成都理工大學 中英合作办学
CDUT Sino-British Collaborative Education

OXFORD BROOKES UNIVERSITY

UNDERGRADUATE PROJECT REPORT

Project Title:	**Inception Separable CNN for Retinal Disease Classification**
Surname:	**Ke**
First Name:	**Yifan**
Student Number:	**202018010304**
Supervisor Name:	**Dr. Grace Ugochi Nneji**
Module Code:	**CHC 6096**
Module Name:	**Project**
Date Submitted:	**May 6, 2024**

Chengdu University of Technology Oxford Brookes College

Chengdu University of Technology

BSc (Single Honours) Degree Project

Programme Name: **Computer Science**

Module No.: **CHC 6096**

Surname: **Ke**

First Name: **Yifan**

Project Title: **Inception Separable CNN for Retina Disease Classification**

Student No.: **202018010304**

Supervisor: **Dr. Grace Ugochi Nneji**

2ND Supervisor (if applicable): **Not Applicable**

Date submitted: **May 6, 2024**

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

At

Chengdu University of Technology Oxford Brookes College

Declaration

Student Conduct Regulations:

Please ensure you are familiar with the regulations in relation to Academic Integrity. The University takes this issue very seriously and students have been expelled or had their degrees withheld for cheating in assessment. It is important that students having difficulties with their work should seek help from their tutors rather than be tempted to use unfair means to gain marks. Students should not risk losing their degree and undermining all the work they have done towards it. You are expected to have familiarised yourself with these regulations.

https://www.brookes.ac.uk/regulations/current/appeals-complaints-and-conduct/c1-1/

Guidance on the correct use of references can be found on www.brookes.ac.uk/services/library, and also in a handout in the Library.

The full regulations may be accessed online at

https://www.brookes.ac.uk/students/sirt/student-conduct/

If you do not understand what any of these terms mean, you should ask your Project Supervisor to clarify them for you.

I declare that I have read and understood Regulations C1.1.4 of the Regulations governing Academic Misconduct, and that the work I submit is fully in accordance with them.

Signature: Ke Yifan Date: May 6, 2024

REGULATIONS GOVERNING THE DEPOSIT AND USE OF OXFORD BROOKES UNIVERSITY MODULAR PROGRAMME PROJECTS AND DISSERTATIONS

Copies of projects/dissertations, submitted in fulfilment of Modular Programme requirements and achieving marks of 60% or above, shall normally be kept by the Oxford Brookes University Library.

I agree that this dissertation may be available for reading and photocopying in accordance with the Regulations governing the use of the Oxford Brookes University Library.

Signature: Ke Yifan Date: May 6, 2024

Acknowledgment

Thanks to all my friends and teachers who provided me with ideas and help to complete this project. First of all, I would like to thank Dr. Grace Ugochi Nneji, whose guidance and advice have been crucial throughout this project. Her expertise and insights provided me with guidance to better understand the problem and find solution. And thanks to Dr. Joojo Walker for her support during the writing of this paper. His guidance not only helped me improve my writing skills, but also provided me with thesis templates and ideas that enabled me to organize the structure and content of the paper more systematically. In addition, I would like to thank Chengdu University of Technology and Oxford Brookes University for providing me with rich learning resources and a good learning environment. These resources provided me with academic support and the necessary knowledge base, which laid the foundation for me to successfully complete this assignment.

Table of contents

List of Figures

List of Tables

Abstract

Retinal diseases have significant impact on vision health and need to be recognized in time for more effective treatment, artificial recognition of retinal disease is not efficient and accurate. Therefore, this project aims to develop a retinal disease classification model which combines the Inception model and separable convolutional neural network (CNN) architectures. In this project, cataract, diabetic retinopathy, glaucoma and normal retinal optical coherence tomography (OCT) images can be identified by the proposed model.

This project compares existing classification models for retinal disease recognition from relevant papers, to establish a baseline for assessment and provide research direction for the project. Through iterative hyper-parameter tuning and model optimization, the proposed model accuracy is about 91.9%, and the validation loss is about 0.057. The average precision, recall rate and F1-score of the model reaches to 0.91. It compares the proposed model with classic models like ResNet and VGG16, and models proposed by other papers using the same dataset, result shows the proposed model in this project has relatively good performance. In addition, the project includes a graphical user interface (GUI) to improve user operability. Through these evaluation indicators, it shows that the proposed model can provide a better solution for the field of retinal disease classification.

The aim of this project is to improve the efficiency of treatment for medical professionals by speeding up the diagnosis of retinal diseases. And patients will receive more professional treatment through more accurate and timely diagnosis.

Keyword: Retinal disease classification, Convolutional Neural Network (CNN), Inception model, Separable CNN, Medical image analysis

Abbreviations

CNN	Convolutional Neural Networks
DME	Diabetic Macular Edema
AMD	Age-related Macular Degeneration
OCT	Optical Coherence Tomography
CNV	Choroidal Neovascularization
GUI	Graphical User Interface
VGG	Visual Geometry Group
TN	True Negative 、
TP	True Positive
FP	False Positive
FN	False Negative
ReLU	Rectified Linear Unit
ResNet	Residual Networks
GPU	Graphical Processing Units
CPU	Central Processing Units
HTML	Hypertext Mark-up Language
CSS	Cascading Style Sheets
API	Application Programming Interface
LIME	Local Interpretable Model-agnostic Explanations

Glossary

Optical Coherence Tomography: Imaging technique using light to capture detailed cross-sectional views of biological tissues, commonly used in ophthalmology.

Inception Model: Deep learning architecture designed for efficient utilization of computational resources by utilizing various sizes of convolutional filters.

Convolutional Neural Network: Deep learning algorithm particularly effective for image recognition tasks, mimicking the visual cortex's processing in the brain.

Separable CNN: Convolutional Neural Network architecture employing depthwise and pointwise convolutions to reduce computational cost and parameters.

Data Augmentation: Technique to artificially increase the size of a dataset by applying transformations like rotation or flipping to original data samples.

Data Pre-processing: Preparing data for analysis by cleaning, transforming, and organizing it to enhance the performance of machine learning models.

Fine-Tuned: Process of adjusting pre-trained models on a specific dataset to improve performance for a particular task.

Weight Initialisation: Setting initial values for the parameters (weights) of a neural network before training to avoid convergence issues and accelerate learning.

Batch Normalisation: Technique to improve the training speed and stability of neural networks by normalizing the inputs of each layer.

Dropout: Regularization technique in neural networks where randomly selected neurons are ignored during training to prevent overfitting.

Confusion Matrix: Table used to evaluate the performance of a classification model, showing the count of true positive, true negative, false positive, and false negative predictions.

GoogLeNet: developed by Google, is famous for its Inception modules, which efficiently use parameters and perform well in deep learning tasks.

VGGNet: Created by the Visual Geometry Group at Oxford, is known for its deep architecture and simple convolutional layer stacking, excelling in image recognition tasks.

Chapter 1 Introduction

1.1 Background

Retinal disease is a widespread condition in all age groups with serious consequences such as cataracts which manifest as white cloudy areas in the lens of the eye leading to loss of vision in the eye [1], Glaucoma causes damage to the optic nerve [2], Diabetic retinopathy is a complication of diabetes mellitus which leads to loss of vision in many middle aged and elderly people [3]. Different kinds of retinal diseases can present different two-dimensional views in optical coherence tomography (OCT) scans, thus retinal diseases can be identified by recognising retinal OCT images [4]. Early detection and accurate classification of these retinal diseases are crucial for timely intervention and treatment. However, it is not efficient to identify and classify retinal disease images only manually. The main goal of this project is to develop a retinal disease classification model that aims to improve the early diagnosis of retinal diseases, thereby providing more timely and accurate treatment. The retinal disease classes that can be identified in the project include cataract, diabetic retinopathy, glaucoma, while normal retinal OCT images is also one of the classes that can be identified in the project. In order to achieve highly accurate and efficient retinal disease image classification that meets the needs of healthcare professionals and patients, the project employs a separable Convolutional Neural Network (CNN) with the Inception model, which provides high performance while efficiently managing the model size and computational cost.

1.2 Aim

This project aims to develop an advanced retinal disease classification model using Inception and the separable CNN model. The main aim is to improve the accuracy and efficiency of the diagnosis of retinal diseases through a more efficient automatic classification model. Figure 1 shows the samples OCT images of different types of retinal disease and the normal.

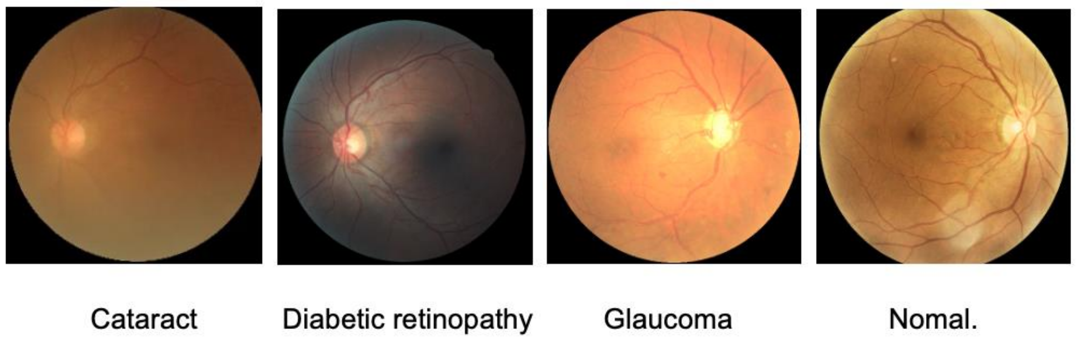


Figure 1 samples of retinal disease OCT images

1.3 Object

The objectives of this project conclude tasks showing below:

a. Search and compare existing retinal disease classification models.

b. Prepare retinal image dataset and perform data preprocessing on the dataset.

c. Construct deep learning model based on separable CNN with Inception model.

d. Train, validate and evaluate the performance of the model for precise diagnosis of retina disease.

e. Optimize the model to improve accuracy.

f. Compare the performance of different model with the proposed model using the same dataset.

g. Present the graphic user interface (GUI) to the direct audience.

1.4 Project Overview

1.4.1 Scope

The purpose of this project is to delve into the integration of separable CNN with Inception model for retinal classification. The proposed model is designed to analyse retinal images and define them into different disease categories that can help the healthcare professionals in the early detection.

The project has potentially implications on the field of ophthalmology and medical area, where early diagnosis is critical to preventing vision loss and reducing the healthcare burden.

1.4.2 Audience

This project will be of great benefits to the following audience：

For the government, the project can reduce the pressure on medical resources and improve medical services through its high efficiency, so that to improve the quality of life of the public. The project will also help drive medical innovation, leading to the development of new technologies for diagnosing retinal diseases.

For medical professionals such as doctors, ophthalmologists and medical researchers, the model can be applied to the early detection of retinal diseases and provide assistance for diagnosis, to improve treatment efficiency and reduce the pressure of diagnosis and treatment.

In addition, by obtaining more accurate and timely diagnosis, and more effective treatment options, patients with retinal diseases will receive better treatment and get the faster recovery, thereby improving their health status and quality of life.

Chapter 2 Background Review

This section will discuss the different research analysed for the diagnose of retinal classification by different researchers.

The model of Meenu et al. [5] uses a depth-separable CNN and combines 2D spatial convolution and point-by-point convolution. This model requires less computation than traditional convolutional methods, effectively reducing training time and space requirements. The final model achieves an accuracy of 93.5% on the OCT2017 dataset

Kermany et al. [6] presented one kind of transfer learning approach to categorize different retinal diseases using a pretrained Inception V3 model based on ImageNet. This method achieved an accuracy of 96.6%, sensitivity of 97.8% and specificity of 97.4% in the disease classification.

Alqudah [7] proposed a novel automatic convolutional neural network (CNN) architecture. The study introduced a CNN model that specifically categorized retinal scan images into four categories: CNV, DME, AMD, and Drusen. Adam optimizer was used to train the model, and the average accuracy of the final model for the classification of the four categories reached 95.3%.

Rajagopalan et al. [8] proposed CNN model consists of 5 convolutional layers along with 3 fully connected layers. The convolutional layers are used to extract features from the input OCT image, while the pooling layers help to reduce the size of the network, improve computational efficiency and reduce overfitting. Also, the hyperparameters of the model are optimised to improve the performance. This model was evaluated on Mendeley database and obtained 95.7% accuracy.

Mahendran et al. [9] harnessed machine learning algorithms for the examination of retinal diseases. Their approach involves the utilization of a decision tree classifier to distinguish between normal and disease-affected images. The dataset images underwent bilateral filtering techniques to mitigate noise. Subsequently, Otsu's segmentation was applied to isolate the macula region, and the segmented structures were fed into the classifier, culminating in an impressive accuracy of 92%.

Najeeb et al. [10] proposed an algorithm based on a single-layer convolutional neural network structure with the Adam optimizer, which is a lightweight model that reduces the computational complexity and resource consumption to some extent, while achieving an accuracy of 95.66%. However, the model does not perform very well in some retinal diseases such as DME, and there are classification errors.

The deep multilayer CNN model proposed by Bhadra and Kar [11] is used to classify three different retinal diseases CNV, DME, DRUSEN as well as healthy retina. In this model, two convolutional operations were performed in each layer, ReLU was used as an activation function in the dense layer and SoftMax was used as an activation function in the output layer. This model performed with 96.5% accuracy.

The study by Intaraprasit et al. [12] classified four types of retinal images: cataract, glaucoma, diabetic retinopathy, and normal retina, using the MobileNetV2 deep learning network, this model performs strongly on the dataset across all metrics. With a training accuracy of 99.83%, a validation accuracy of 99.52%.

Additionally, Table 1 displays a summary of different research carried out for retinal disease classification and their approaches with performances.

Researchers	**Year**	**Techniques**	**Accuracy**
Meenu et al. [5]	2022	Depthwise separable CNN	93.5%
Kermany et al. [6]	2018	Inception V3	96.6%
Alqudah et al. [7]	2019	Novel Automatic CNN structure	95.3%
Rajagopalan et al. [8]	2021	Deep CNN framework	95.7%
Mahendran et al. [9]	2020	Decision tree classifier	92%
Najeeb et al. [10]	2018	Single layer CNN structure.	95.66%

Bhadra and Kar [11]	2020	Deep multi-layered CNN	96.5%
Intaraprasit et al. [12]	2023	MobileNetV2	99.8%

Table 1 Literature Review of different approach about retinal disease classification

Comprehensively comparing these models, Kermany et al. [6], Alqudah [7] provided a migration learning approach to improve classification accuracy using pre-trained models. Najeeb et al. [10] utilised a network with a lightweight structure to reduce the computational cost. MobileNetV2 employed by Intaraprasit et al. [12] has the highest accuracy on the dataset. Overall, each model has its own merits and can be useful for retinal disease classification and timely diagnosis in the medical field.

According to the analysis and comparison of the above studies, the model proposed in this project is based on Inception and separable CNN, which combines the advantages of different networks. The operation of depthwise separable convolution can reduce the number of parameters and computation. Compared with the MobileNetV2 used by Intaraprasit et al. [12], the model in this project is more lightweight and can be better applied in resource-constrained environments. And the model adopts modular design, and each Inception module has multiple branches, which helps to capture more features and improve the performance of the model. In summary, the model proposed in this project can reduce the computational cost while maintaining the accuracy, which can be well adapted to the task of retinal image classification.

Chapter 3 Methodology

3.1 Approach

This section discusses the dataset used in this project and its corresponding data augmentation strategies and data pre-processing used in the model, gives the structures of the proposed model and the explanation of the main technologies in model. And the concept and calculation methods of the evaluation strategies used in this project also discusses in this section.

3.1.1 Model Architecture

The model proposed in this project is based on Inception and Separable CNN. The definition of each technique will be introduced below, and it also shows how they can be used in this proposed model. Finally, the proposed model structure is discussed in depth, and the Inception module and separable CNN structure in the model structure are analysed separately.

3.1.1.1 InceptionNet

Inception network is a deep learning network proposed by Google research team, which can solve the problems of large amount of calculation and parameters in traditional convolutional neural networks effectively [13]. Inception is a multi-branch structure, which allows it to extract features at different scales. Generally, the Inception module consists of multiple convolution kernels and pooling layers of different sizes, which usually contain 1x1 convolutional layers to reduce the dimension of the feature map and reduce the amount of computation and the number of parameters. At the same time, the introduction of 3x3 convolution, 5x5 convolution and 3x3 pooling can increase the width of the network and adapt to the ability of the network to adapt to the scale. Finally, deep concatenation is used to synthesize features after each block to obtain nonlinear attributes and improve the expression ability of the model. At the same time, pooling layer and stride are often used in the network to reduce the spatial resolution of the feature map, reduce the amount of calculation and memory consumption [14-15].

The standard structure of Inception network is presented as the Figure 2 and Figure 3.

