 128; 64 | 0.5 | 0.0001 | **128** | 20 |
| Test19 | 512; 256; 128; 64 | 0.5 | 0.0001 | **64** | 20 |
| Epochs test | Test20 | 512; 256; 128; 64 | 0.5 | 0.0001 | 128 | **15** |
| Test13 | 512; 256; 128; 64 | 0.5 | 0.0001 | 128 | **20** |
| Test21 | 512; 256; 128; 64 | 0.5 | 0.0001 | 128 | **25** |

**Table11: ResNet Hyperparameters setting**

* **MobileNet Hyperparameters setting:**

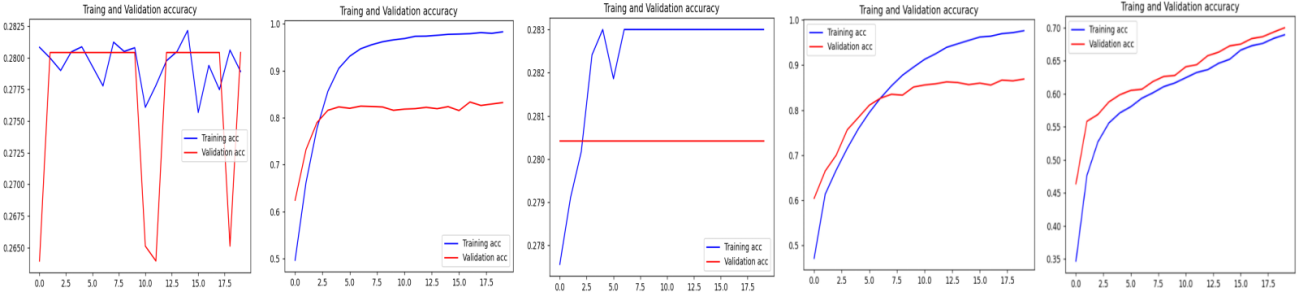
As for mobilenet, it mainly adjusts four super parameters, namely Drop out and Learning rate, Batch size and Epochs. MobileNet Hyperparameters setting is shown in table 12.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Test | Drop out  (119 row) | Learning rate | batch\_size | Epochs |
| Drop out  (119 row) test | Test22 | **0.5** | 0.002 | 32 | 30 |
| Test23 | **0.4** | 0.002 | 32 | 30 |
| Test24 | **0.3** | 0.002 | 32 | 30 |
| Test25 | **0.2** | 0.002 | 32 | 30 |
| Learning rate test | Test26 | 0.3 | **0.003** | 32 | 30 |
| Test24 | 0.3 | **0.002** | 32 | 30 |
| Test27 | 0.3 | **0.001** | 32 | 30 |
| Test28 | 0.3 | **0.0001** | 32 | 30 |
| batch\_size test | Test24 | 0.3 | 0.002 | **32** | 30 |
| Test29 | 0.3 | 0.002 | **64** | 30 |
| Epochs test | Test30 | 0.3 | 0.002 | 32 | **25** |
| Test24 | 0.3 | 0.002 | 32 | **30** |

**Table 12: MobileNet Hyperparameters setting**

* 1. Evaluations of the project model
     1. Evaluation of the AlexNet model

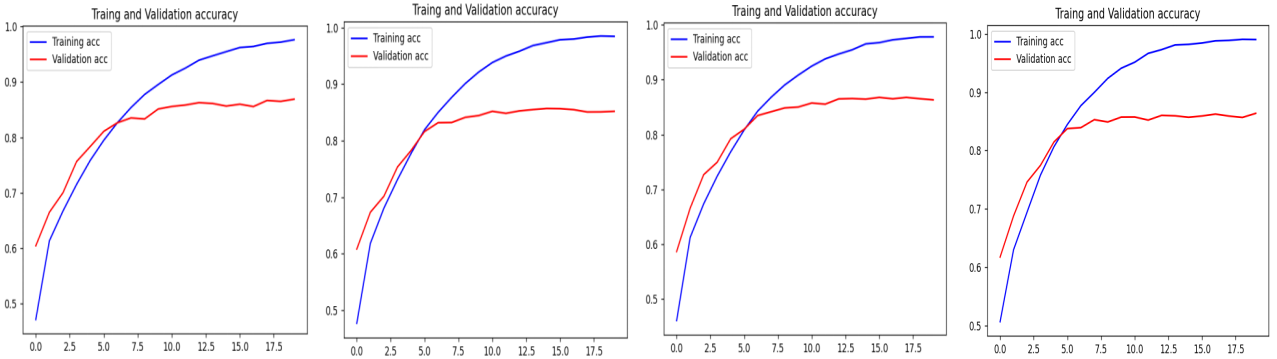
The optimal results were obtained with learning rate = 0.0001, dropout = 0.5 and batch\_size = 64, epochs = 20, after a number of tuning sessions with control variables. Figure 14-1 to figure 14-4 show the change of accuracy of AlexNet parameter adjustment . Table 13-1 to table 13-4 show the data of the evaluation metrics when AlexNet adjusts parameters.



**Figure 14-1**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Learning rate test | Test1  Learning rate = 0.01 | Test2  Learning rate = 0.003 | Test3  Learning rate = 0.001 | Test4  Learning rate = 0.0001 | Test5  Learning rate = 0.00001 |
| Accuracy | 0.5 | 0.5 | 0.98953127 | **0.99546875** | 0.71062499 |
| Loss | 1.32134962 | 1.30115568 | 0.03708589 | **0.01434312** | 0.65347212 |
| Precision | 0.25 | 0.25 | 0.98970680 | **0.99548415** | 0.73481878 |
| Recall | 0.5 | 0.5 | 0.98953125 | **0.99546875** | 0.710625 |
| F1-score: | 0.33333333 | 0.33333333 | 0.98954171 | **0.99547028** | 0.71120726 |

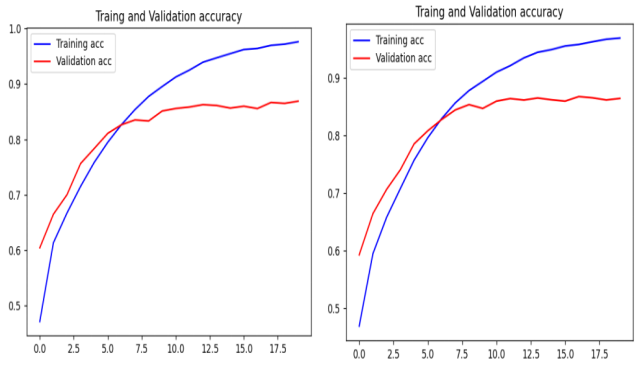
**Table 13-1**

****

**Figure 14-2**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Dropout  (81 row)  test | Test4  Dropout = 0.5 | Test6  Dropout = 0.4 | Test7  Dropout = 0.3 | Test8  Dropout = 0.2 |
| Accuracy | **0.99546875** | 0.99062502 | 0.99515622 | 0.99234372 |
| Loss | **0.01434312** | 0.03234234 | 0.01592272 | 0.01946141 |
| Precision | **0.99548415** | 0.99068651 | 0.99517624 | 0.99237596 |
| Recall | **0.99546875** | 0.99062500 | 0.99515625 | 0.99234375 |
| F1-score: | **0.99547028** | 0.99061322 | 0.99515871 | 0.99234449 |