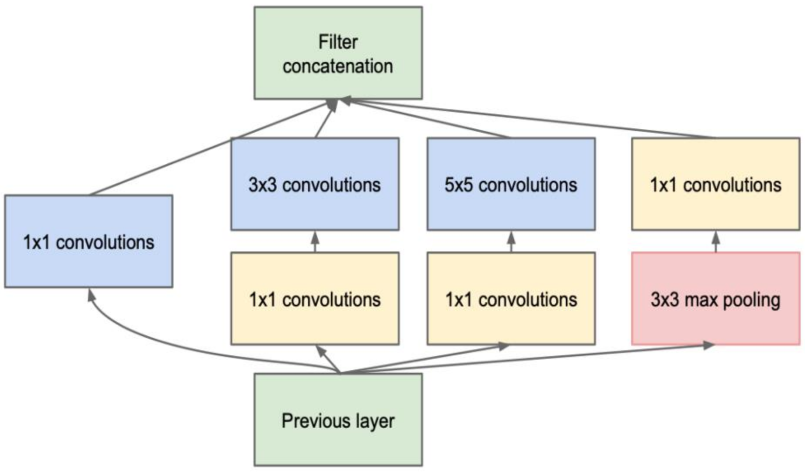


Figure 2 Inception module with dimension reductions [15]

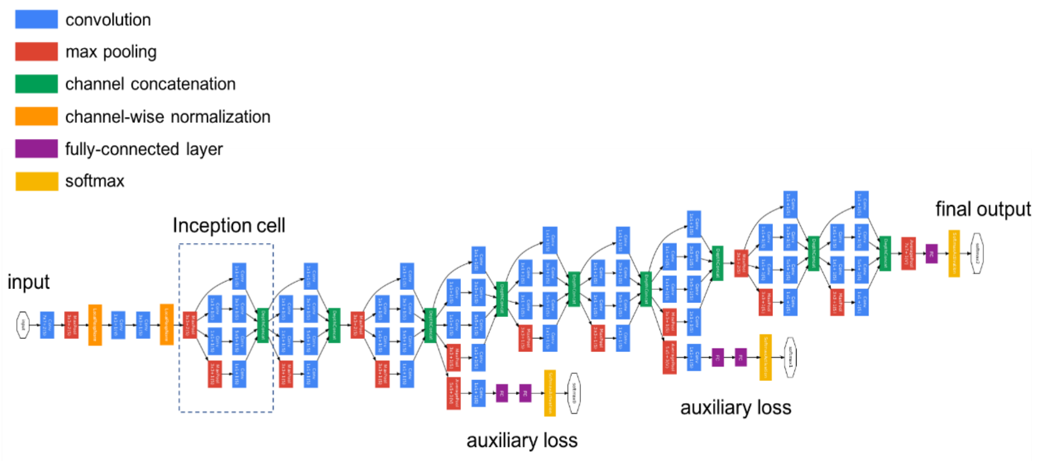


Figure 3 Inception network [15]

Multiple inception modules are included in the model proposed in this project. The Inception A module contains multiple 1x1, 5x5, and 3x3 kernels and performs channel merging. The Inception B and inception C modules contain 3x3 and 7x7 convolution kernels, where branching and channel merging are performed. The Inception D module does the branching and pooling operations. The Inception E module contains 1x1, 3x3, and 5x5 convolution kernels with branching and channel merging.

The design of these Inception modules enables the network to capture feature information of different scales at the same time, thereby improving the representation ability of the model for complex features. Through this way of parallel operation and channel merging, the Inception module solves the problem of gradient disappearance and parameter explosion when the number of network layers increases to a certain extent, and also improves the computational efficiency and performance of the model.

3.1.1.2 Inception V3

Inception V3 is one of the pretrained models on the TensorFlow. After the Inception V1 and Inception V2, it rethinks the original structure of computer vision [16]. Inception v3 is a convolutional neural network used to aid image analysis and object detection, originally as a module of GoogLeNet [17]. Inceptionv3 is designed to allow deeper networking while also keeping the number of parameters from growing too large. Its computation cost is only about 2.5 higher than that of GoogLeNet and it is still much more efficient than VGGNet [18].

The principle is that activati1ons that are close to each other are highly correlated, and dimensionality reduction of these activations before polymerization results in similar local characterization. With proper decomposition, the resulting parameters are more decoupled, and the number of parameters is reduced, thus speeding up training.

The structure of Inception V3 model is shown in Figure 4.

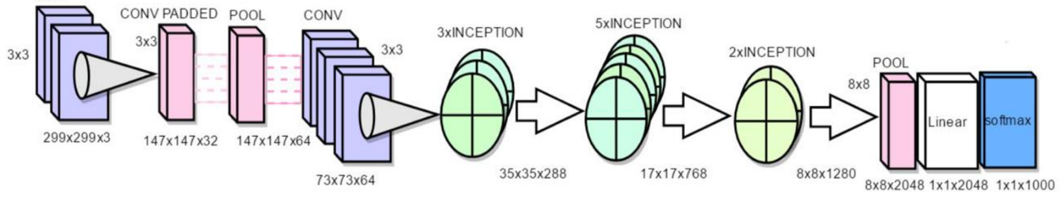


Figure 4 The model progress in InceptionV3 network [16]

In this project, the proposed model also contains the relevant technology of Inception V3, which includes different scale Convolution kernels and pooling operations, and implements a similar operation of Depthwise Separable Convolution. These operations help to improve the performance of the model.

3.1.1.3 Depthwise Separable CNN

The operations of depthwise separable convolutions include depthwise convolution and pointwise convolution, whose biggest advantage is that they are computationally efficient. The calculation of Depthwise Convolution is relatively simple. It uses a convolution kernel for each channel of the input feature map, and then concatenates the outputs of all convolution kernels to obtain its output. After the depth convolution, the pointwise convolution operation applies a 1x1 convolution kernel to each single-channel feature map. This 1x1 convolution operation will combine and mix the single-channel feature maps generated by the deep convolution to produce the final output feature map [19-20].

Unlike space-separable convolution, depth separable convolution deals with nuclei that cannot be decomposed into two smaller nuclei [21]. And unlike standard CNN, where convolution is only applied to a single channel at a time, Deep-wise operates on M channels. So, the size of the filter/kernel here is Dk x Dk x 1. Assuming that there are M channels in the input data, then M such filters are required. The output size is Dp x Dp x M, this operations are shown as Figure 5 and Figure 6.

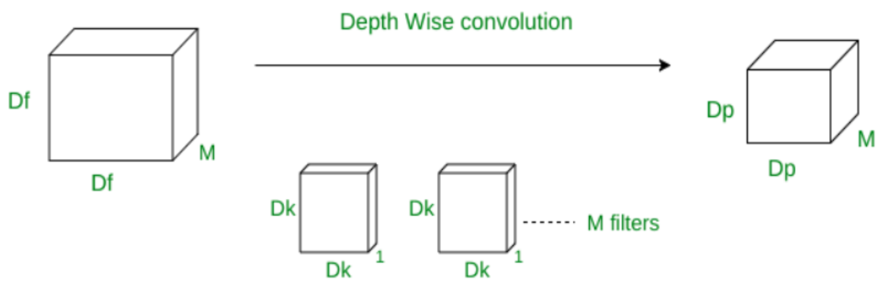


Figure 5 Operation of Depthwise convolution

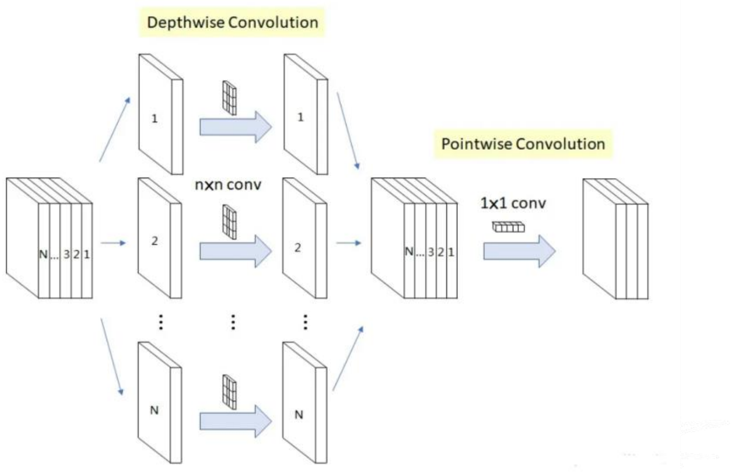


Figure 6 Operation of Depthwise convolution 2

In the model proposed in this project, such depthwise separable convolutions are applied to certain branches in the Inception module, as well as in other convolutional layers. Firstly, the intermediate feature map is generated by the depth convolution, and then the information of each channel is fused by the pointwise convolution. Then, the final output feature map is obtained by Batch normalization and ReLU activation function.

3.1.1.4 Pooling layer

a. Average Pooling

Average Pooling is a pooling operation that calculates the average of the patches of a feature map and uses it to create a subsampled (pooled) feature map. It is usually used after the convolution layer. It adds a small amount of translation invariance - meaning that a small amount of translation over an image does not significantly affect the value of most pooled outputs. It extracts features more smoothly than maximum pooling, which extracts more obvious features such as edges [22]. Figure 7 shows the working principles of average pooling.

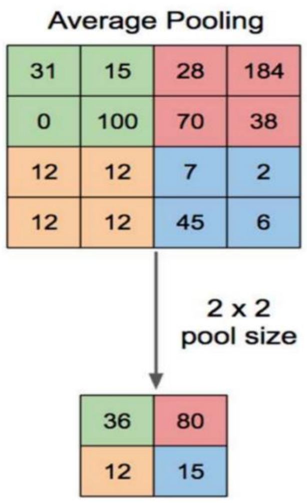


Figure 7 Working principle of average pooling

In this model, the operation of Average pooling is realized by calling the correlation function. Its parameters are shown in Table 2 below.

Kernel size	Stride
2	3

Table 2 Parameters of average pooling

This operation reduces the dimensionality of the feature map, but is smoother than Max pooling, which can help mitigate the overfitting problem and improve the generalization ability of the model.

b. Max Pooling

Max Pooling is performed by calculating the maximum value of the patch of a feature map and using it to create a subsampled (pooled) feature map. It is usually used after the convolution layer. It adds a small amount of translation invariance - meaning that a small amount of translation over an image does not significantly affect the value of most pooled outputs. Figure 8 shows the working principles of max pooling.

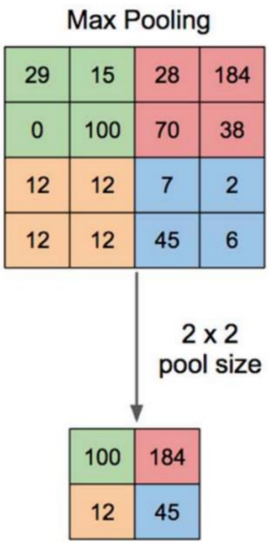


Figure 8 Working principle of max pooling

In this proposed model, the operation of Max pooling is realized by calling the related function. Its parameters are shown in Table 3 below.

Kernel size	Stride
3	2

Table 3 Parameters of max pooling

This operation helps to extract salient features in the image, retain the main information, and reduce the spatial dimension of the feature map, which is beneficial to reduce over-fitting and improve the computational efficiency of the model.

3.1.1.5 ReLU activation function

Rectified Linear Unit (ReLU) is an important method to introduce nonlinear features into deep learning models, which can effectively solve the problem of gradient disappearance and thus improve the performance of deep learning models. It explains the positive part of its argument and is one of the most popular activation functions in deep learning [23].

Its calculation formula is:

$$f(x) = \max(0, x) \qquad Equation(1)$$

And the corresponding function curve is represented as Figure 9:

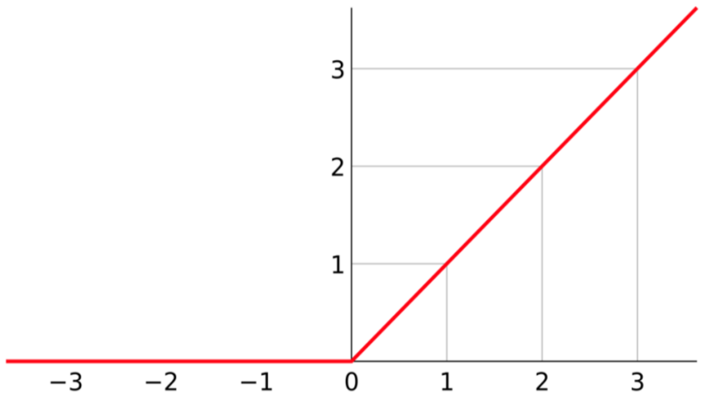


Figure 9 ReLU activation function curve

Compared with other activation functions, ReLU function has better gradient propagation. Compared with Sigmod activation function which is saturated in both directions, ReLU can better solve the problem of gradient disappearance [24]. At the same time, because of the simple calculation, it has higher computational efficiency [25].

The model proposed in this project introduces the ReLU activation function in the application of depthwise separable convolution, aiming to improve the generalization ability of the model.

3.1.2 Optimization strategy

In machine learning, optimization strategies refer to a set of methods and techniques to be adopted in order to achieve the best performance of the model. These methods and techniques aim to tune the parameters or hyperparameters of the model in order to reduce the value of the loss function, or to achieve the best value of other performance metrics. In this project, the optimization strategies used mainly include the use of Adam optimizer, the use of cross-entropy loss function, parameter update, and dropout.

a. Adam optimizer

Adam optimizer is an adaptive learning rate optimization algorithm. It can dynamically adjust the learning rate according to the gradient of each parameter, iteratively update the weights of the network according to the training data, adapt to the changes of different parameters, and help accelerate the convergence of the model [26].

In this project, Adam optimizer is selected to train the model. In the training process, the learning rate of 0.0001 is set to 0.0001 by comparing the performance with 0.001. Such a choice can speed up the convergence and improve the efficiency of the training process.

b. Cross-entropy loss function

In multi-class classification tasks, the cross-entropy loss function is often used to measure the difference between the probability distribution of the model output and the true label, which helps the model to better learn the classification boundary [27].

In the training process of this project, the loss value related to the true label is predicted by calling the defined cross-entropy function. After getting this value, the loss is backpropagated by calling the related function, the gradient of each parameter is calculated, and the model parameters are updated by using the optimizer. In these ways, the classification problem of multiple retinal image types required by the project can be effectively handled.

c. Parameter update

During model training, the introduction of parameter updates plays an important role. By updating and optimizing the parameters, the adaptability and accuracy of the model can be improved, and then the prediction ability and generalization ability of the model can be improved.

According to the above instructions, during training, the gradient of the loss function with respect to the model parameters is calculated by the backpropagation algorithm, and the model parameters are updated according to the gradient using the optimizer. By updating the model parameters, the model gradually approaches the optimal solution during the retraining process.

d. Dropout

Overfitting occurs when a complex neural network is trained on a small dataset and performs well on the training set but poorly on the validation set. The purpose of dropout is to prevent such overfitting and improve the performance of the neural network by preventing the feature detectors from working together [28].

Dropout is also used in this project to prevent overfitting. Firstly, the spatial dimension of the feature map is reduced to 1*1 by the average pooling operation, and then the dropout operation is applied. In this operation, if the model belongs to the training phase, dropout will lose the output of some neurons with a certain probability to prevent overfitting. If it is in the verification phase, dropout will not take effect, which can ensure that the feature information of the image can be fully utilized to achieve higher verification accuracy in the verification phase. By applying dropout, the proposed model not only reduces the risk of overfitting, but also ensures the generalization ability.

3.1.3 Model Architecture

The overall structure of the model proposed in this project includes the Stem Network, multiple Inception modules, and the final fully connected layer. The Stem Network is responsible for the preliminary feature extraction and dimensionality reduction processing of the input image, including a series of convolutional layers and pooling operations. Inception module is the core part of the model, which is divided into five modules A, B, C, D and E. Each module extracts and combines features through different branch structures, and uses depthcnv class to realize depthwise separable convolutions to multi-scale extraction of features and parameter efficiency optimization. Figure 10 displays the overall model architecture of the proposed model in this project.

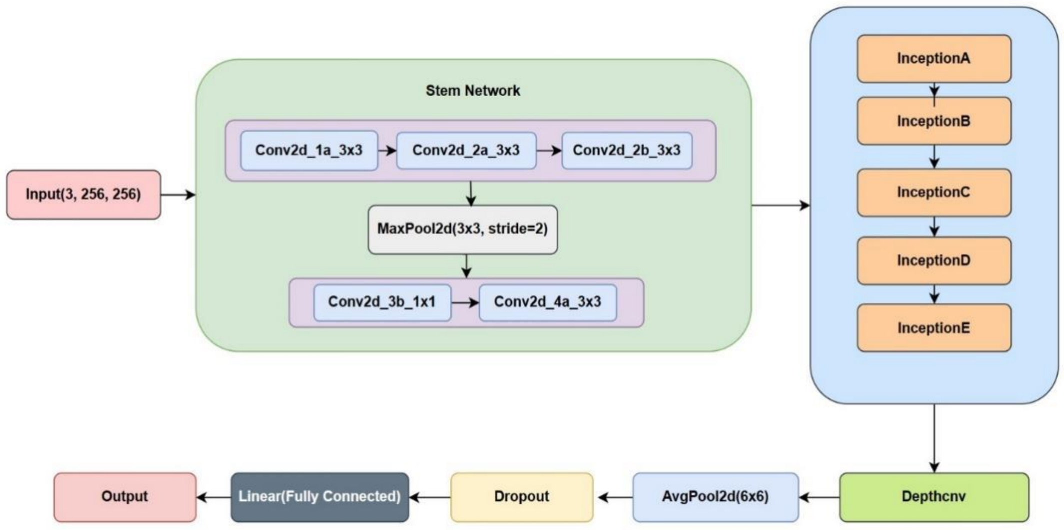


Figure 10 Architecture of model in the project

The Inception A module contains four branches: 1x1 convolution, 5x5 convolution, double 3x3 convolution, and pooling branch. Branches are connected by convolution and pooling operations. Finally, the outputs of all branches are concatenated in the channel dimension. The 1x1 convolution branch uses 1x1 convolution for feature extraction and dimension reduction of the input, and the number of output channels is 64. The 5x5 convolution branch first performs feature extraction by 1x1 convolution, then increases the receptive field by 5x5 convolution, and finally the number of output channels is 64. The double 3x3 convolution branch extracts complex features through two layers of 3x3 convolution to increase the nonlinear expression ability of the network, and the final number of output channels is 96. The pooling branch uses the average pooling operation for feature dimensionality reduction.

Similar to the Inception A module, the structure of Inception B contains three branches: a 3x3 convolutional branch, a double 3x3 convolutional branch and a max-pooling branch. A 3x3 convolution is used for feature extraction and dimensionality reduction, and the step size is set to 2 to reduce the feature map size. The Max pooling operation is used for feature dimension reduction and size reduction.

The Inception C module includes 1x1 convolution branch, 7x7 convolution branch, dual 7x7 convolution branch, and pooling branch. The 1x1 convolution branch uses 1x1 convolution for feature extraction and dimensionality reduction, and the number of output channels is 192. The 7x7 convolution branch extracts feature through a series of 1x7 and 7x1 convolution operations, and the number of output channels is 192. The pooling branch uses the average pooling operation for feature dimensionality reduction, and the number of output channels is 192.

The architectures of module Inception A, B and C can show as the Figure 11 below:

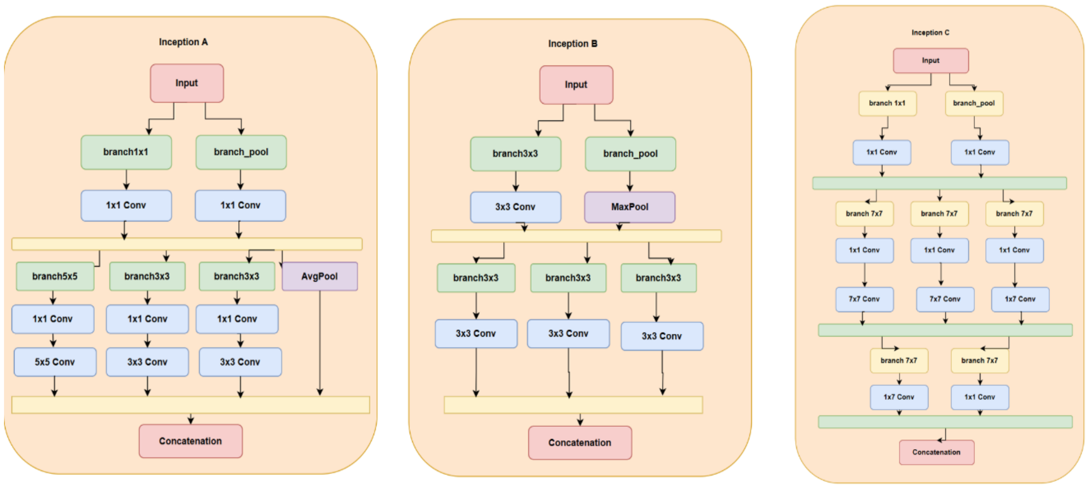


Figure 11 Architecture of Inception module A, B, C

And each parameter and their corresponding values are shown as the Table 4-6 below.

Branch	Conv idex	In_channels	Out_channels	Kernal size	padding
branch1x1	Conv1	3	64	1	-
branch5x5_1	Conv1	3	48	1	-
	Conv2	48	64	5	2
branch3x3dbl_1	Conv1	4	64	1	-
	Conv2	64	96	3	1
	Conv3	96	96	3	1
branch_pool	Conv1	3	pool_features	1	-

Table 4 Parameters of Inception A module

Branch	Conv idex	In_channels	Out_channels	Kernal size	padding	Stride
Branch3x3	Conv1	3	384	3	-	2
branch3x3dbl_1	Conv1	3	64	1	-	-
	Conv2	64	96	3	1	-
	Conv3	96	96	3	1	2
branch_pool	Conv1	3	192	1	-	-

Table 5 Parameters of Inception B module

Branch	Conv idex	In_channels	Out_channels	Kernal size	padding
branch1x1	Conv1	3	192	1	-
branch7x7	Conv1	3	channels_7x7	1	-
	Conv2	channels_7x7	channels_7x7	(1, 7)	(0, 3)
	Conv3	channels_7x7	192	(7, 1)	(3, 0)
branch7x7dbl	Conv1	3	192	1	-
	Conv2	channels_7x7	channels_7x7	(7, 1)	(3, 0)
	Conv3	channels_7x7	channels_7x7	(1, 7)	(0, 3)
	Conv4	channels_7x7	channels_7x7	(7, 1)	(3, 0)
	Conv5	channels_7x7	192	(1, 7)	(0, 3)
branch_pool	Conv1	3	192	1	-

Table 6 Parameters of Inception C module

The Inception D module includes three main branches and a pooling branch, which process the input features through different convolution operations and pooling operations to realize the extraction of multi-scale features. Finally, the processed features are concatenated in the channel dimension, which also enriches the expression ability of the model for complex features. The feature map size is reduced and the network is down-sampled by convolution and pooling operation with stride=2. This structure design can increase the nonlinearity and representation ability of the network.

The Inception E module includes four branches, which are 1x1 convolution branch, 3x3 convolution branch, double-layer 3x3 convolution branch, and pooling branch. This design helps the network to extract and fuse multi-scale and multi-level features, and improves the model's ability to process complex data.

The architecture of inception D and inception E module are shown as the Figure 12 below:

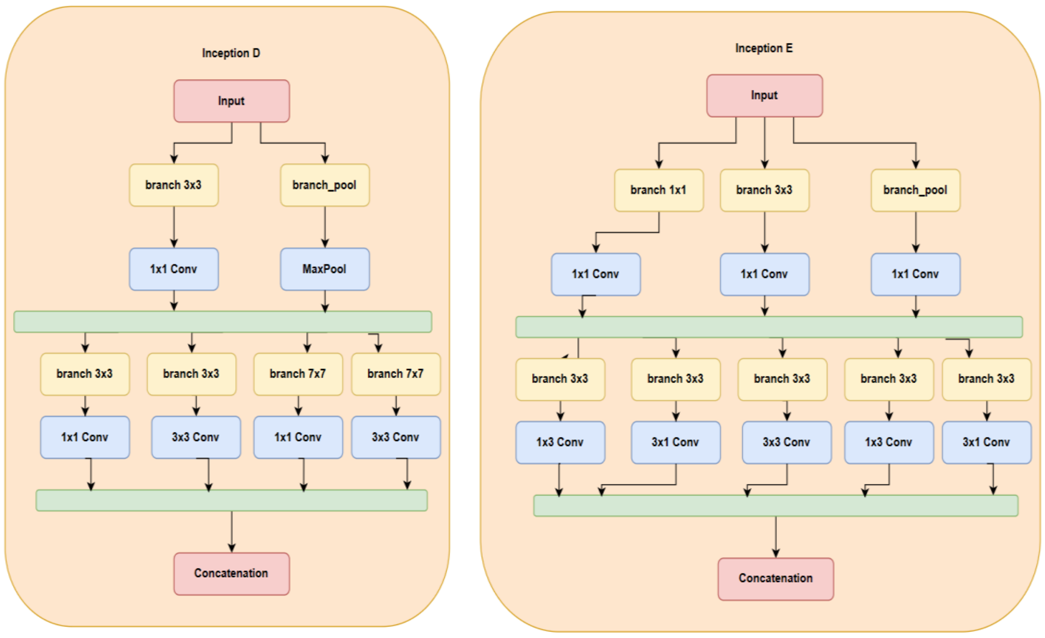


Figure 12 Architecture of Inception module D, E

Parameters of inception D and inception E modules with their corresponding values are summarized as the Table 7-8 below:

Branch	in_channels	out_channels	kernel_size	padding
branch3x3_1	3	192	1	-
branch3x3_2	192	320	3	2
branch7x7x3_1	3	192	1	-
branch7x7x3_2	192	192	(1, 7)	(0, 3)
branch7x7x3_3	192	192	(7, 1)	(3, 0)
branch7x7x3_4	192	192	3	2

Table 7 Parameters of Inception D module

Branch	in_channels	out_channels	kernel_size	padding
branch1x1	3	320	1	-
branch3x3_1	3	384	1	-
branch3x3_2a	384	384	(1, 3)	(0, 1)
branch3x3_2b	384	384	(3, 1)	(0, 1)
branch3x3dbl_1	3	448	1	-
branch3x3dbl_2	448	384	3	1
branch3x3dbl_3a	384	384	(3, 1)	(0, 1)
branch3x3dbl_3b	384	384	(3, 1)	(1, 0)
branch_pool	3	192	1	-

Table 8 Parameters of Inception E module

The architecture of the depthwise separable convolution module consists of two 3x3 convolutional layers (Conv1 and Conv2), and one 1x1 convolutional layer (Conv3). Each convolutional layer is followed by a Batch Normalization operation, followed by a ReLU activation function. Such a structure is depthwise separable convolutional structure, which reduces the number of parameters by separating spatial and channel features, thus improving the efficiency of the model. As shown in Figure 13:

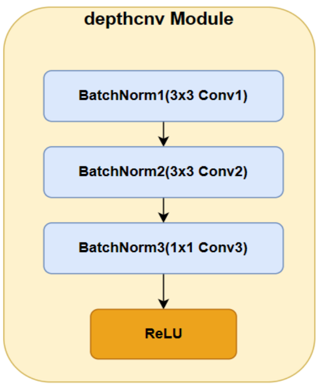


Figure 13 Depthwise Convolutional module in the proposed model

3.2 Dataset

3.2.1 About Dataset

This project uses a retinal disease classification dataset from kaggle with link: https://www.kaggle.com/datasets/gunavenkatdoddi/eye-diseases-classification/data (Accessed: November 7, 2023)

The dataset consists of Normal, Diabetic Retinopathy, Cataract and Glaucoma retinal OCT images and each class have approximately 1000 images. These images are collected from various sorces like IDRiD, Oculur recognition, HRF etc. For training this model, the dataset is divided to train set in 90% and validation set in 10%.

Figure 14 displays the distribution of each class in the selected dataset, which shows this dataset totally contains 1038 cataract images, 1098 diabetic retinopathy images, 1007 glaucoma images and 1074 normal retinal images.

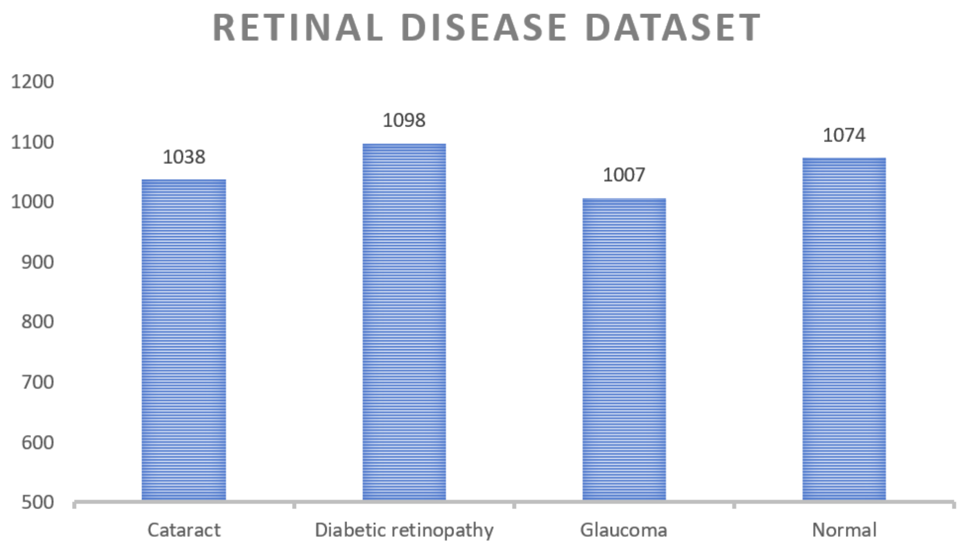


Figure 14 Image distribution of the retinal disease dataset

During training, the dataset is divided to training set and validation set, the structure of the folder in local shown as Figure 15 below, it shows that the training set contains 3799 images and the validation set contains 420 images.

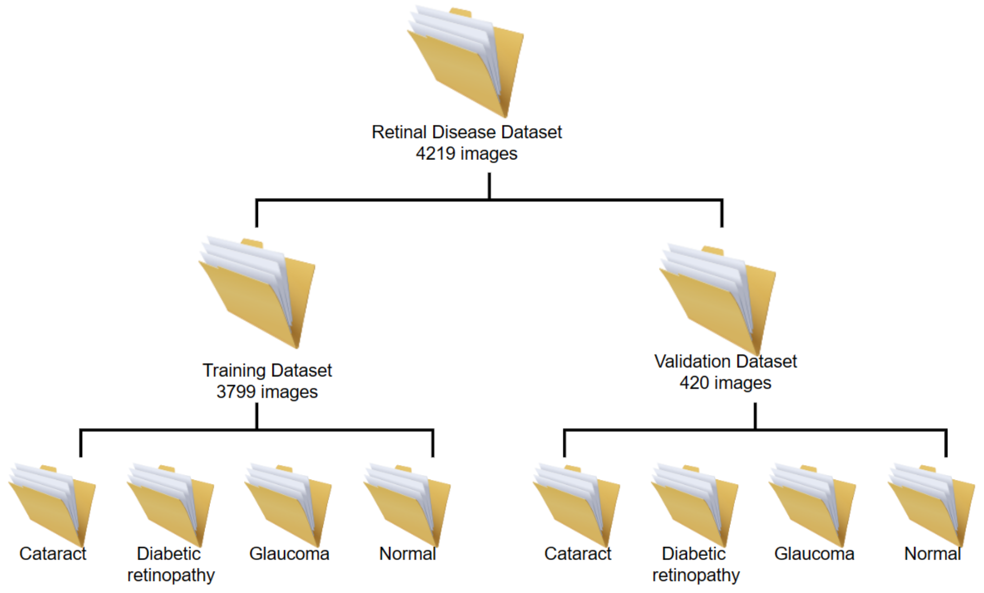


Figure 15 Folder structure of the dataset

3.2.2 Data pre-processing

In the process of model training, the use of data preprocessing technology helps to make the model better learn and understand the image features, enhance the model's processing ability and robustness to the image, and thus improve the model's performance and generalization ability. In this project, the data preprocessing techniques used are data augmentation and data normalization. The details of each strategy can be described below.

a. Data augmentation

Data augmentation is a statistical technique that allows maximum likelihood estimation from incomplete data [29]. Data augmentation is widely used in machine learning especially the image classification area, in order to reduce the overfitting of model, and improve generalization ability and performance of the model during training [30].

This technique is also used during model training in this project. First, the input image is cropped during training to resize the image to the specified size (256*256). Through this operation, the model can process images of different sizes during training, thus improving the adaptability to image scale changes. And then randomly flip the image horizontally, which allows it to generate more diverse training samples and thus reduce the dependence of the model on the left and right direction of the image. And randomly rotating the image at a certain Angle range (±20 degrees), which helps the model learn object features at different angles and increases the model's robustness to image rotation.

b. Data normalization

Normalization is a common method of data preprocessing, its main purpose is to transform the original data into a form with a unified scale, so as to improve the training effect and convergence speed of the model. By mapping the data into a specific interval range, it makes them comparable and comparable. In addition, normalization can also enhance the stability and generalization ability of the model, and avoid the instability or overfitting phenomenon caused by the large value range of a variable [31]. The normalization formula used in this project is as follow:

$$nomalized_{image} = \frac{image - mean}{std} \qquad Equation(2)$$

Where image represents the original image data and mean std is the standard deviation.

The pixel values for each channel are transformed according to the formula described above for normalization purposes. In the actual operation of this project, the pixel values of each channel were processed by normalization to ensure that each channel has a mean of 0.5 and a standard deviation of 0.5.

3.3 Technology

The techniques used in this project, including the choice of hardware and software, the details are shown in the following Table 9.

Type	**Purpose**	**Name and version**
Software	Framework of deep learning model	Pytorch
	IDE	Pycharm (Version 2023. 2)
	Operating System	Windows 11
	Version management	Git repository
	Reference management	Zotero
Hardware	Central Processing Unit (CPU)	13th Gen Intel(R) Core (TM) i7-13650HX
	Graphics Processing Unit (GPU)	NVIDIA GeForce RTX 4060

Table 9 Technologies in the project

3.4 Testing and Evaluation Plan

The evaluation strategy included in this report includes Loss, Accuracy, while the confusion matrix and various categories of performance metrics such as Precision, Recall, F1-Score are used to evaluate the model's performance in more detail. Table 10 represents the confusion matrix, while the formulas below show the various evaluation metrics derived from the confusion matrix, including recall, precision, which can show the relevance between predications and the actual results.

And the relevant formulas showing as below:

	Actual Positive	Actual Negative
Predicted Positive	TP	FP
Predicted Negative	FN	TN

Table 10 Confusion Matrix

Accuracy measures the overall performance of the model, which represents the proportion of the number of samples correctly classified on the whole dataset to the total number of samples. The formula is as follows:

$$\frac{TP + TN}{TP + TN + FP + FN} \qquad Equation(3)$$

Precision$_i$ represents the precision on a certain class i of the dataset, and it measures the proportion of samples that the model predicted to belong to class i, out of all the samples that were predicted to belong to class i. The formula is shown below:

$$Precision_i = \frac{TP_i}{TP_i + FP_i} \qquad Equation(4)$$

The following formula represents Macro-average Precision, where Precision is the precision for class i and L is the total number of classes. When evaluating multi-class classification models, it treats the performance of each class equally, regardless of the distribution of the classes in the data.

$$Precision_{macro} = \frac{\sum_{i=1}^{L} Precision_i}{L} \qquad Equation(5)$$

Recall$_i$ evaluates the recall of the model for class i during training. It measures the proportion of instances where the model successfully predicted class i to be the true class i. The formula is shown below:

$$Recall_i = \frac{TP_i}{TP_i + FN_i} \qquad Equation(6)$$

The following formula represents Macro-average Recall, where Recall_i denotes the recall of the model for class i and L is the total number of classes. Macro-average Recall can provide an overall performance metric for model evaluation in multi-class classification models.

$$Recall_{macro} = \frac{\sum_{i=1}^{L} Recall_i}{L} \qquad Equation(7)$$

The F1-Score is the reconciled mean of precision and recall and provides a comprehensive performance measure. It is useful for problems with unbalanced category distributions, and the F1 score is calculated as follows:

$$F1 - score = \frac{2 * Recall_{macro} * Precision_{macro}}{Recall_{macro} + Precision_{macro}} \qquad Equation(8)$$

3.5 Project version Management

This project will use the Git repository to store and manage all the versions, the link is: https://github.com/Kkkkyf7/Project and it is expected that the project will include 5 versions as illustrated in Table 11.