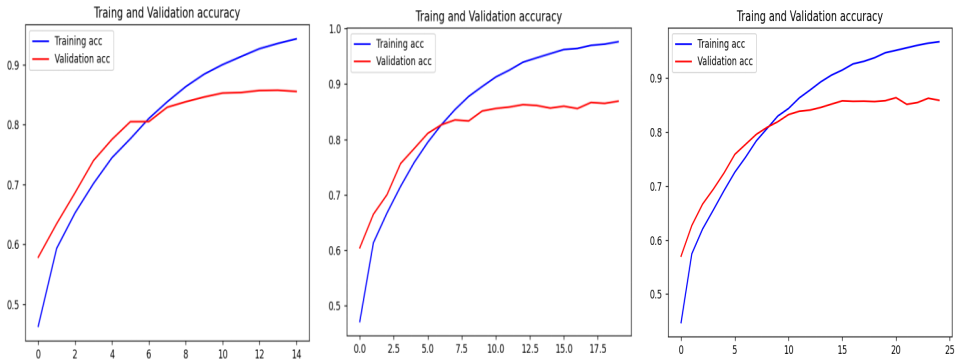
**Table 13-2**

****

**Figure 14-3**

|  |  |  |
| --- | --- | --- |
| batch\_size test | Test4  batch\_size = 64 | Test9  batch\_size = 32 |
| Accuracy | **0.99546875** | 0.99328124 |
| Loss | **0.01434312** | 0.02357180 |
| Precision | **0.99548415** | 0.99330302 |
| Recall | **0.99546875** | 0.99328125 |
| F1-score: | **0.99547028** | 0.993281293 |

**Table 13-3**

****

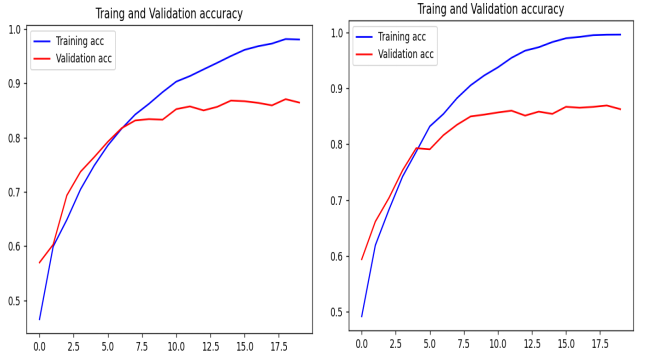
**Figure 14-4**

|  |  |  |  |
| --- | --- | --- | --- |
| Epochs test | Test10  Epochs=15 | Test4  Epochs=20 | Test11  Epochs=25 |
| Accuracy | 0.99390625 | **0.99546875** | 0.99328124 |
| Loss | 0.02392680 | **0.01434312** | 0.02161609 |
| Precision | 0.99390612 | **0.99548415** | 0.99328067 |
| Recall | 0.99390625 | **0.99546875** | 0.99328125 |
| F1-score: | 0.99390517 | **0.99547028** | 0.99327854 |

**Table 13-4**

* + 1. Evaluation of the ResNet model

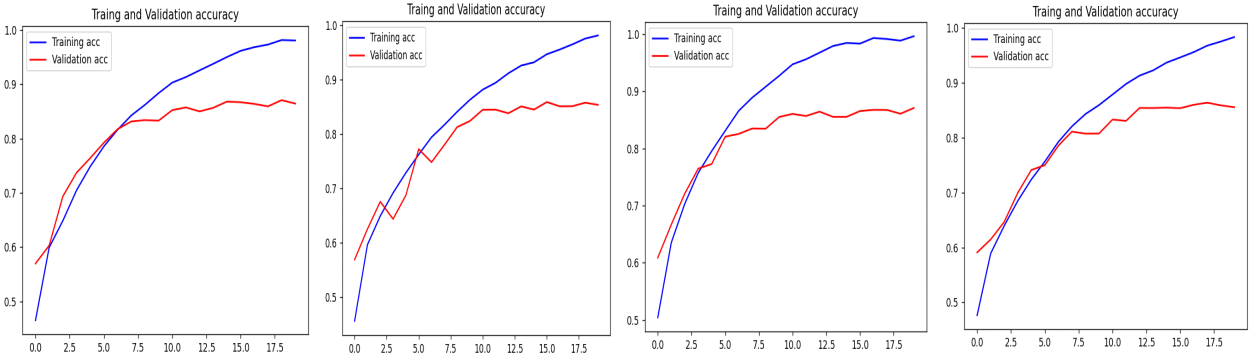
The optimal results were obtained with learning rate = 0.0001, dropout = 0.5, batch\_size = 128, epochs = 20 and number of neurons in the Dense layer = 512; 256; 128;64, after a number of tuning sessions with control variables. Figure 15-1 to figure 15-5 show the change of accuracy of ResNet parameter adjustment. Table 14-1 to table 14-5 show the data of the evaluation metrics when AlexNet adjusts parameters.



**Figure 15-1**

|  |  |  |
| --- | --- | --- |
| Number of neurons in the Dense layer | Test12  Number of neurons in the Dense layer = 512; 256; 128 | Test13  Number of neurons in the Dense layer = 512; 256; 128;64 |
| Accuracy | 0.98328125 | **0.99281250** |
| Loss | 0.04883653 | **0.02714268** |
| Precision | 0.98345836 | **0.99282491** |
| Recall | 0.98328125 | **0.99281250** |
| F1-score: | 0.98325560 | **0.99280679** |

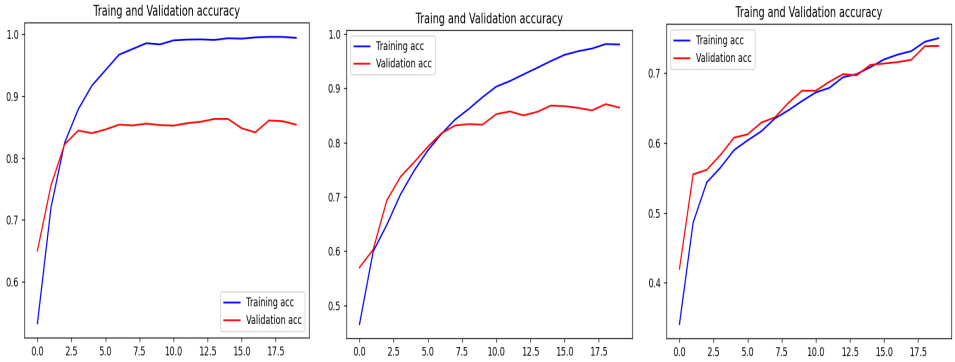
**Table 14-1**



**Figure 15-2**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Drop out  (123 row) | Test13  Dropout = 0.5 | Test14  Dropout = 0.4 | Test15  Dropout = 0.3 | Test16  Dropout = 0.2 |
| Accuracy | **0.99281250** | 0.98671877 | 0.99174997 | 0.98593747 |
| Loss | **0.02714268** | 0.04932992 | 0.02950645 | 0.03834989 |
| Precision | **0.99282491** | 0.98706251 | 0.99180511 | 0.98628637 |
| Recall | **0.99281250** | 0.98671875 | 0.99175000 | 0.98593750 |
| F1-score: | **0.99280679** | 0.98677776 | 0.99175789 | 0.98594679 |

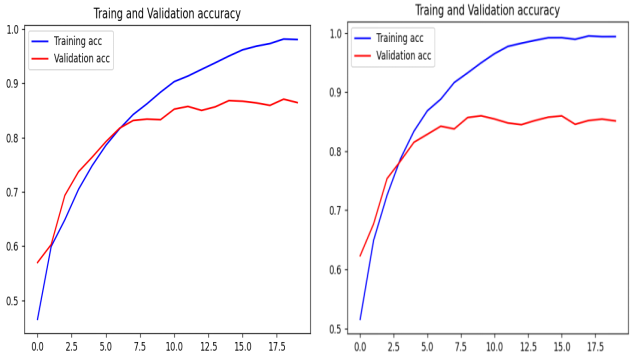
**Table 14-2**

****

**Figure 15-3**

|  |  |  |  |
| --- | --- | --- | --- |
| Learning rate test | Test17  Learning rate = 0.001 | Test13  Learning rate = 0.0001 | Test18  Learning rate = 0.00001 |
| Accuracy | 0.99243752 | **0.9928125** | 0.81093752 |
| Loss | 0.03140144 | **0.02714268** | 0.50044369 |
| Precision | 0.99245359 | **0.99282491** | 0.82842600 |
| Recall | 0.99243750 | **0.99281250** | 0.81093750 |
| F1-score: | 0.99243109 | **0.99280679** | 0.81134984 |

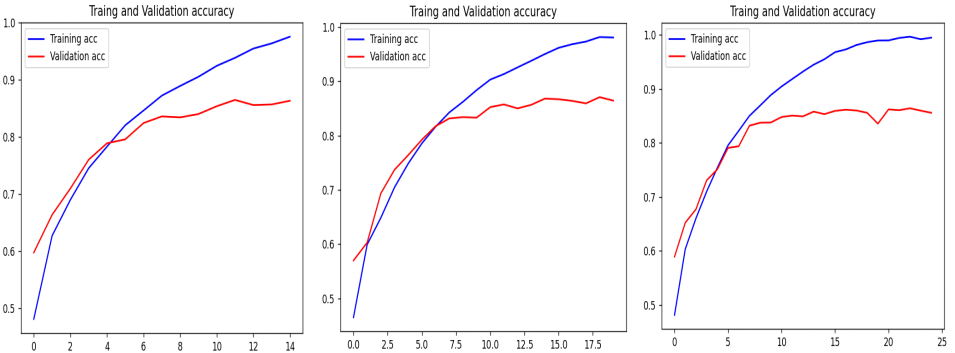
**Table 14-3**

****

**Figure 15-4**

|  |  |  |
| --- | --- | --- |
| batch\_size test | Test13  batch\_size = 128 | Test19  batch\_size = 64 |
| Accuracy | **0.99281250** | 0.99078124 |
| Loss | **0.02714268** | 0.03393398 |
| Precision | **0.99282491** | 0.99081683 |
| Recall | **0.99281250** | 0.99078125 |
| F1-score: | **0.99280679** | 0.99076797 |

Table 14-4

****

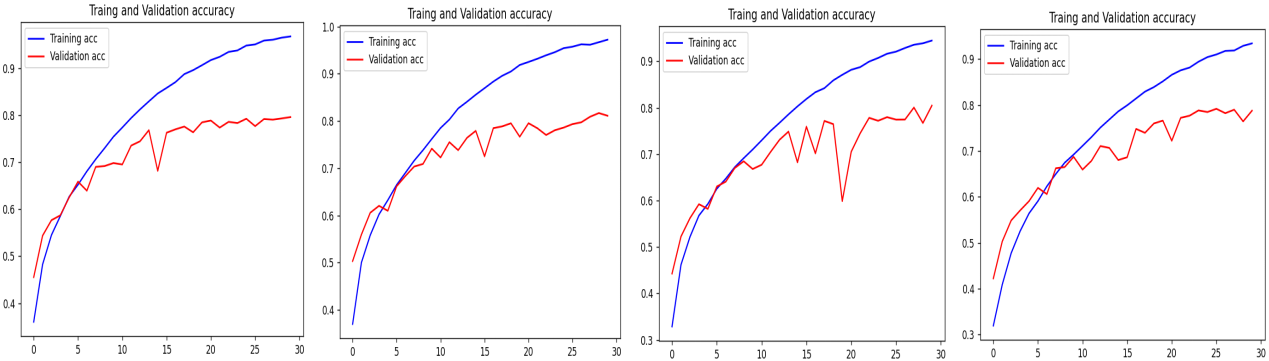
**Figure 15-5**

|  |  |  |  |
| --- | --- | --- | --- |
| Epochs test | Test20  Epochs=15 | Test13  Epochs=20 | Test21  Epochs=25 |
| Accuracy | 0.98953127 | **0.99281250** | 0.98500001 |
| Loss | 0.03012043 | **0.02714268** | 0.05304170 |
| Precision | 0.98952721 | **0.99282491** | 0.98530150 |
| Recall | 0.98953125 | **0.99281250** | 0.98500000 |
| F1-score: | 0.98952580 | **0.99280679** | 0.98496603 |