Version 1	Initial version, include the basic separable convolution and inception modules.
Version 2	Performance improvement version, improving classification accuracy by improving convolutional kernel size, number of layers, activation function, etc.
Version 3	Introduce more branches of the Inception modules or more separable convolutional layers.
Version 4	The parameters are adjusted to achieve the best performance of the model.
Version 5	Build a web application with flask and deploy the model to the web application.

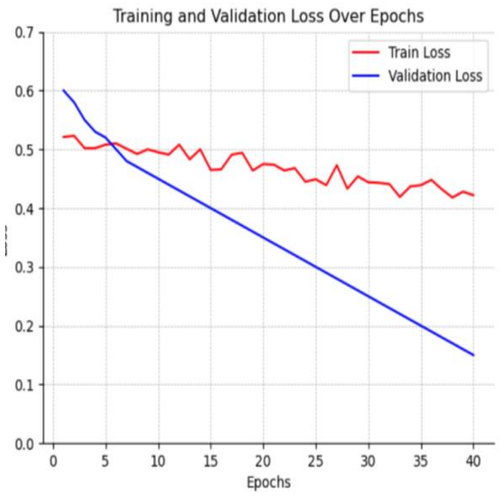
Table 11 Project version management

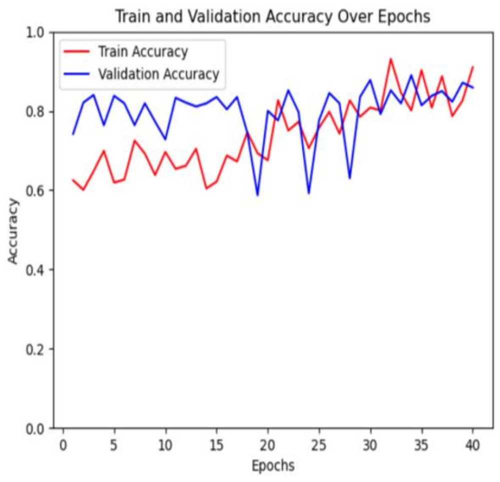
Chapter 4 Implementation and Results

This section provides detailed descriptions and documentation of results and testing. Critical evaluation and discussion of results, issues encountered constraints, limitations, and originality.

4.1 Results over different hyper-parameters

When batch size = 5, epoch = 40, the initial training loss is about 0.521, which gradually decreases during training. At the 40th epoch, the training loss stabilized at about 0.422. The initial verification accuracy is about 74.2%. During the training, the validation accuracy fluctuated, eventually reaching about 85.9% at the 40th epoch. The curves of loss and accuracy showing as in Figure 16 below.



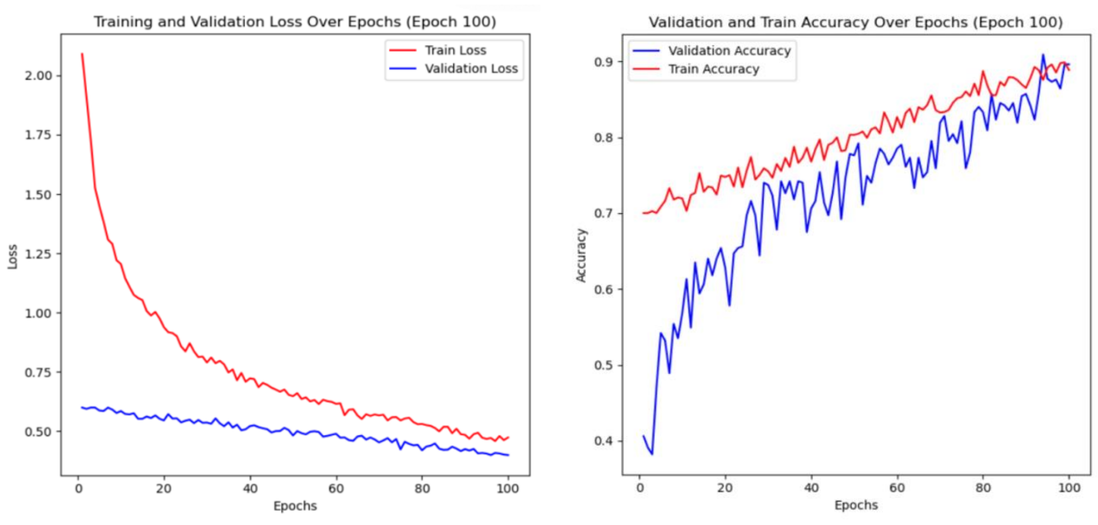


a) Train loss = 0.422, val loss = 0.152 b) Train acc = 85.9%, val acc = 85.6%

Figure 16 Accuracy and loss curve over batch size=5 and epoch=40

When batch size = 5, epoch=100, the initial training loss is about 2.09 and gradually decreased over the whole of the training. At the 100th epoch, the training loss stabilized at about 0.474. The initial verification accuracy is about 40.6%. During the training, the validation accuracy fluctuated, eventually reaching about 89.6% at the 100th epoch.

The curves of loss and accuracy showing as in Figure 17.



a) Train loss = 0.474, val loss = 0.371 b) Train acc = 89.9%, val acc = 89.6%

Figure 17 Accuracy and loss curve over epoch=100 and batch size=5

By comparing the performance of the model with epoch of 40 and 100, it is observed that the model's performance on the verification set improved with increasing training cycles. At epoch 40, training losses had decreased and validation accuracy is about 85.9%. However, after more training epochs, the training loss of model stabilized at about 0.474, and the validation accuracy further improved to about 89.6%. This means that more training can lead to improved model performance, but care should also be taken to prevent overfitting. In practice, it is important to choose the right training cycle to avoid unnecessary computational costs while improving model performance.

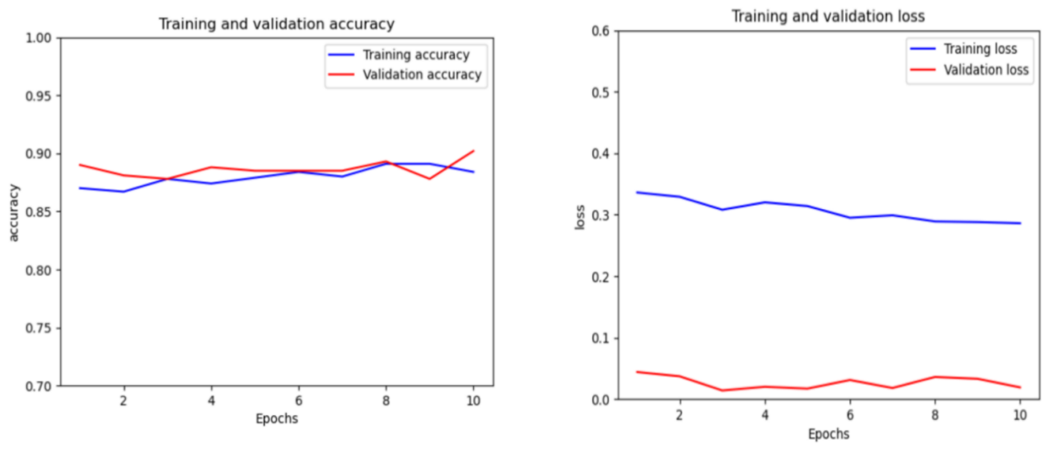
The comparison is display in Table 12.

Epoch	Validation Accuracy	Train Loss
40	85.9%	0.422
100	89.6%	0.474

Table 12 Performance of model with different epoch setting when batch size=5

Then introduce additional evaluation metrics, such as precision, recall and the F1-Score to evaluate the model, and find suitable hyper-parameters for this model by constant fine-tune.

When batch size = 10, epoch = 10, the training accuracy is stabled at about 88.4%, validation accuracy goes up to 90.2%, training loss from 0.336 to 0.286 and validation loss fluctuates in 0.017 to 0.044. Corresponding curves are shown as Figure 16, but due to the limitation of epoch setting, the figure can not show the current trend efficiently.



a) Train acc = 88.4%, val acc = 90.2%　　b) Train loss = 0.286, val loss = 0.044

Figure 18 Accuracy and loss curve over batch size=10 and epoch=10

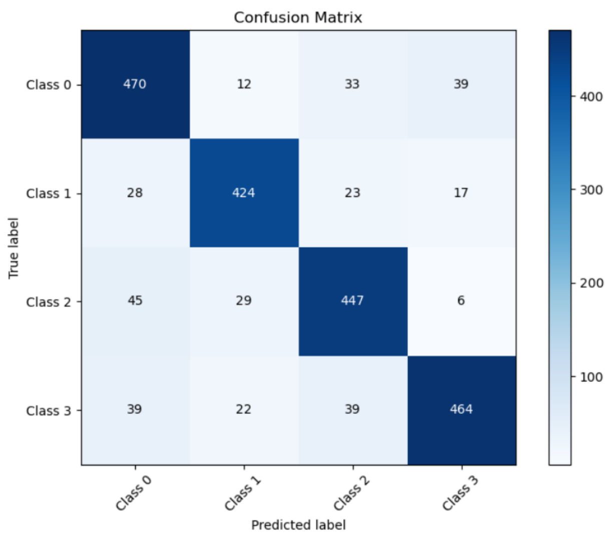
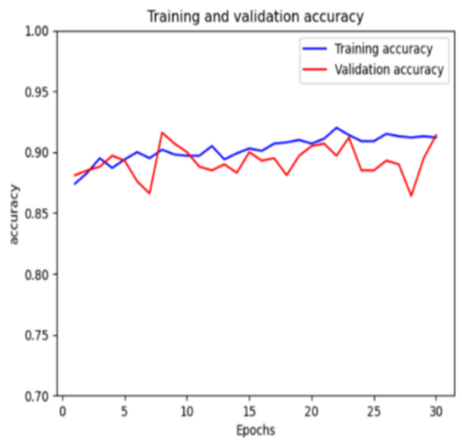
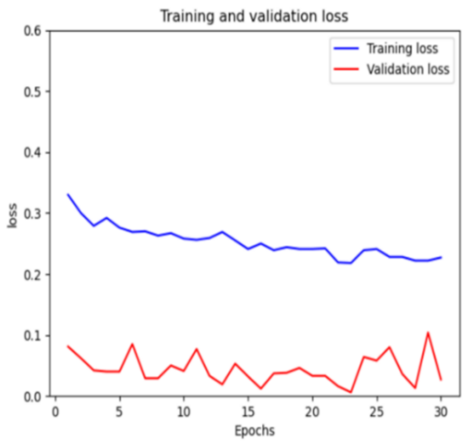


Figure 19 Confusion Matrix over epoch=10 and batch size=10

From the confusion matrix, prediction performance of the model is relatively better on the category 0 and category 3, which presented the class of cataract and glaucoma OCT images separately, this indicates that the model has a strong ability to recognize these two categories. However, the ability of classification about class 1 and class 2 which means the classification ability of diabetic retinopathy and the normal class needs to be further optimized. In order to solve this problem, the parameters need more adjustment.

Setting batch size to 70, When epoch = 30, the model converges gradually and makes significant improvement. The training loss gradually decreases from the initial 0.33 to 0.227, and the training accuracy steadily increases to 91.2%, which indicates that the model has achieved good fitting effect on the training data. At the same time, the performance of the validation set also gradually improved, the validation loss decreased from 0.081 to 0.027, and the validation accuracy increased from 88.1% to 91.4%, indicating that the model has a good generalization ability. In addition, the F1 (Macro) index increased from 0.879 to 0.913, and the recall score and precision score also showed a steady growth trend, achieving better results on several evaluation strategies. Figure 20 which contains the accuracy and loss curves can express these trends and results.



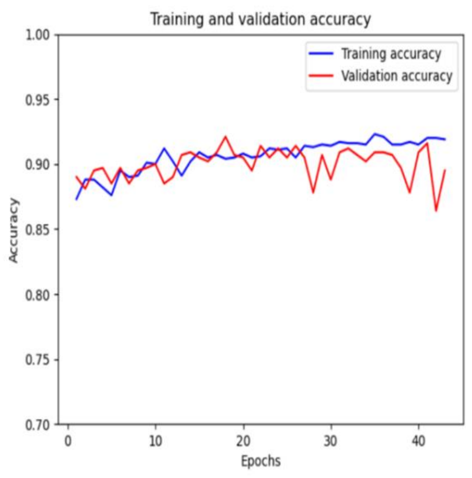


a) Train acc = 91.4%, val acc = 91.2%   b) Train loss = 0.227, val loss = 0.027

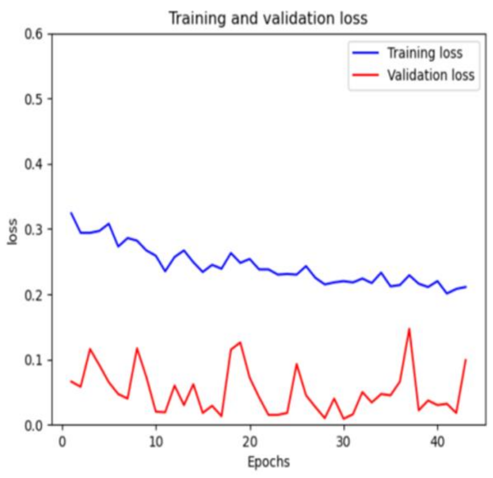
Figure 20 Accuracy and loss curve over epoch=30 and batch size=70

Then set the epoch=40 and keep the batch size=70. It shows a relatively stable convergence trend. In the initial stage, the training loss is 0.324, which gradually decreased to 0.211 as the training progressed. The training accuracy rate increased from 87.3% to 91.9%, indicating that the model's fitting ability to the training data is continuously enhanced. On the validation set, the performance of the model is also improved, the validation loss fluctuates between 0.013 and 0.126, and the validation accuracy reaches the highest of 91.5% and stabilizes around 0.910. The F1 (Macro) indicator increased from 0.890 to 0.895. In addition, both recall score and precision score improved, from 0.888 to 0.910 and from 0.898 to 0.910 respectively.

Relevant curve showing as Figure 21 below:



a) Train acc = 91.9%, val acc = 91.5%



b) Train loss = 0.211, val loss = 0.071

Figure 21 Accuracy and loss curve under epoch=40 and batch size=70

The confusion matrix illustrates in the Figure 22, and this confusion matrix can show that the ability to process each class is improved than before.

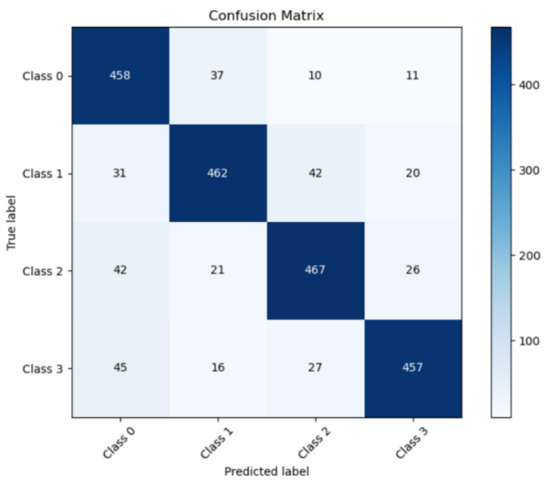
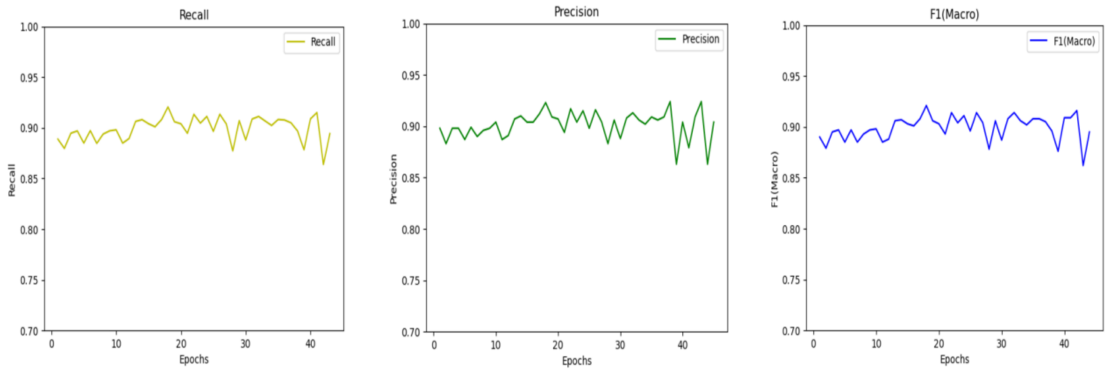


Figure 22 Confusion matrix over epoch=40 and batch size=70

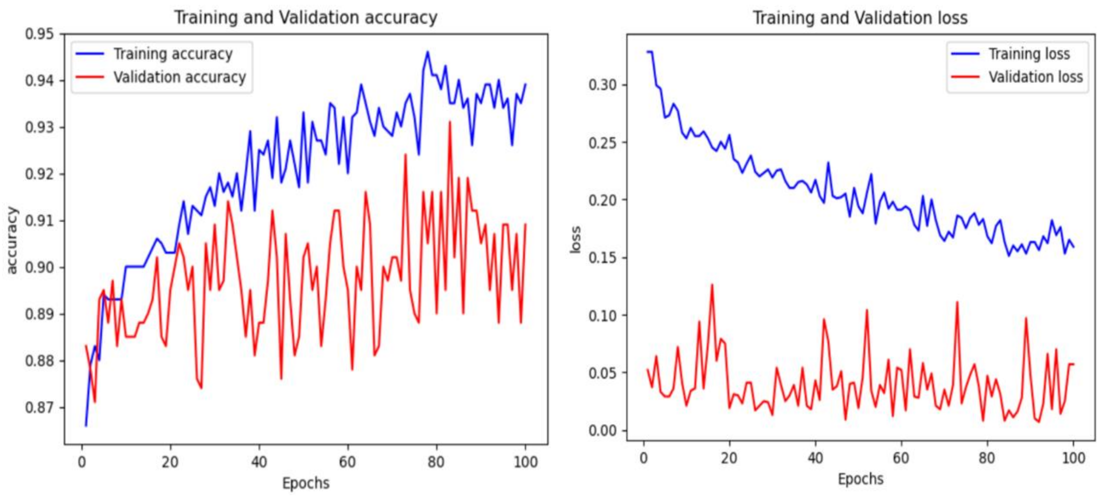
And Figure 23 represents the performance of model via different evaluation metrics Recall, Precision and F1-Score (Macro).