**Table 14-5**

* + 1. Evaluation of the MobileNet model

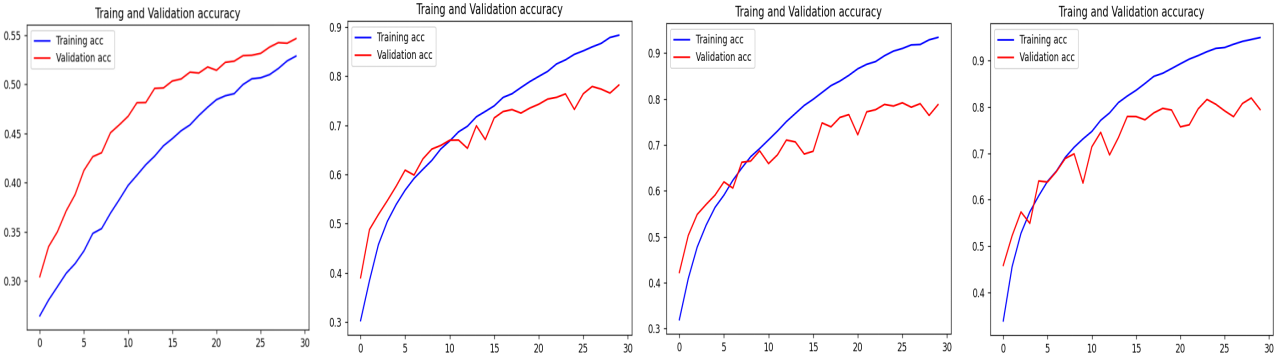
The optimal results were obtained with learning rate = 0.002, dropout = 0.3 and batch\_size = 32, epochs = 30, after a number of tuning sessions with control variables.Figure 16-1 to figure 16-4 show the change of accuracy of MobileNet parameter adjustment. Table 15-1 to table 15-4 show the data of the evaluation metrics when AlexNet adjusts parameters.



**Figure 16-1**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Drop out  (119 row) test | Test22  Drop out = 0.5 | Test23  Drop out = 0.4 | Test24  Drop out = 0.3 | Test25  Drop out = 0.2 |
| Accuracy | 0.92906248 | 0.93515623 | **0.95484376** | 0.93515622 |
| Loss | 0.19979401 | 0.18739666 | **0.14991681** | 0.22915682 |
| Precision | 0.93235556 | 0.93704296 | **0.95528154** | 0.93876346 |
| Recall | 0.92906250 | 0.93515625 | **0.95484375** | 0.93515625 |
| F1-score: | 0.92939266 | 0.93529014 | **0.95484705** | 0.93536748 |

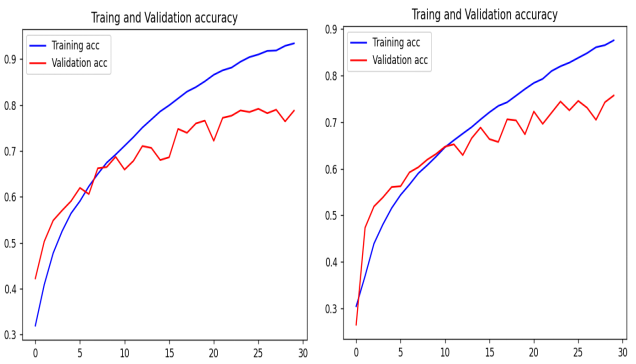
**Table 15-1**

****

**Figure 16-2**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Learning rate test | Test26  Learning rate = 0.003 | Test24  Learning rate = 0.002 | Test27  Learning rate = 0.001 | Test28  Learning rate = 0.0001 |
| Accuracy | 0.91796875 | **0.92906248** | 0.90765625 | 0.56203126 |
| Loss | 0.24142819 | **0.19979401** | 0.23847621 | 0.91068792 |
| Precision | 0.92485517 | **0.93235555** | 0.90797017 | 0.60413553 |
| Recall | 0.91796875 | **0.92906250** | 0.90765625 | 0.56203125 |
| F1-score: | 0.91569420 | **0.92939266** | 0.90653308 | 0.54518059 |

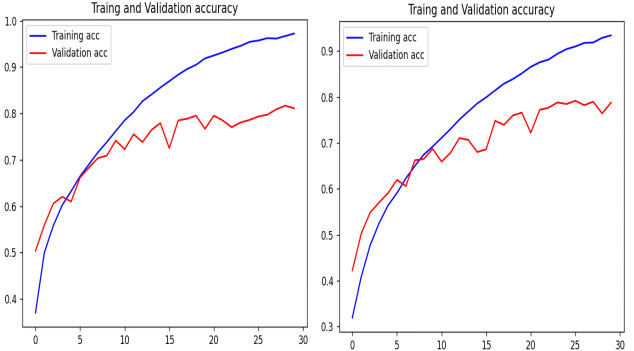
**Table 15-2**

****

**Figure 16-3**

|  |  |  |
| --- | --- | --- |
| batch\_size test | Test24  batch\_size = 32 | Test29  batch\_size = 64 |
| Accuracy | **0.92906248** | 0.89406251 |
| Loss | **0.19979401** | 0.27115225 |
| Precision | **0.93235555** | 0.89418793 |
| Recall | **0.92906250** | 0.89406250 |
| F1-score: | **0.92939266** | 0.89272331 |

**Table 15-3**

****

**Figure 16-4**

|  |  |  |
| --- | --- | --- |
| Epochs test | Test30  Epochs =25 | Test24  Epochs =30 |
| Accuracy | 0.86140626 | **0.92906248** |
| Loss | 0.36389976 | **0.19979401** |
| Precision | 0.88382730 | **0.93235555** |
| Recall | 0.86140625 | **0.92906250** |
| F1-score: | 0.86233129 | **0.92939266** |

**Table 15-4**

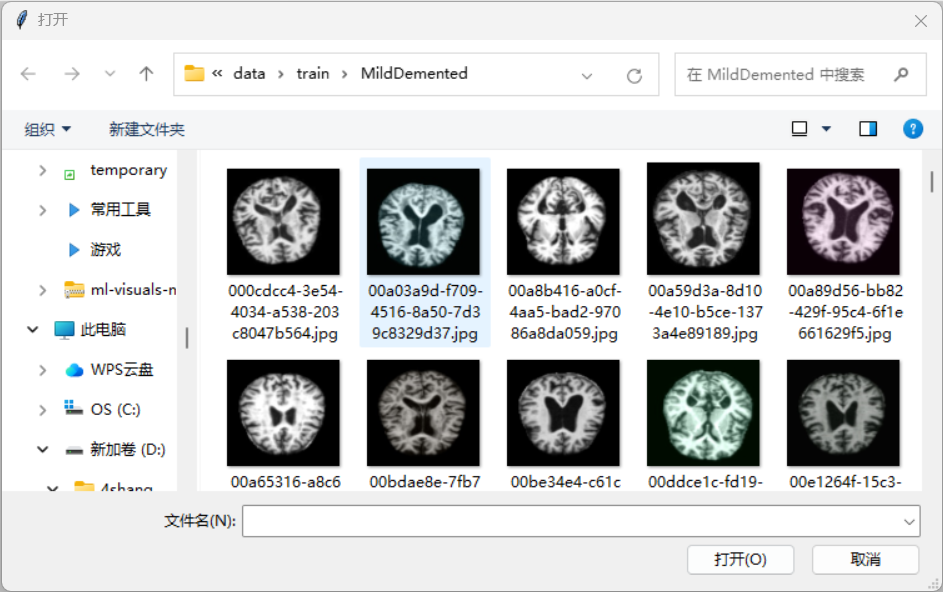
Table16 shows the best results of three individual models after adjustment and training. Compared with the ensemble project, the ensemble strong learner obtains higher evaluation data than the single model, and the ensemble effect of the project is confirmed. Meanwhile, compared to other experimenters' experiments, the accuracy of this project has been significantly improved. However, due to differences in the original dataset and experimental equipment, this comparison can only be used as a reference and cannot be used as the final comparison result.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | ResNet | AlexNet | MobileNet | Ensemble | Loddo et. al., [36] | Li et. al., [37] | Razzak et. al., [38] |
| Accuracy | 0.99281250 | 0.99546875 | 0.95484375 | **0.9965625** | **0.9867** | **0.9861** | **0.979** |
| Loss | 0.02714268 | 0.01434312 | 0.14991681 | **-** | **-** | **-** | **-** |
| Precision | 0.99282491 | 0.99548415 | 0.95528153 | **0.99656293** | **-** | **-** | **-** |
| Recall | 0.99281250 | 0.99546875 | 0.95484375 | **0.9965625** | **-** | **-** | **-** |
| F1-score | 0.99280679 | 0.99547028 | 0.95484705 | **0.99656138** | **-** | **-** | **-** |

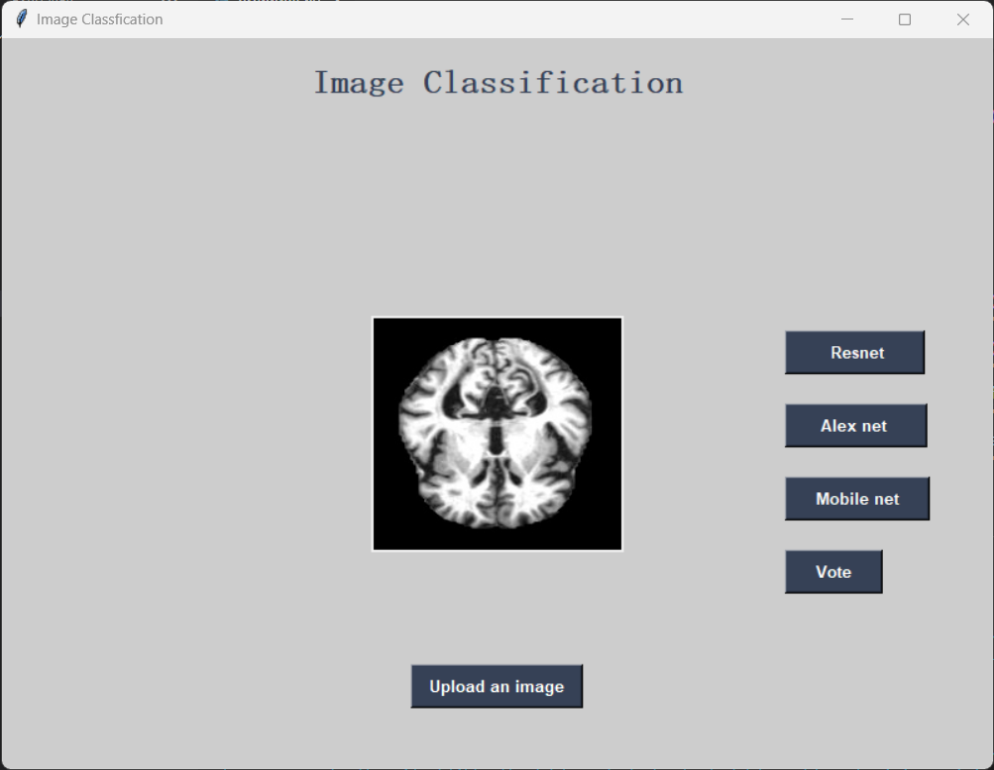
**Table 16: Final comparison**

* 1. GUI

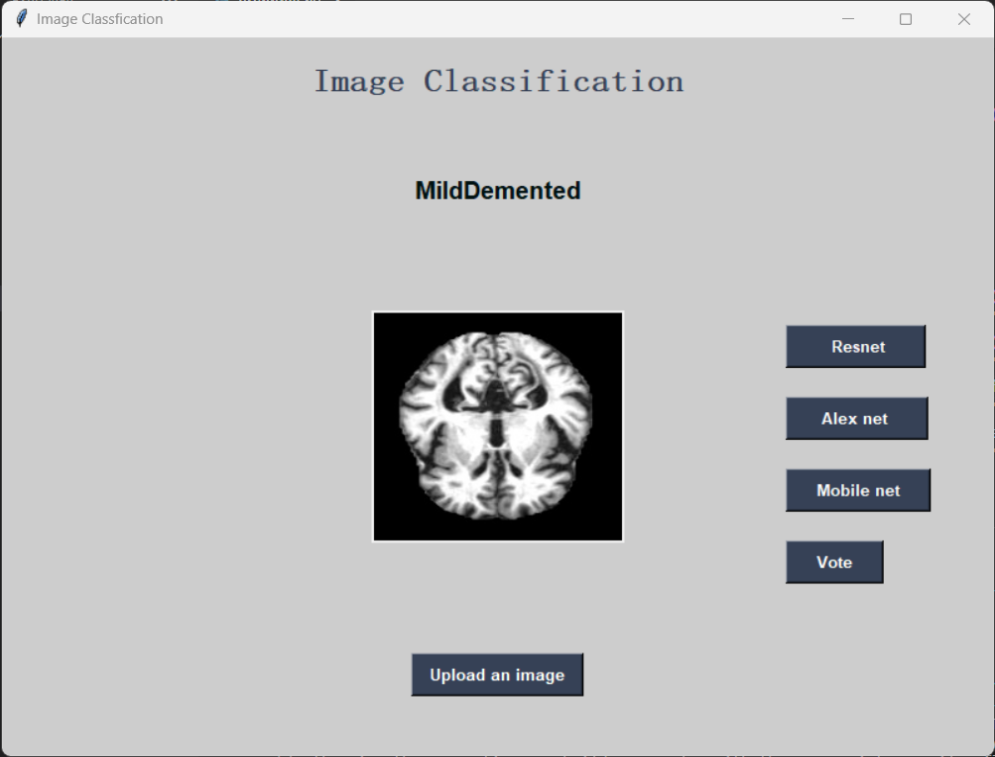
The project is based on the implementation of four classifications of Alzheimer's MRI images while also building the Graphics User Interface, which enables the uploading of files and the selection of models to complete the recognition of image types. Figure 17-1 to figure 17-3 show the project GUI.