a) Recall = 0.910 b) Precision = 0.904 c) F1-Score = 0.910

Figure 23 Different evaluation metrics for epoch=40 and batch size=70

Continue to increase the epoch to 100 under the batch size=70. The obtained data show that the train accuracy gradually increases and finally reaches 94.6%, but the validation accuracy is unstable and fluctuates around 91.9%. train loss eventually stabilized at around 0.16, F1-score, precision, recall score all stable at about 0.909. The details of these results are shown in Figure 24-25.



a) Train acc = 94.6%, val acc = 91.9%   b) Train loss = 0.159, val loss = 0.057

Figure 24 Accuracy and loss over epoch=100 and batch size=70

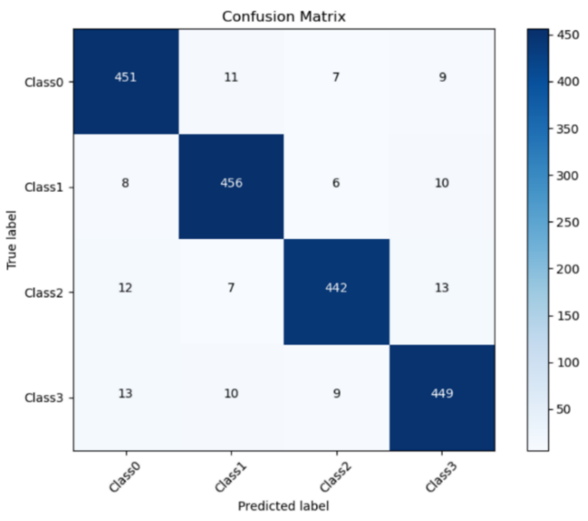
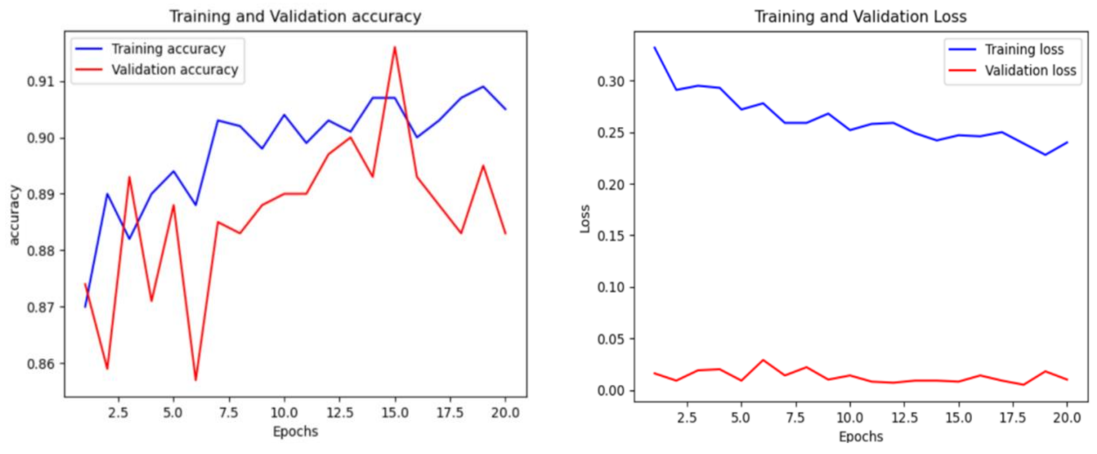


Figure 25 Confusion matrix over epoch=100 and batch size=70

Under such parameter conditions, the above curve and confusion matrix can show that the performance of the model has reached the best situation shown so far, and the recognition ability of OCT for all kinds of retinal diseases is relatively balanced.

When batch size=80 and epoch=20, the performance of the model has not been improved, in which the train accuracy finally reached 91.5%, and the recall score and precision reached about 0.88. This data shows that increasing the batch size does not make the model perform better. The relevant curves are shown in Figure 26 below.



a) Train acc = 90.5%, val acc = 88.3%   b) Train loss = 0.240, val loss = 0.010

Figure 26 Accuracy and loss over epoch=20 and batch size=80

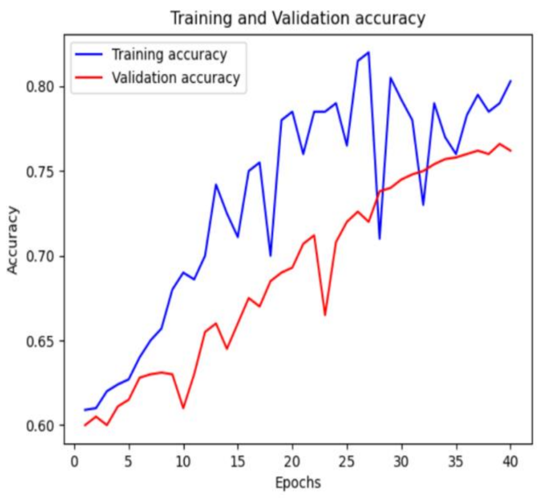
Comparation of results under the different hyper-parameter summarized in Table 13.

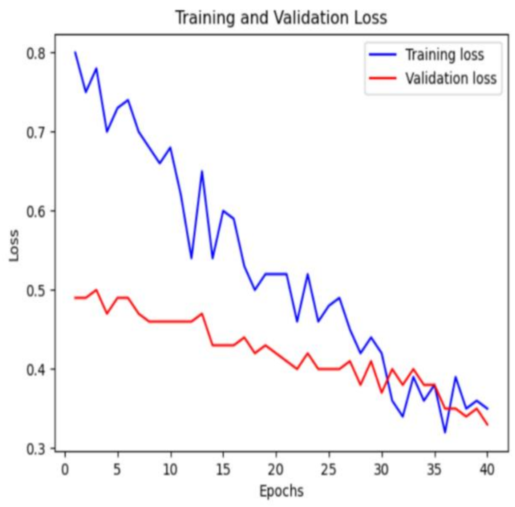
Batch size	Epoch	Train acc	Val acc	Train loss	Val loss	Precision (macro)	Recall (macro)	F1
5	100	89.7%	89.6%	0.474	0.512	-	-	-
	40	85.6%	85.9%	0.422	0.161	-	-	-
10	10	88.4%	90.2%	0.286	0.019	0.902	0.902	0.903
50	40	91.5%	88.8%	0.211	0.007	0.888	0.888	0.887
70	30	91.2%	91.4%	0.227	0.027	0.913	0.913	0.913
	40	91.9%	91.5%	0.211	0.071	0.904	0.910	0.910
	100	94.6%	91.9%	0.159	0.057	0.909	0.909	0.908
80	20	90.5%	88.3%	0.240	0.010	0.885	0.883	0.883

Table 13 Comparison of performance in different parameters

By comparing the model performance under different parameters, when batch size = 70, the model has higher accuracy and lower loss. Meanwhile, all evaluation indicators such as precision, recall score and F-1 score are more than 0.9, indicating good performance. In this case, the accuracy and loss for epoch=40 are slightly better than epoch=30. When epoch reaches 100, considering the comprehensive situation, the performance of the model is relatively the best, the training accuracy is continuously improved, and the loss function curve is also in a downward trend. However, the performance of the model on the validation set shows a downward trend in the last few epochs, which may be due to the overfitting during the training process, so certain this phenomenon from happening again, measures such as early stopping strategy should be used to avoid.

Then more parameter tuning is attempted. The above results are based on the Adam optimizer updating the parameters in the model, the learning rate is 0.0001, now adjust the learning rate to 0.001, the batch size is set to 70, the results are as the following curves in Figure 27.





a) Train acc = 80.0%, val acc = 78.0%   b) Train loss = 0.36, val loss = 0.34

Figure 27 Accuracy and loss curve over learning rate=0.001

The performance of model in different learning rate are summarized as the Table 14.

Learning rate	Train_accuracy	Val_accuracy	Train_loss	Val_loss
0.001	80.0%	78.0%	0.36	0.34
0.0001	91.9%	91.5%	0.211	0.071

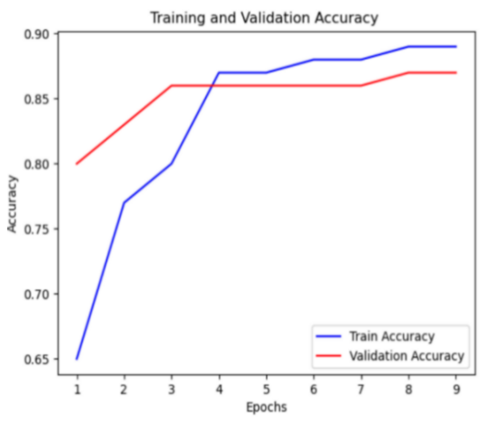
Table 14 Comparison of model performance in different learning rate

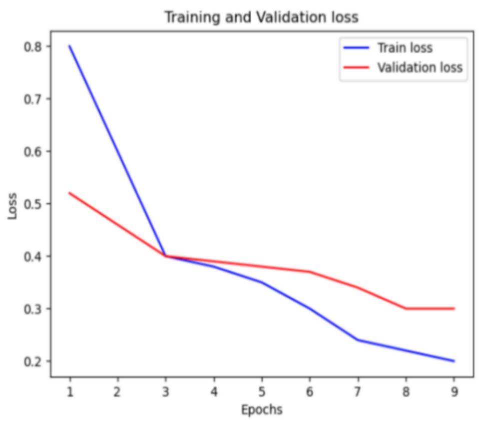
Over the same batch size = 70 and epoch =40, the data above show that the model does not performance better when learning rate = 0.001. On the training set, the accuracy can only reach 0.8 and unstable, and the performance on the validation set is only around 78.0%, and the loss is even worse compared to the previous attempts. This indicates that under the current training setting, a higher learning rate will lead to a decrease in the performance of the model and fail to achieve the desired results.

4.2 Performance of different model

To better evaluate the model performance, this project compares the performance of different models when processing the same dataset.

When using the ResNet to process the dataset, result shows as Figure 28, the train accuracy is close to 89%, validation accuracy is about 87.2% in the end, and both the train and validation loss are higher than the model in this project.

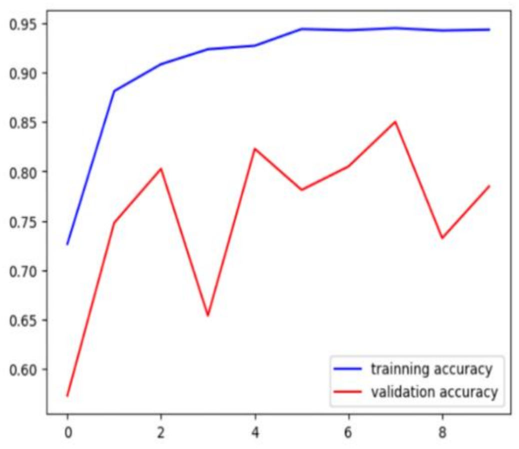


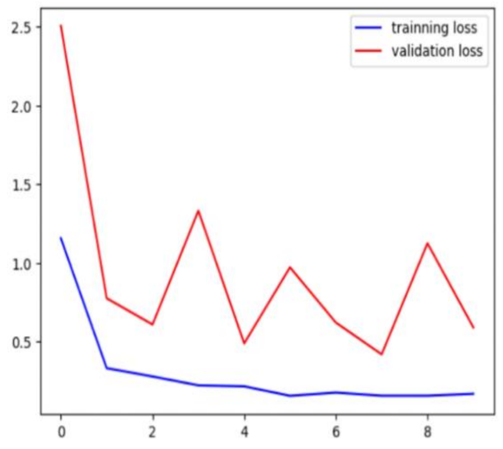


a) Train acc = 89.1%, val acc = 87.2%   b) Train loss = 0.19, val loss = 0.30

Figure 28 Performance of ResNet

For MobileNetV2, the accuracy of the model on the training set reached 95.1%, but the accuracy on the validation set is not too high and unstable, only about 80%, each curve is shown in the Figure 29 below.

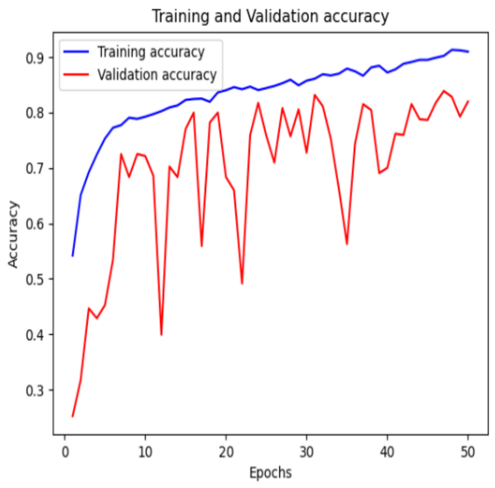


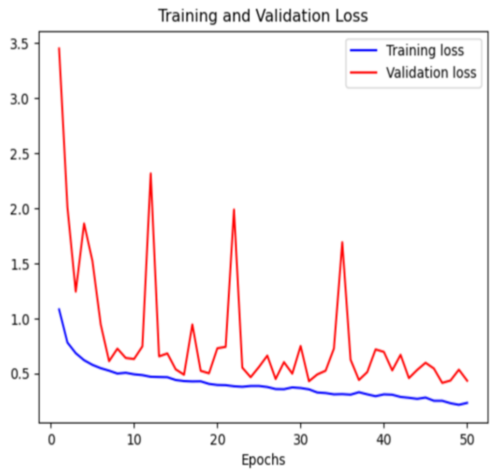


a) Train acc = 95.1%, val acc = 82.3%   b) Train loss = 0.12, val loss = 0.59

Figure 29 Performance of MobileNetV2

With the addition of the batch normalisation layer and Sigmoid activation function on top of the Sequential model, the train accuracy of the dataset finally reached 90.9%. The validation accuracy is about 79.2%. The specific curve is shown in the Figure 30 below.

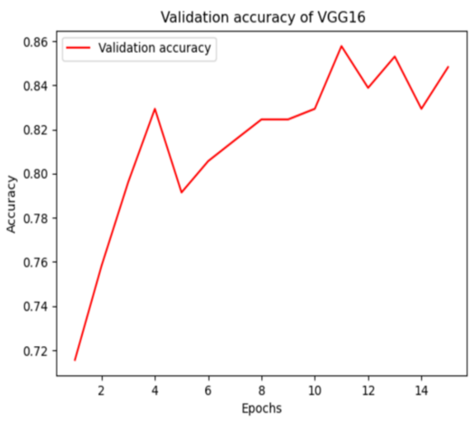


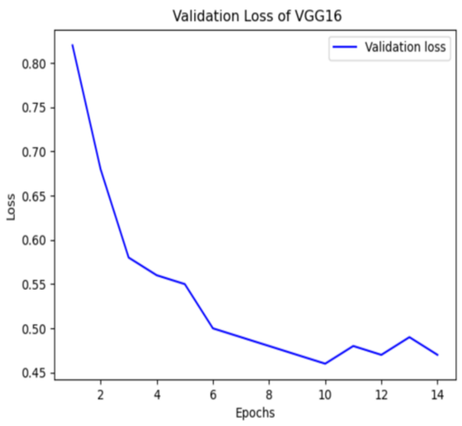


a) Train acc = 90.9%, val acc = 79.2%   b) Train loss = 0.32, val loss = 0.43

Figure 30 Performance of Sequential

After fine-tuning the parameters of the model based on the pre-trained VGG16 network, and use cross-entropy loss function and Adam optimizer to train the model, the validation accuracy could reach 85.2% and the loss is about 0.45. Figure 31 shows the curves of validation loss and accuracy.





a) val acc = 85.2%

b) val loss = 0.45

Figure 31 Performance of VGG16

To summarize these different models, the evaluation metrics are represented in the Table 14, where the Accuracy and Loss means validation accuracy and validation loss.

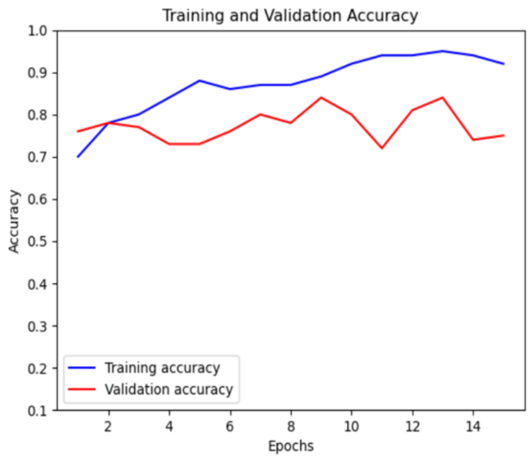
Model	Accuracy	Loss
ResNet	87.2%	0.30
MobileNetV2	82.3%	0.59
Sequential	79.2%	0.43
VGG16	85.2%	0.45
Proposed model	91.9%	0.057

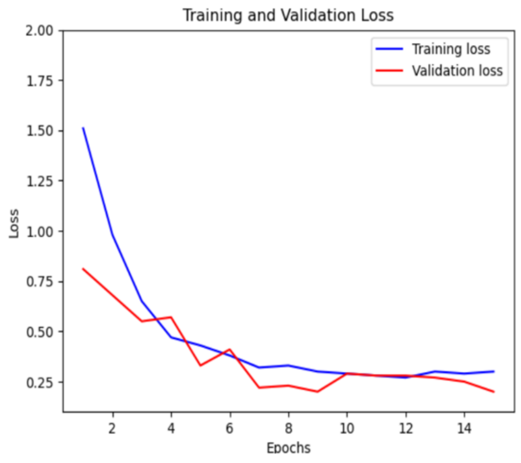
Table 15 Performance comparison of different models

Through the evaluation of the performance of different models under the same data set, the performance of the model proposed in this project in the classification of retinal diseases can be more intuitively seen. First of all, the train loss and validation loss of the model proposed in this project are smaller than other models. In addition, although the train accuracy of 94.6% is slightly lower than the 95.1% of MobileNetV2, the accuracy of the model on the verification set is significantly higher than that of MobileNetV2.