****

**Figure 17-1**

****

**Figure 17-2**

****

**Figure 17-3**

# Professional issue

## Project Management

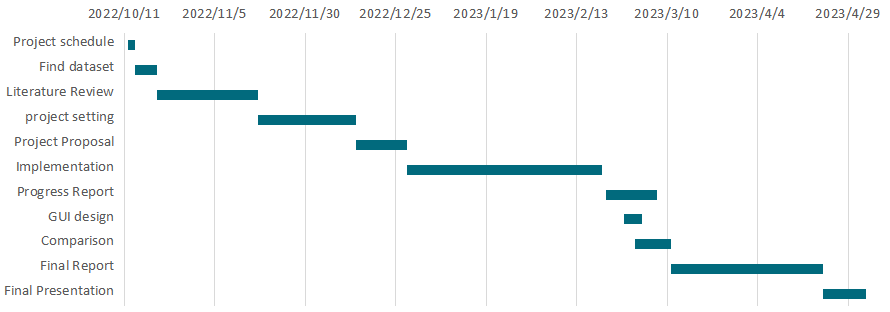
1. 1. 1. Activities

Table 18 shows the details of the activities.

|  |  |  |  |
| --- | --- | --- | --- |
| Activities | Details | Completion time | Completion status |
| Project schedule | Set a time schedule for the project | 2 days | completed |
| Find dataset | [Augmented Alzheimer MRI Dataset V2 | Kaggle](https://www.kaggle.com/datasets/uraninjo/augmented-alzheimer-mri-dataset-v2) | 6 days | completed |
| Literature Review | Project background research | 4 days | completed |
| Single model literature summary | 4 days | completed |
| Multi-model literature summary | 4 days | completed |
| Ensemble model literature summary | 4 days | completed |
| Model selection, ensemble algorithm selection | 12 days | completed |
| project setting | Create Project version management | 2 days | completed |
| Requirements Analysis | 1 days | completed |
| Complete computer environment configuration and processing of data sets | 5 days | completed |
| Establishing the report framework | 3 days | completed |
| Train three models and decide on the direction of model adjustment | 14 days | completed |
| Determine the evaluation metrics of the model | 2 days | completed |
| Project Proposal | Adjustment of project proposal report according to teacher's comments | 14 days | completed |
| Implementation | Adjusting the structure of the three models based on the results of the previous trains | 25 days | completed |
| Adjust model hyperparameters according to evaluation metrics | 10days | completed |
| Combining three models using the bagging algorithm | 14 days | completed |
| Adjust ensemble model hyperparameters according to evaluation metrics | 5 day | completed |
| Progress Report | Adjustment of project progress report according to teacher's comments | 14 days | completed |
| GUI design |  | 5 days | completed |
| Comparison | Summarize individual models, ensemble models, and compare data from other experiments. | 10 days | completed |
| Final Report | Adjustment of project final report according to teacher's comments | 42 days | completed |
| Final Presentation |  | 14 days | completed |

**Table17 :The Details of the Activities**

## Schedule

Completed 

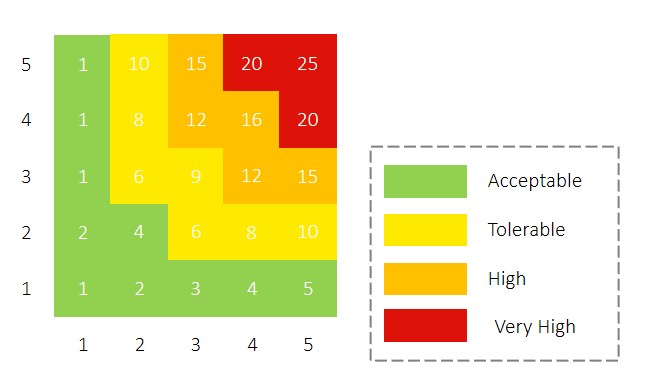
**Figure 18: Schedule Gantt Diagram**

## Project Data Management

## Project Deliverables

* Weekly report (completed)
* The project proposal (completed)
* Code: Image classification and diagnosis system of Alzheimer's disease (completed)
* The progress report (completed)
* Project presentation : PPT and poster (completed)
* Final report - 10000 words (completed)

## Risk Analysis



**Figure 19: Risk Severity Matrix**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Risk ID** | **Potential Risk** | **Cause ID** | **Potential Causes** | **Severity** | **Likelihood** | **Risk** |
| R1.1 | Late in deadline | C1.1.1 | Poor time management | 3 | 5 | 15 |
| C1.1.2 | Illness | 1 | 2 | 2 |
| C1.1.3 | Technology problems | 5 | 5 | 25 |
| R1.2 | Operating speed | C.1.2.1 | Insufficient video memory | 1 | 5 | 5 |
| R1.3 | Algorithm selection | C1.3.1 | Defects of bagging algorithm | 1 | 5 | 5 |
| R1.4 | Picture pixels | C1.4.1 | Picture pixels are 64\*64 due to device limitations, which may result in lower accuracy. | 2 | 5 | 10 |
| R1.5 | Model selection | C1.5.1 | MobileNet is often used on moving objects | 1 | 3 | 3 |
| R1.6 | High cost | C1.6.1 | The amount of computation is huge | 3 | 5 | 15 |
| C1.6.2 | Longer model training time | 3 | 5 | 15 |
| R1.7 | Problems with the models | C1.7.1 | Models are prone to problems such as over-fitting | 5 | 4 | 20 |
| C1.7.2 | Performance of multiple model ensemble is difficult to control | 5 | 4 | 20 |
| R1.8 | Problems with the item data loss | C1.8.1 | Data mismanagement leading to deficiencies | 5 | 2 | 10 |
| R1.9 | Professional Issues | C1.9.1 | Legitimacy of data sources, such as data sets | 3 | 2 | 6 |

**Table18 : Risk Analysis**

|  |  |  |
| --- | --- | --- |
| **Cause ID** | **Mitigation ID** | **Mitigation** |
| C1.1.1 | M1.1.1 | Do not make too detailed daily arrangements, but take the number of days as the standard, and more strictly abide by the schedule. |
| C1.1.2 | M1.1.2 | Adjust the project schedule reasonably according to the situation. |
| C1.1.3 | M1.1.3 | Learn more about deep learning through various channels such as the Internet or teachers. |
| C.1.2.1 | M1.2.1 | Adjust the appropriate model frame as well as the size of the input and the model's hyperparameters. |
| C1.3.1 | M1.3.1 | Selecting a Base Classifier with High Stability |
| C1.4.1 | M1.4.1 | Adjust the appropriate model hyperparameters |
| C1.5.1 | M1.5.1 | When choosing another model, choose the two models that are more commonly used for image recognition. |
| C1.6.1 | M1.6.1 | Adjust the appropriate model frame as well as the size of the input and the model's hyperparameters. |
| C1.6.2 | M1.6.2 |
| C1.7.1 | M1.7.1 | Adjust the appropriate model frame and the model's hyperparameters. |
| C1.7.2 | M1.7.2 |
| C1.8.1 | M1.8.1 | Keep project versioning updated. |
| C1.9.1 | M1.9.1 | Choose legitimate and open source data. |

**Table19 : The Mitigation of the Risk Analysis**

## Professional Issues

Regardless of any project, various legal, social, humanitarian, and environmental factors should be considered, especially when the project uses open source data, so the legitimacy of the project should be ensured.

* Social issues

In the project, there may be social issues involved, such as the protection of personal privacy related to data, and the difficulty of the project being recognized by the public due to the undeveloped market. The main purpose of creating this project is to help society address the social impact of Alzheimer's disease, which is also a social issue that requires constant attention.

* Legal issues

This project needs to comply with relevant regulations and standards, such as FDA regulatory requirements, HIPAA and other laws and regulations, as well as specific legal requirements of each country/region. At the same time, due to the sensitivity and privacy of medical data, the source of training data must be recognized and approved, and comply with legal and ethical norms to ensure the confidentiality and confidentiality of the data. And the data should come from diverse subjects to ensure the universality of the results.Therefore, all data presented in this project are publicly available open source data for research purposes, with the source indicated, and there is no lack of copyright or plagiarism.

* Humanitarian issues

In this project, humanitarian issues may be involved, such as how to apply the results to treat patients, how to improve the accuracy of patient diagnosis, and how the project can help reduce the burden on healthcare workers.

* Environmental issues

Due to different regional resources, there are differences in the incidence of Alzheimer's disease, which may lead to uneven disease types or large differences in incidence rate in different regions, which may affect the training of models. At the same time, how to reduce energy consumption in data use and processing, and how to optimize algorithms to reduce the use of computing resources also need to be considered.

# Conclusion

During the course of the project, three models were used: ResNet, MobileNet and AlexNet. By fine-tuning the model and combining their predictions, it is possible to achieve a reliable, accurate quadrilateral diagnosis of MRI images of Alzheimer's disease. At the same time, by comparing the performance of single model and ensemble learning model, it is found that ensemble learning has significant advantages, which proves the superiority of ensemble learning from multiple indicators. This is a meaningful project given the increasing impact of Alzheimer's disease on society and individuals. The program's 99.65% accuracy of the quadrotaxa can be used as one of the most trusted diagnostic tools available to healthcare professionals, and the GUI is designed to enable healthcare professionals and patients and their families to make better use of the tool.

* Future Plan

In this project, due to the limitation of equipment, the size of the picture is limited and the adjustment of the model is too simple, which leads to the need for further adjustment of the equipment and model in the future to make it more trustworthy for medical staff. First, the equipment needs to be able to perform larger calculations, the processing of data sets is more refined, and the model debugging and training also need to be better matched and optimized. Recently completed a better GUI design, which has been recognized by the public.