Comparing the proposed model with models proposed by other researchers mentioned in the chapter 2 using the same dataset, the relevant results are shown below.

Najeeb et al. [10] used the Single layer CNN structure to build the network model. When the data set used in this project was used to train the model, the accuracy was about 80.5% as Figure 32, which is lower than the model proposed in this project. This may be due to the lightweight network structure which weakens the performance of the model.

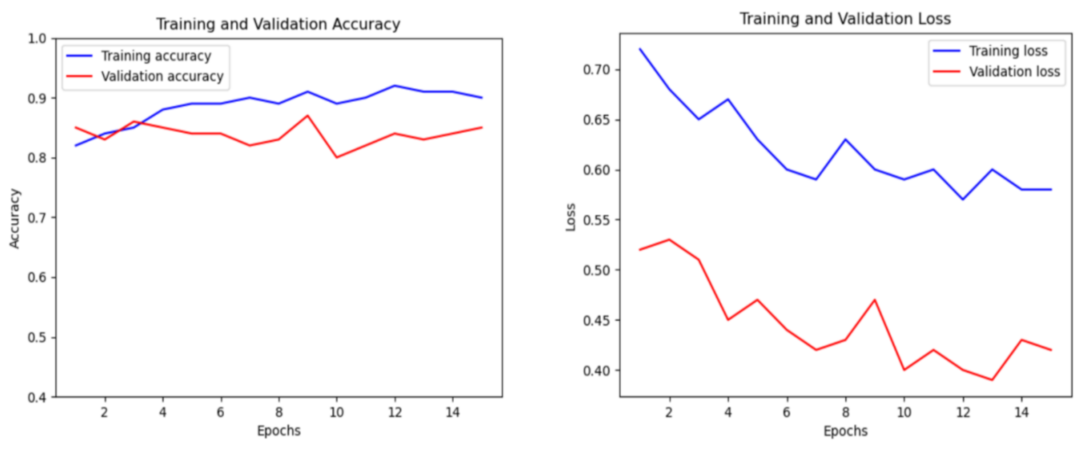




a) Train acc = 92.1%, val acc = 80.5% b) Train loss = 0.32, val loss = 0.26

Figure 32 Performance of the model proposed by Najeeb et al.

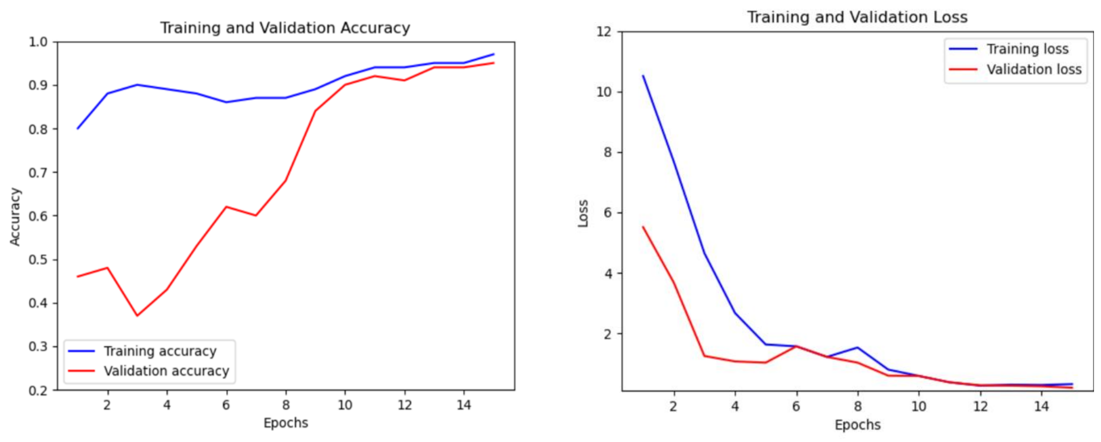
The model proposed by Bhadra and Kar [11] mainly applies deep multi-layered CNN. When this model deals with the same dataset used this project, the results are not as it shows in the original paper. The accuracy on the validation set only reaches about 85% as Figure 33, which may be caused by the hardware or parameter settings.



a) Train acc = 91.3%, val acc = 85.2%   b) Train loss = 0.61, val loss = 0.42

Figure 33 Performance of the model proposed by Bhadra and Kar

The model proposed by Intaraprasit [12] mainly applies MobileNetV2 and makes adjustments. The results show that the verification accuracy of this model on the data set reaches 95% as Figure 34, which is higher than the model proposed in this project, but the loss is lower.



a) Train acc = 95.6%, val acc = 95.0%   b) Train loss = 0.19, val loss = 0.21

Figure 34 Performance of the model proposed by Intaraprasit et al.

Table 16 summarizes the performance of the above models

Researchers	Techniques	Accuracy	Loss
Najeeb et al. [10]	Single layer CNN structure.	80.5%	0.26
Bhadra and Kar [11]	Deep multi-layered CNN	85.2%	0.42
Intaraprasit et al. [12]	MobileNetV2	95.0%	0.21
Proposed model in project	Inception, Separable CNN	91.9%	0.067

Table 16 Comparison of models proposed by different researchers with this project

Through comparison, it shows that except the model proposed by Intaraprasit is greater than the model proposed by this project in terms of accuracy, the performance of this model is higher than models, which indicates that the model still has room for improvement. This may be due to the fact that different models need different parameters to achieve the best performance, and the relevant parameters are not adjusted during the comparison and training process. However, it can also show that the performance of the model is relatively high under the same parameters and environment configuration.

4.3 Model Explainability

A normal retinal OCT image is selected, and the results predicted by the model are displayed as text along with the input image, as shown in Figure 35.

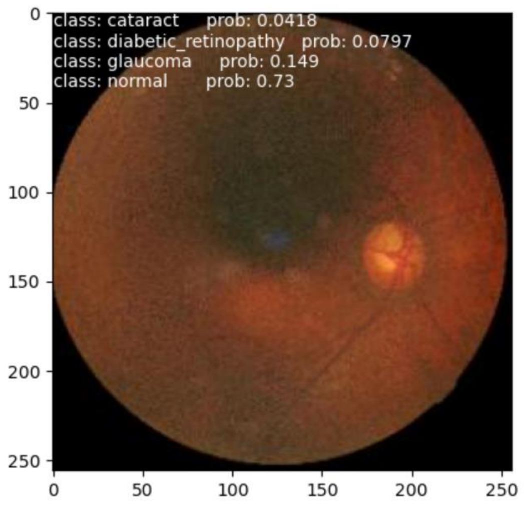

class: cataract prob: 0.0418
class: diabetic_retinopathy prob: 0.0797
class: glaucoma prob: 0.149
class: normal prob: 0.73

Figure 35 Prediction of proposed model

To improve the interpretability of the Model, this project makes use of LIME (Local Interpretable Model-agnostic Explanations) technology. By emphasizing the areas in the original image that contribute most to the classification result, the LIME interpreted image as shown in Figure 36, illustrates the interpretation of the original image classification findings, it can analyse how much each feature contributes to the final result.

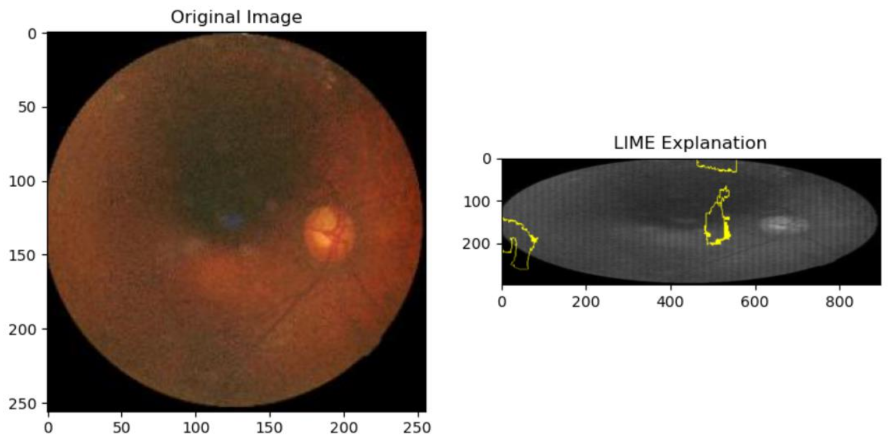
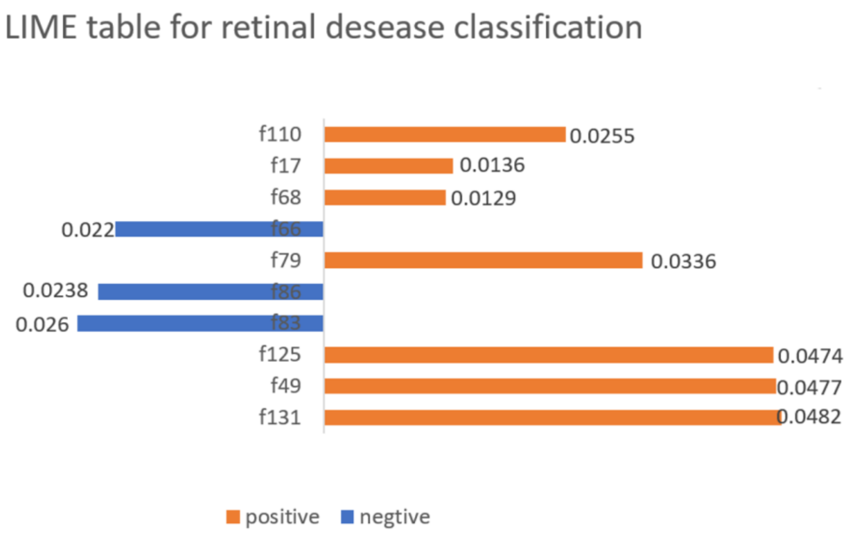
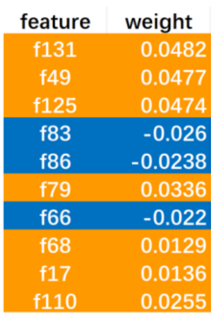


Figure 36 LIME explanation





feature	weight
f131	0.0482
f49	0.0477
f125	0.0474
f83	-0.026
f86	-0.0238
f79	0.0336
f66	-0.022
f68	0.0129
f17	0.0136
f110	0.0255

Figure 37 LIME table for retinal disease classification

Figure 37 represents the interpretation of the predictions of the proposed model for a particular input. Where Feature represents a feature or region in the image, and Weight represents the contribution of the model to the prediction of a particular class based on this feature. Blue indicates that the feature contributes negatively to the model's classification, and orange indicates positively.

4.4 Development of GUI

This section records the implementation and development of graphical user interface for this project, which is a web interface, and also introduces the usage of this web application.

This project develops a small web application using the Flask framework and deploys the trained retinal disease classification model into this application for a more concise and efficient user experience.

As illustrate in the Figure 38 below, this project realizes a simple home page through HTML and CSS technology, HTML is responsible for the page structure and content presentation, while CSS is responsible for the page style and layout, including the setting of the background image, font style, button style, etc. In this page, and there is a "learn more" button on this page, when pressed, it will jump to the second page, the working principle of the button relies on the mechanism of the hyperlink element in HTML, the user clicks on the button to trigger the click event of the hyperlink, the browser will load the corresponding resources according to the target address of the link which is the "Learn More" page.

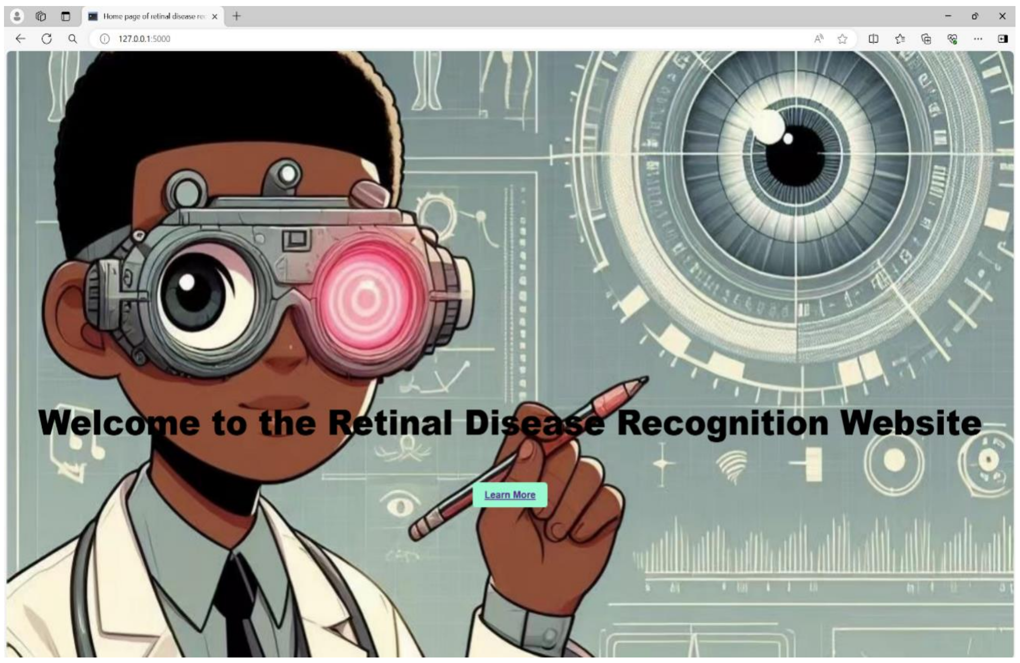

Home page of retinal disease re
127.0.0.1:5000
Welcome to the Retinal Disease Recognition Website
Learn More

Figure 38 Home page of the web application

Figure 39 shows the Learn More page for web application, this page also utilizes HTML, CSS to achieve image display and simple interactive functions. OCT images of three common retinal diseases as well as normal retina are shown here, and when the mouse is moved to the position of the image, the profile corresponding to the image is displayed. And when click the “Start button”, it will turn to the detect page.

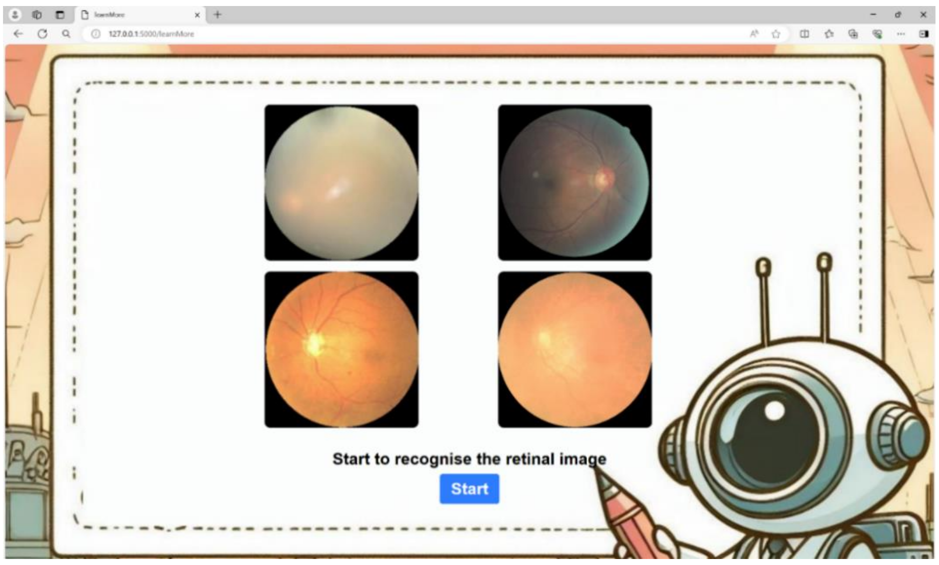


Figure 39 Learn More page of web application

The mouse hover effect in this page is implemented using CSS. Each image item contains an overlay, which by default has a transparency of 0, i.e. invisible. The basic properties of the overlay are set through CSS styles, and a transition effect is used to achieve a smooth display.

When the mouse hovers over an image item, it triggers the display effect of the overlay, and the transparency changes from 0 to 1, so that the title and description information inside the overlay is gradually displayed. After the mouse is removed, the transparency transitions back to 0 again and the overlay is hidden, restoring the default state. Figure 40 can represent this function.

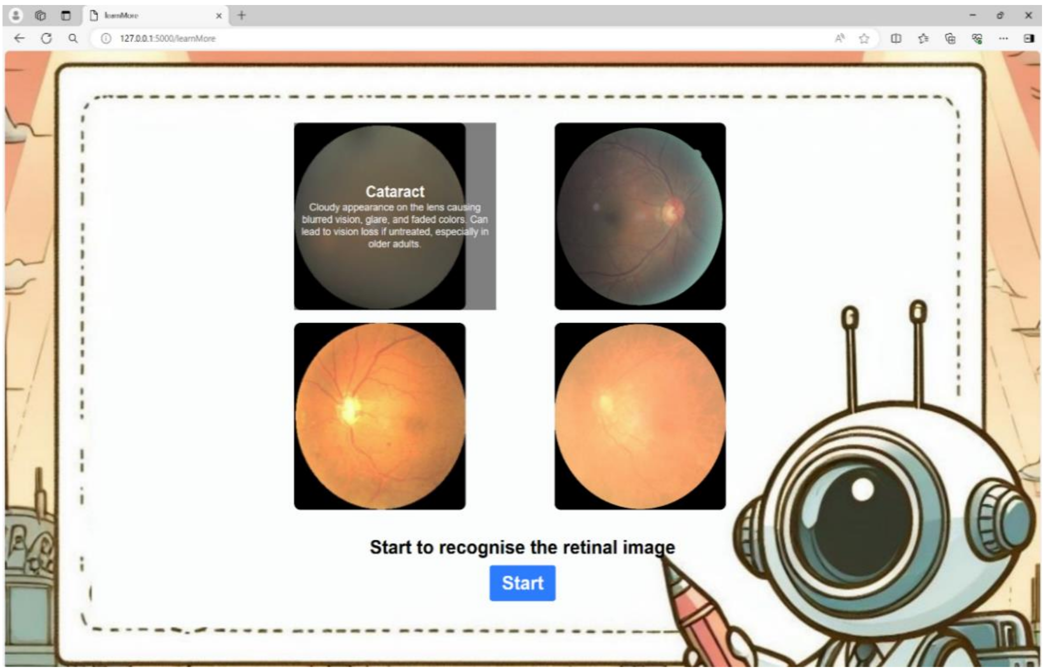

Cataract
Cloudy appearance on the lens causing blurred vision, glare, and faded colors. Can lead to vision loss if untreated, especially in older adults.
Start to recognise the retinal image
Start

Figure 40 Mouse hover effect of Learn More page

The detect page as Figure 41 contains functions choose image, upload image, delete image and predict. Users can upload retinal images, and the back-end makes classification predictions based on a pre-trained model and returns the results to the front-end page to display to the user.