# References

1. An, N., Ding, H., Yang, J., Au, R., & Ang, T. F. A. (2020). Deep ensemble learning for Alzheimer's disease classification. Journal of Biomedical Informatics, 105, 103411. <https://doi.org/10.1016/j.jbi.2020.103411>
2. Bi, X., Liu, W., Liu, H., & Shang, Q. (2021). Artificial Intelligence-based MRI Images for Brain in Prediction of Alzheimer’s Disease. Journal of Healthcare Engineering, 2021, 1–7. https://doi.org/10.1155/2021/8198552
3. Vemuri, P., & Jack, C. R. (2010). Role of structural MRI in Alzheimer’s disease. Alzheimer’s Research & Therapy, 2(4), 23. <https://doi.org/10.1186/alzrt47>
4. Fox N. C., Schott J. M. (2004). Imaging cerebral atrophy: normal ageing to Alzheimer's disease. Lancet 363, 392–394. 10.1016/S0140-6736(04)15441-X [[Abstract](http://europepmc.org/article/MED/15074306)] [[CrossRef](https://dx.doi.org/10.1016/S0140-6736(04)15441-X" \t "https://europepmc.org/article/MED/pmc_ext)] [[Google Scholar](https://scholar.google.com/scholar_lookup?journal=Lancet&title=Imaging+cerebral+atrophy:+normal+ageing+to+Alzheimer's+disease&author=N.+C.+Fox&author=J.+M.+Schott&volume=363&publication_year=2004&pages=392-394&pmid=15074306&doi=10.1016/S0140-6736(04)15441-X&" \t "https://europepmc.org/article/MED/pmc_ext)]
5. Falardeau M. Respect towards people with Alzheimer's disease [J]. Soins Gerontol, 2011(91): 10-12.23.
6. Atiya M, Hyman B T, Albert M S, et al. Structural magnetic resonance imaging in established and prodromal Alzheimer disease: a review[J].Alzheimer Dis Assoc Disord, 2003, 17(3): 177-195.
7. Saima R,Mohamad H,Muhammad A I, et al. A review on neuroimaging-based classification studies and associated feature extraction methods for Alzheimer's disease and its prodromal stages[J].NeuroImage, 2017(3):57-76.
8. Yan, W., Qu, G., Hu, W., Abrol, A., Cai, B., Qiao, C., Plis, S. M., Wang, Y.-P., Sui, J., & Calhoun, V. D. (2022). Deep Learning in Neuroimaging: Promises and challenges. IEEE Signal Processing Magazine, 39(2), 87–98. <https://doi.org/10.1109/MSP.2021.3128348>
9. LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. Nature, 521(7553), 436-444.
10. Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning. MIT press.
11. Schmidhuber, J. (2015). Deep learning in neural networks: An overview. Neural networks, 61, 85-117.
12. Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems (pp. 1097-1105).
13. LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11), 2278-2324.
14. Sermanet, P., LeCun, Y., & Chintala, S. (2013). Convolutional neural networks applied to house numbers digit classification. In International Conference on Pattern Recognition (pp. 3288-3291). IEEE.
15. Nielsen, M. (2015). Neural networks and deep learning: A textbook. Determination Press.
16. Aggarwal, C. C. (2018). Neural networks and deep learning. Springer.
17. Bishop, C. M. (1995). Neural networks for pattern recognition. Oxford university press.
18. Liu, X., He, J., Gao, J., Deng, L., Duh, K., & Wang, Y. (2019). Ensemble deep learning: A review. Neurocomputing, 338, 4-16. doi: 10.1016/j.neucom.2019.01.034
19. Muazzam Maqsood; Faria Nazir; Umair Khan; Farhan Aadil; Habibullah Jamal; Irfan Mehmood; Oh-Young Song;  "Transfer Learning Assisted Classification And Detection Of Alzheimer's Disease Stages Using 3D MRI Scans",   SENSORS (BASEL, SWITZERLAND),  2019.
20. Yosra Kazemi; Sheridan K. Houghten;  "A Deep Learning Pipeline to Classify Different Stages of Alzheimer's Disease from FMRI Data",   2018 IEEE CONFERENCE ON COMPUTATIONAL INTELLIGENCE IN ...,  2018.  (IF: 3)
21. Bumshik Lee; Waqas Ellahi; Jae Young Choi;  "Using Deep CNN with Data Permutation Scheme for Classification of Alzheimer's Disease in Structural Magnetic Resonance Imaging (sMRI)",   IEICE TRANS. INF. SYST.,  2019.  (IF: 3)
22. Hina Nawaz; Muazzam Maqsood; Sitara Afzal; Farhan Aadil; Irfan Mehmood; Seungmin Rho;  "A Deep Feature-based Real-time System for Alzheimer Disease Stage Detection",   MULTIMEDIA TOOLS AND APPLICATIONS,  2020.  (IF: 3)
23. Achraf Ben Miled; Taoufik Yeferny; Amira ben Rabeh;  "MRI Images Analysis Method for Early Stage Alzheimer's Disease Detection",   ARXIV,  2020.
24. Chao Li; Quan Wang; Xuebin Liu; Bingliang Hu;  "An Attention-Based CoT-ResNet With Channel Shuffle Mechanism for Classification of Alzheimer's Disease Levels",   FRONTIERS IN AGING NEUROSCIENCE,  2022.
25. Liu, M., Tang, J., Yu, W., & Jiang, N. (2021). Attention-Based 3D ResNet for Detection of Alzheimer’s Disease Process (pp. 342–353). <https://doi.org/10.1007/978-3-030-92185-9_28>
26. Fulton, L., Dolezel, D., Harrop, J., Yan, Y., & Fulton, C. (2019). Classification of Alzheimer’s Disease with and without Imagery Using Gradient Boosted Machines and ResNet-50. Brain Sciences, 9(9), 212. https://doi.org/10.3390/brainsci9090212/
27. Lu, X., Wu, H., & Zeng, Y. (2019). Classification of Alzheimer’s disease in MobileNet. Journal of Physics: Conference Series, 1345(4), 042012. https://doi.org/10.1088/1742-6596/1345/4/042012
28. Shahwar, T., Zafar, J., Almogren, A., Zafar, H., Rehman, A. U., Shafiq, M., & Hamam, H. (2022). Automated Detection of Alzheimer’s via Hybrid Classical Quantum Neural Networks. Electronics, 11(5), 721. <https://doi.org/10.3390/electronics11050721>
29. Alex Fedorov; R Devon Hjelm; Anees Abrol; Zening Fu; Yuhui Du; Sergey Plis; Vince D. Calhoun;  "Prediction Of Progression To Alzheimer's Disease With Deep InfoMax",   ARXIV-CS.LG,  2019.  (IF: 3)
30. Mosleh Hmoud Al-Adhaileh;  "Diagnosis and Classification of Alzheimer's Disease By Using A Convolution Neural Network Algorithm",  2021.
31. Heta Acharya; Rutvik Mehta; Dheeraj Kumar Singh;  "Alzheimer Disease Classification Using Transfer Learning",   2021 5TH INTERNATIONAL CONFERENCE ON COMPUTING ...,  2021.
32. Haijing Sun; Anna Wang; Wenhui Wang; Chen Liu;  "An Improved Deep Residual Network Prediction Model for The Early Diagnosis of Alzheimer's Disease",   SENSORS (BASEL, SWITZERLAND),  2021.
33. Mohammed, B. A., Senan, E. M., Rassem, T. H., Makbol, N. M., Alanazi, A. A., Al-Mekhlafi, Z. G., Almurayziq, T. S., & Ghaleb, F. A. (2021). Multi-Method Analysis of Medical Records and MRI Images for Early Diagnosis of Dementia and Alzheimer’s Disease Based on Deep Learning and Hybrid Methods. Electronics, 10(22), 2860. https://doi.org/10.3390/electronics10222860
34. Loris Nanni; Matteo Interlenghi; Sheryl Brahnam; Christian Salvatore; Sergio Papa; Raffaello Nemni; Isabella Castiglioni;  "Comparison Of Transfer Learning And Conventional Machine Learning Applied To Structural Brain MRI For The Early Diagnosis And Prognosis Of Alzheimer's Disease",   FRONTIERS IN NEUROLOGY,  2020.  (IF: 3)
35. Peng Zhang; Shukuan Lin; Jianzhong Qiao; Yue Tu;  "Diagnosis of Alzheimer's Disease with Ensemble Learning Classifier and 3D Convolutional Neural Network",   SENSORS (BASEL, SWITZERLAND),  2021.
36. Andrea Loddo; Sara Buttau; Cecilia Di Ruberto;  "Deep Learning Based Pipelines for Alzheimer's Disease Diagnosis: A Comparative Study and A Novel Deep-ensemble Method",   COMPUTERS IN BIOLOGY AND MEDICINE,  2021.  (IF: 3)
37. Minglei Li; Yuchen Jiang; Xiang Li; Shen Yin; Hao Luo;  "Ensemble of Convolutional Neural Networks and Multilayer Perceptron for The Diagnosis of Mild Cognitive Impairment and Alzheimer's Disease",   MEDICAL PHYSICS,  2022.
38. Imran Razzak; Saeeda Naz; Hamid Alinejad-Rokny; Tu N Nguyen; Fahmi Khalifa;  "A Cascaded Mutliresolution Ensemble Deep Learning Framework for Large Scale Alzheimer's Disease Detection Using Brain MRIs",   IEEE/ACM TRANSACTIONS ON COMPUTATIONAL BIOLOGY AND ...,  2022.