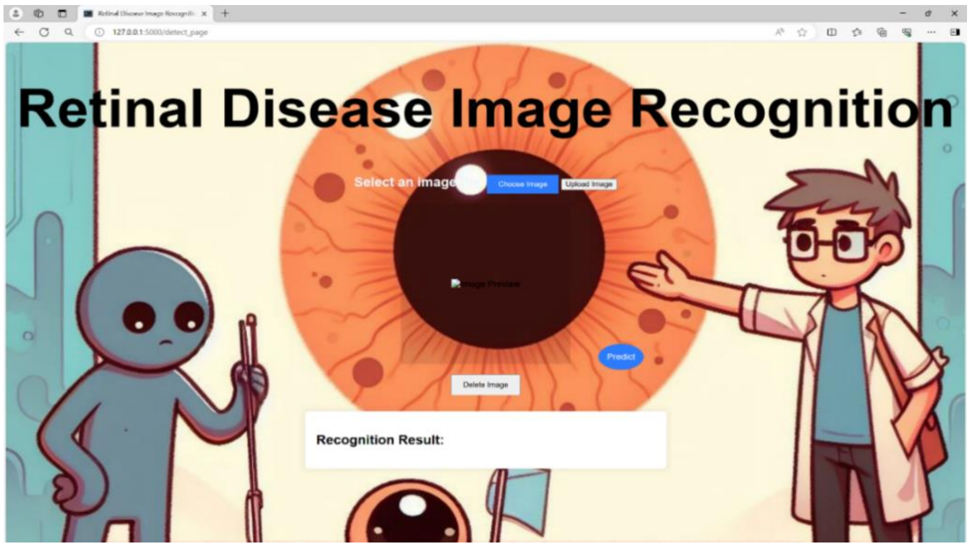


Figure 41 Detect page of web application

The "Upload image" method relies heavily on front-end JavaScript technologies. When the user selects an image file, the method first gets the file selected by the user and creates a new image object to load the file. In the callback function after the image is loaded, the uploaded image is displayed on the page with a message that the upload is successful as Figure 42.

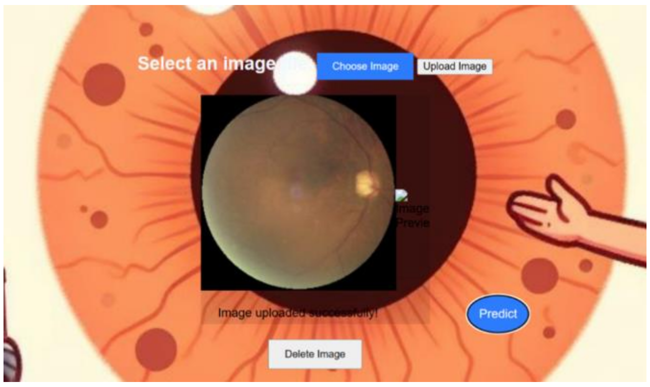
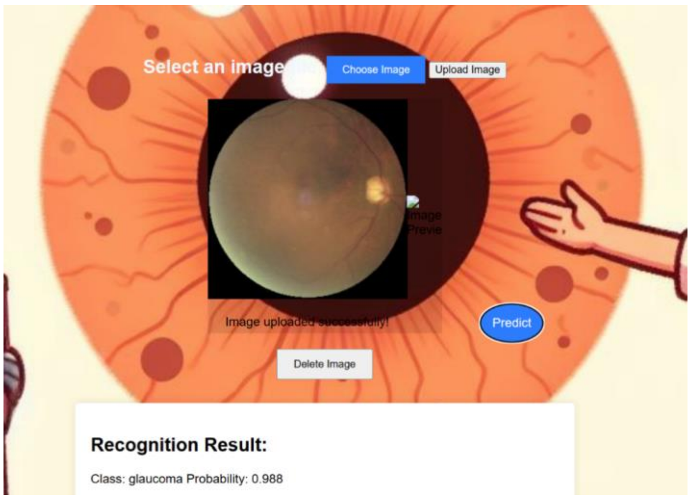


Figure 42 Choose and upload functions in detect page

Figure 43 demonstrates the predict functionality of this web application, which is implemented first through front-end JavaScript in collaboration with the back-end Flask framework. A POST request is sent to the '/predict' route in the back-end via the Fetch API, which sends a FormData object to the back-end for image prediction. Finally, the prediction results returned by the back-end are received and parsed, and displayed to the user on the page.



Recognition Result:

Class: glaucoma Probability: 0.988

Figure 43 The demonstration of the predict function in detect page

Chapter 5 Professional Issues

5.1 Project management

5.1.1 Activities

The activities in this project are presented in the Table 15.

Objectives	Activities	State
Stage 1: Preparation for project	Learn from relevant literature.	Completed
	Compare the previous model from relevant research.	Completed
	Prepare the datasets and the development environment.	Completed
	Learn about CNN.	Completed
Stage 2: Model design and data processing	Decide how to combine separate CNN with inception model to build the model.	Completed
	Create the architecture model.	Completed
Stage 3: Training and optimizing	Process the data to divide them for training, validation, and testing datasets.	Completed
	Train the model by training set.	Completed
	Evaluate the performance of model through the validation set, further optimize the model.	Completed
	Evaluate the performance of model through testing set.	Completed
	Adjust parameters for more training.	Completed
Stage 4: Summarize and	Evaluate the model using more evaluation strategies	Completed

Graphical User Interface design	Design and implement the Graphical User Interface.	Completed
	Summarize the whole data in the project and finish the final report.	Completed
	Summarize the project and prepare for the presentation.	Completed

Table 17 The complete/uncompleted tasks for each objective

5.1.2 Schedule

The schedule of the project represented with Gantt is shown in Figure 44.

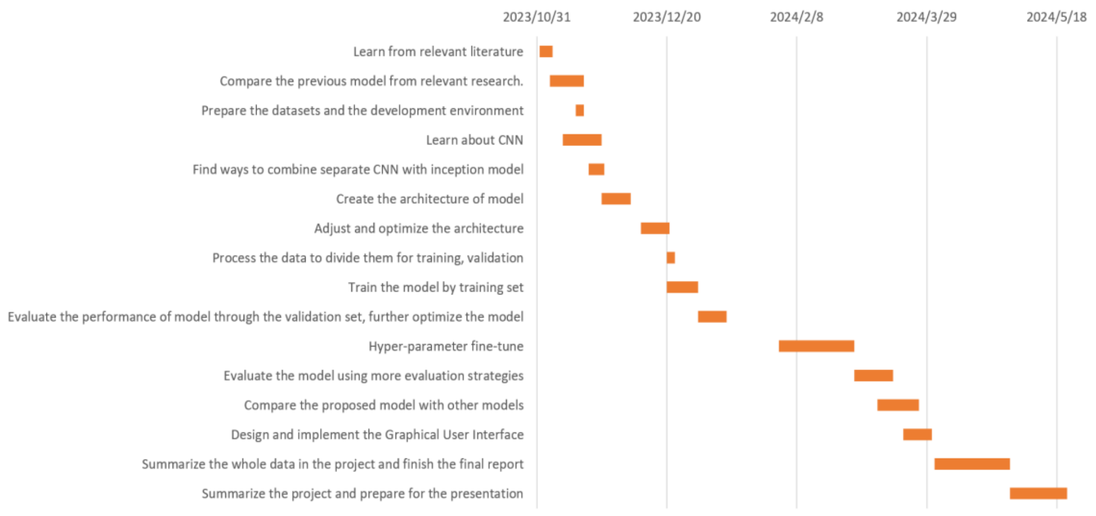


Figure 44 Gantt for the project schedule

5.1.3 Project Data Management

a. The deliverables are stored in the local file which presented as Figure 45.

b. The copy of deliverables is uploaded to the GitHub Figure 46 with link https://github.com/Kkkkyf7/Project.

c. Zotero is used to manage the reference in this project.

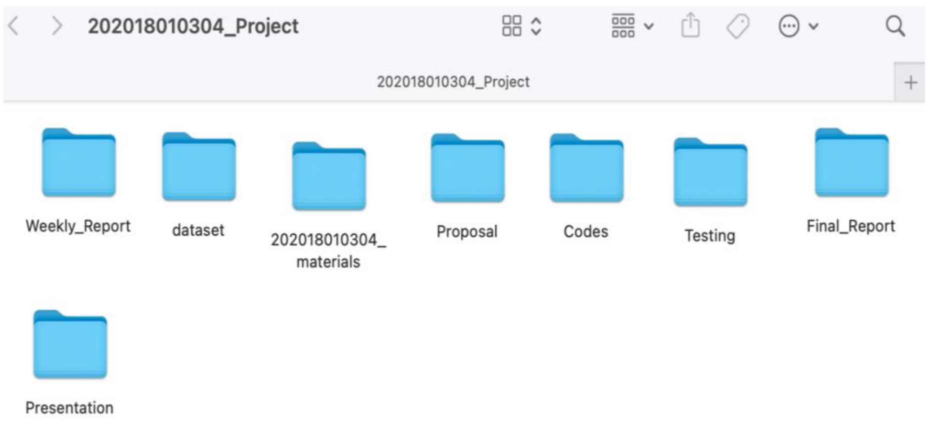
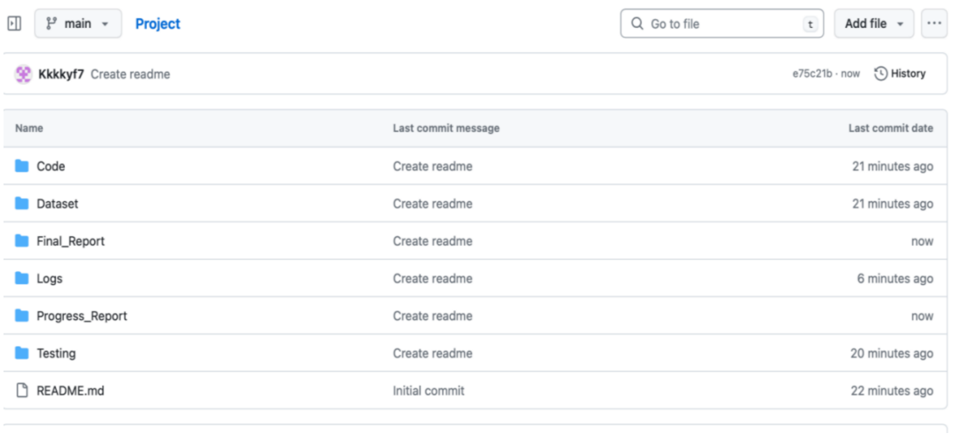


Figure 45 Data management plan in local



Name	Last commit message	Last commit date
Code	Create readme	21 minutes ago
Dataset	Create readme	21 minutes ago
Final_Report	Create readme	now
Logs	Create readme	6 minutes ago
Progress_Report	Create readme	now
Testing	Create readme	20 minutes ago
README.md	Initial commit	22 minutes ago

Figure 46 Data management plan in GitHub

5.1.4 Project Deliverable

The deliverable in this project contains:

a. Project proposal

b. Ethics review form

c. Weekly report

d. Progress report

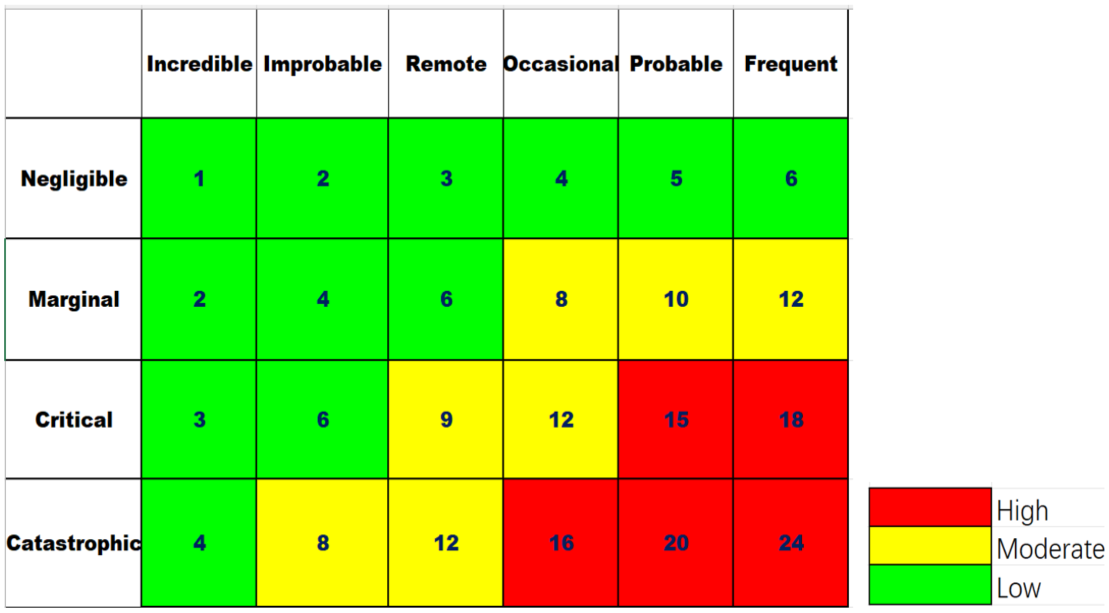
e. Final report

f. Project code

g. Presentation files

5.2 Risk Analysis

Risk analysis is based on the process of identifying, assessing and dealing with the potential risks that may occur project, and it is an indispensable part of project management. The risk levels applied in this project and the representation of their severity is shown in Figure 47 below.



	Incredible	Improbable	Remote	Occasional	Probable	Frequent
Negligible	1	2	3	4	5	6
Marginal	2	4	6	8	10	12
Critical	3	6	9	12	15	18
Catastrophic	4	8	12	16	20	24

🟥	High
🟨	Moderate
🟩	Low

Figure 47 Risk analysis evaluation matrix

The risk analysis of this project is shown as the Table 16 below.

Potential Risks	Potential Causes	Severity	Likelihood	Risk	Mitigation
Inability to complete model training within the specified	Hardware failure or insufficient resources during training.	4	3	12	Hardware and resources assessment in advance to ensure adequate computing resources.
	Model is overly	3	2	6	Optimising the model

time.	complex.				structure.
	Poor quality of training data.	4	4	16	Data pre-processing and data enhancement techniques.
Improper application of separable convolution technique leads to degradation of model performance.	Inappropriate choice of separable convolutional parameters.	4	5	20	Careful tuning of separable convolution parameters.
	Too many separable convolutional layers lead to increased computational complexity of the model.	3	4	12	Optimising network structure to balance model performance and computational complexity.
	Too many repetitive information in the training data.	2	3	6	Data de-duplication and sampling.
Model performance is unstable in some categories.	Unbalanced sample sizes for some categories in the data.	3	2	6	Dealing with sample imbalance using weight balancing strategies or oversampling/ under sampling etc.

	Unclear labelling of categories makes it difficult to identify criteria for correct classification of models.	5	2	10	Revisit and clarify category labelling to ensure clarity of labels.
Overfitting of the model.	There are outliers in the training data.	3	4	12	Outlier detection and handling to ensure data quality.
	Poor feature selection and over-focusing of the model on noisy features.	4	4	16	Perform profiling and selection to exclude irrelevant features.
Model deployment issues in GUI.	The environment configuration prevented the model from being deployed.	2	1	2	Refine the environment configuration as needed.
Future Risks.	Retinal image information may be leaked after deployment.	5	3	15	Encrypt the information of patients.

Table 18 Risk analysis

5.3 Professional Issues

In the field of medical diagnosis, the emergence of machine learning and deep learning models can help the diagnosis process be more efficient, and increase the accuracy of the diagnosis results. However, with great power comes great responsibility, especially in critical application areas such as classification of retinal diseases. This section of the project delved into the intricate interplay between technical, legal, social, ethical, and environmental factors, it highlights the development and deployment of such models, and the issues that needs attention in the future.

In this project, the false judgement of the classification model may lead to the wrong diagnosis of retinal diseases, and if the output of the model affects the final decision of the doctor, it will affect the patient's treatment direction and optimal treatment time, which may eventually lead to treatment failure. Therefore, it is necessary to improve the accuracy of the model, the accuracy of the model for each type of retinal disease must be controlled in a high range. And after the model is automatically classified, it should also be confirmed by professional medical personnel, in order to avoid misdiagnosing.

In terms of legal aspects, if there are medical omissions or wrong treatments which caused by inaccurate model output, it may lead to disputes between doctors and patients, and it even makes the relevant parties bear the legal liability. Therefore, the accuracy of the model should be fully guaranteed, to support the reliability and effectiveness of the relevant diagnostic decisions.

There are also certain social and ethical issues in the process of model deployment and assistance in diagnosis, and the performance of retinal disease recognition models may vary with different patient groups. To ensure the accuracy of the model in different populations and avoid misdiagnosis due to race, gender, or social status, data sampling and model evaluation of different groups should be fully considered. In addition, in the process of using medical data, it is necessary to strictly abide by the relevant regulations of patient privacy protection to ensure that the privacy rights of patients will not be violated during data training and sharing, so as to ensure the legitimate use of medical data.

In terms of the environment, since deep learning model training requires a large amount of computational resource, it is necessary to seek for more efficient computational methods to reduce the consumption and burden on the environment, such as using more energy-efficient hardware and optimising training algorithms. Also, the deployment of retinal disease recognition models may require embedded systems or dedicated hardware. Environmentally friendly e-waste management measures need to be taken when equipment is renewed or maintained to ensure proper disposal of old equipment.

In summary, the development and deployment of retinal disease classification models also represents the convergence of technological innovation, legal responsibility, social responsibility, ethical review, and environmental management. Considering multiple factors, predicting and solving problems in a timely manner is key to the project, which can maximize the potential of machine learning in healthcare. In the process of applying machine learning, privacy, security and sustainability of the technology need to be ensured at the same time in order to more effectively use this technology to bring continuous development and progress to the healthcare industry.

Chapter 6 Conclusion

In conclusion, this project completed a classification model with high performance that can identify OCT images of retinal diseases by combining Inception and separable convolutional neural network architectures. And through iterative hyperparameter tuning and model optimization, the model structure and performance is gradually improved. By automating the identification of retinal diseases, the model proposed in this project significantly reduces the reliance on manual identification methods, thus saving the time and resources of medical professionals.

The main result of the project is to create a deep learning model with a training accuracy of 94.6%, validation accuracy of 91.9%, and validation loss about 0.057. The macro-average precision, recall and F1-Score of the model are close to 0.91, which shows excellent performance in various evaluation metrics. Through comparative analysis with other classic models such as ResNet and MobileNetV2, as well as models proposed by other researchers in the retinal disease classification field on the same data set, the superiority of the proposed model in classification accuracy is further verified. In addition, this project also includes a graphical user interface, which further enhances user operability and makes it more convenient for medical staff or patients to use the model for actual diagnostic analysis.

The project also has limitations. Since the dataset only contains several thousand images, in the actual application process, when encountering new data, or the images under different environments, devices, lighting conditions, the performance of the model may decline. In addition, the medical field is a rigorous field, in order to avoid misdiagnosis, the accuracy of the model is not enough so it needs to continue to improve. And medical image data involves patient privacy, and there is no relevant protection procedure, so relevant protection measures need to be introduced to this project.

In the future work, more adjustments and optimizations are still needed to improve the project capability. In terms of the dataset selection, more diverse and representative images can be used to expand the dataset, to increase the number of retinal disease classes that can be recognised of the project, and improve the generalization ability and performance of the model on real-world. In addition, new machine learning techniques can be added, or more adjustment can be made in the parameters to further optimize and improve the classification accuracy. Furthermore, the project can also be integrated with existing healthcare systems by combining real-time data, which can enable the model to be truly deployed to participate in the medical field, collaborate with domain experts and conduct clinical trials to verify the effectiveness of the model in real-world scenarios.

Reference

[1] M. Edward Wilson, Jr., Rupal H. Trivedi, Suresh K. Pandey (2005). Pediatric cataract surgery techniques, complications, and management. Philadelphia: Lippincott Williams & Wilkins. p. 20. ISBN 978-0-7817-4307-5. Archived from the original on 2015-05-24.

[2] Vass C, Hirn C, Sycha T, Findl O, Bauer P, Schmetterer L (October 2007). "Medical interventions for primary open angle glaucoma and ocular hypertension". The Cochrane Database of Systematic Reviews. 2007 (4): CD003167. doi:10.1002/14651858.CD003167.pub3. PMC 6768994. PMID 1794 3780.

[3] Wong, T., Cheung, C., Larsen, M. et al. Diabetic retinopathy. Nat Rev Dis Primers 2, 16012 (2016). https://doi.org/10.1038/nrdp.2016.12

[4] P. M. Arabi, N. Krishna, N. V. Deepa, V. Ashwini and H. M. Prathibha, "A comparision of OCT and retinal fundus images for age-related Macular degeneration," *2017 8th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, Delhi, India, 2017, pp. 1-5, doi: 10.1109/ICCCNT.2017.8204107.

[5] S. Meenu Mohan and S. Aji, "A Fast Method for Retinal Disease Classification from OCT Images Using Depthwise Separable Convolution," in R. Raje, F. Hussain, and R. J. Kannan (eds.), Artificial Intelligence and Technologies, Lecture Notes in Electrical Engineering, vol. 806, Springer, Singapore, 2022, pp. 223-232, doi: 10.1007/978-981-16-6448-9_18.

[6] D. S. Kermany et al., "Identifying medical diagnosis and treatable diseases by image based deep learning," in Cell, vol. 172, no. 5, pp. 1122-1131, 2018.

[7] A. Alqudah, "A OCT-NET a convolutional network automated classification of multiclass retinal disease classification using SD-OCT images," in International federation for medical and biological engineering, 2019.

[8] N. Rajagopalan, V. Narasimhan, S. Kunnavakkam Vinjimoor et al., "RETRACTED ARTICLE: Deep CNN framework for retinal disease diagnosis using optical coherence tomography images," in J Ambient Intell Human Comput, vol. 12, pp. 7569-7580, 2021, doi: 10.1007/s12652-020-02460-7.

[9] G. Mahendran, M. Periyasamy, S. Murugeswari, and N. K. Devi, "Analysis on retinal diseases using machine learning algorithms," in Mater. Today, Proc., vol. 33, pp. 3102-3107, Jan. 2020.

[10] S. Najeeb, N. Sharmile, M. S. Khan, I. Sahin, M. T. Islam and M. I. Hassan Bhuiyan, "Classification of Retinal Diseases from OCT scans using Convolutional Neural Networks," *2018 10th International Conference on Electrical and Computer Engineering (ICECE)*, Dhaka, Bangladesh, 2018, pp. 465-468, doi: 10.1109/ICECE.2018.8636699.

[11] R. Bhadra and S. Kar, "Retinal Disease Classification from Optical Coherence Tomographical Scans using Multilayered Convolution Neural Network," *2020 IEEE Applied Signal Processing Conference (ASPCON)*, Kolkata, India, 2020, pp. 212-216, doi: 10.1109/ASPCON49795.2020.9276708.

[12] P. Intaraprasit, T. H. Bui and M. Phu Paing, "MobileNetV2-based Deep Learning for Retinal Disease Classification on a Mobile Application," *2023 15th Biomedical Engineering International Conference (BMEiCON)*, Tokyo, Japan, 2023, pp. 1-5, doi: 10.1109/BMEiCON60347.2023.10322079.

[13] X. -F. Xu, L. Zhang, C. -D. Duan and Y. Lu, "Research on Inception Module Incorporated Siamese Convolutional Neural Networks to Realize Face Recognition," in *IEEE Access*, vol. 8, pp. 12168-12178, 2020, doi: 10.1109/ACCESS.2019.2963211.

[14] F. Chollet, "Xception: Deep Learning with Depthwise Separable Convolutions," in 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 2017, pp. 1800-1807, doi: 10.1109/CVPR.2017.195.

[15] C. Szegedy et al., "Going deeper with convolutions," in 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 2015, pp. 1-9, doi: 10.1109/CVPR.2015.7298594.

[16]Xiaoling Xia, Cui Xu, and Bing Nan, ‘Inception-v3 for flower classification’, in 2017G. Mahendran, M. Periyasamy, S. Murugeswari, and N. K. Devi, "Analysis on retinal diseases using machine learning algorithms," in Mater. Today, Proc., vol. 33, pp. 3102-3107, Jan. 2020.

[17] Tang (May 2018). Intelligent Mobile Projects with TensorFlow. Packt Publishing. pp. Chapter 2. ISBN 9781788834544.

[18] F. Chollet, "Xception: Deep Learning with Depthwise Separable Convolutions," in 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 2017, pp. 1800-1807, doi: 10.1109/CVPR.2017.195.

[19] Venkatesan, Ragav; Li, Baoxin (2017-10-23). Convolutional Neural Networks in Visual Computing: A Concise Guide. CRC Press. ISBN 978-1-351-65032-8. Archived from the original on 2023-10-16. Retrieved 2020-12-13.

[20] Balas, Valentina E.; Kumar, Raghvendra; Srivastava, Rajshree (2019-11-19). Recent Trends and Advances in Artificial Intelligence and Internet of Things. Springer Nature. ISBN 978-3-030-32644-9. Archived from the original on 2023-10-16. Retrieved 2020-12-13.

[21] L. Wanchen, "Analysis on the Weight initialization Problem in Fully-connected Multi-layer Perceptron Neural Network," 2020 International Conference on Artificial Intelligence and Computer Engineering (ICAICE), Beijing, China, 2020, pp. 150-153, doi: 10.1109/ICAICE51518.2020.00035.

[22] K. J, P. Dharanyadevi and G. Zayaraz, "Handwritten Digit Recognition Using CNN with Average Pooling and Global Average Pooling," *2023 6th International Conference on Contemporary Computing and Informatics (IC3I)*, Gautam Buddha Nagar, India, 2023, pp. 599-603, doi: 10.1109/IC3I59117.2023.10398014.

[23] Brownlee, Jason (8 January 2019). "A Gentle Introduction to the Rectified Linear Unit (ReLU)". Machine Learning Mastery. Retrieved 8 April 2021.

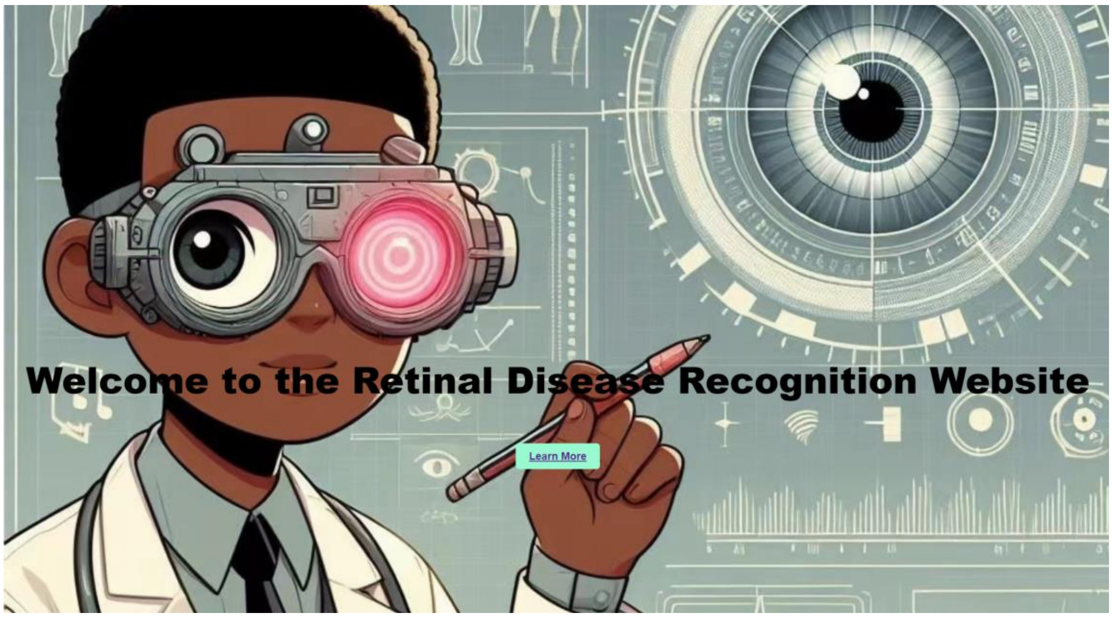
[24] Xavier Glorot; Antoine Bordes; Yoshua Bengio (2011). Deep sparse rectifier neural networks (PDF). AISTATS. Rectifier and softplus activation functions. The second one is a smooth version of the first.

[25] Liu, Danqing (30 November 2017). "A Practical Guide to ReLU". Medium. Retrieved 8 April 2021.
[26] S. Mehta, C. Paunwala and B. Vaidya, "CNN based Traffic Sign Classification using Adam Optimizer," 2019 International Conference on Intelligent Computing and Control Systems (ICCS), Madurai, India, 2019, pp. 1293-1298, doi: 10.1109/ICCS45141.2019.9065537.
[27] Anqi Mao, Mehryar Mohri, Yutao Zhong. Cross-entropy loss functions: Theoretical analysis and applications. ICML 2023. https://arxiv.org/pdf/2304.07288.pdf
[28] Hinton, G.E., Srivastava, N., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2012). Improving neural networks by preventing co-adaptation of feature detectors. ArXiv, abs/1207.0580.
[29] Dempster, A.P.; Laird, N.M.; Rubin, D.B. (1977). "Maximum Likelihood from Incomplete Data Via the EM Algorithm". Journal of the Royal Statistical Society: Series [26] B (Methodological). 39 (1): 1–22. doi:10.1111/j.2517-6161.1977.tb01600.x.
[30] Shorten, Connor; Khoshgoftaar, Taghi M. (2019). "A survey on Image Data Augmentation for Deep Learning". Mathematics and Computers in Simulation. 6. springer: 60. doi:10.1186/s40537-019-0197-0.
[31] L. Qi, H. Yang, Y. Shi and X. Geng, "NormAUG: Normalization-Guided Augmentation for Domain Generalization," in IEEE Transactions on Image Processing, vol. 33, pp. 1419-1431, 2024, doi: 10.1109/TIP.2024.3364516

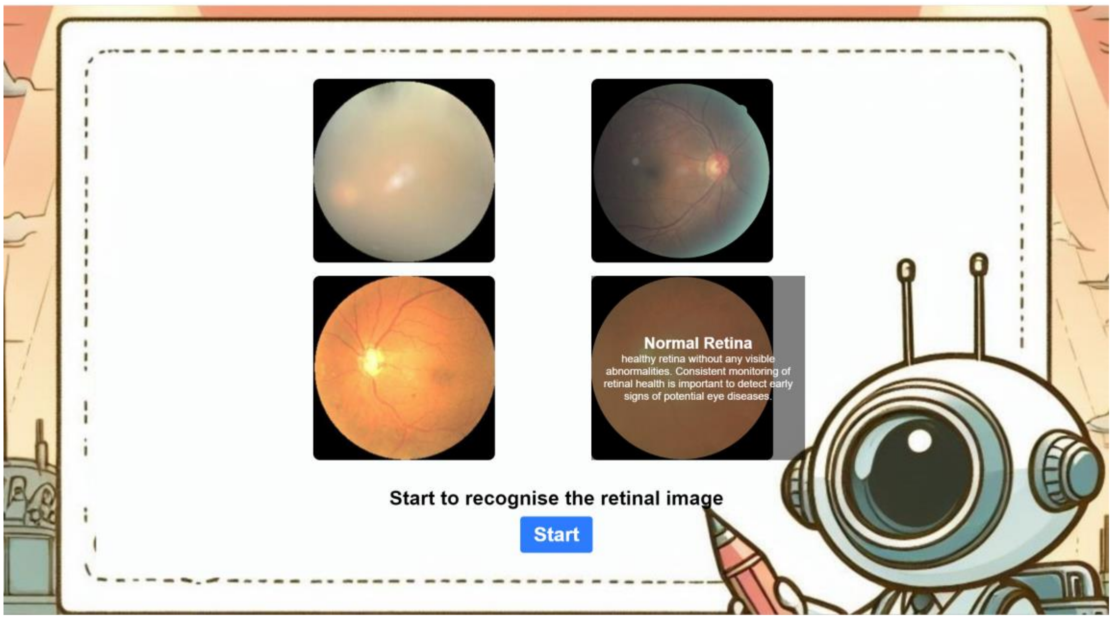
Appendices

Repository URL of GitHub: https://github.com/Kkkkyf7/Project

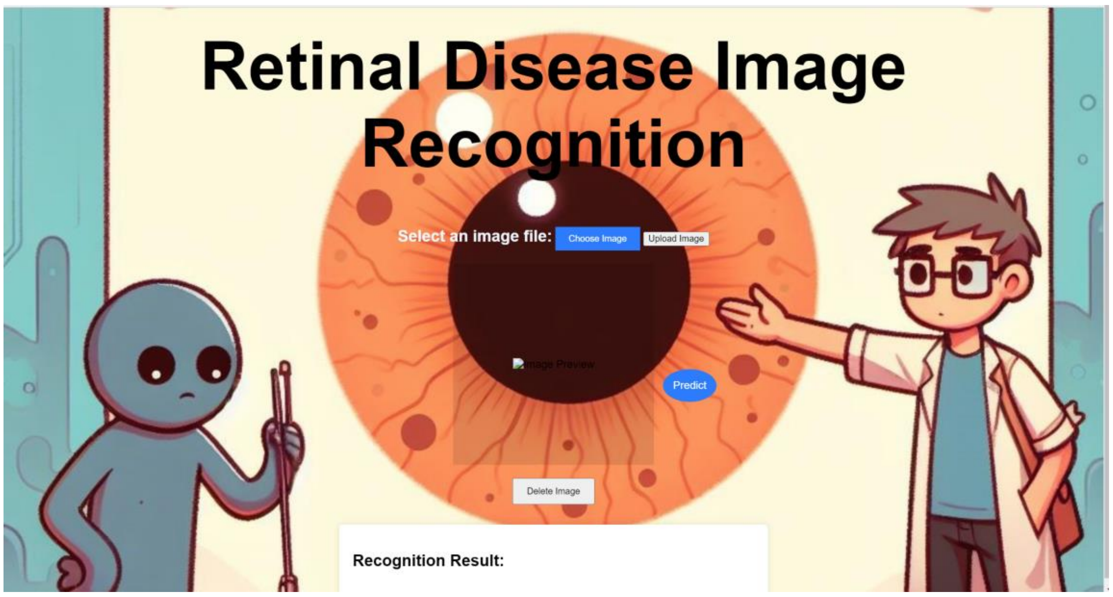
The demonstration of complete web pages:



Appendices Fig 1 Home page of web page


Normal Retina
healthy retina without any visible abnormalities. Consistent monitoring of retinal health is important to detect early signs of potential eye diseases.
Start to recognise the retinal image
Start

Appendices Fig 2 Learn more page of web interface



Appendices Fig 3 Detect page of web interface

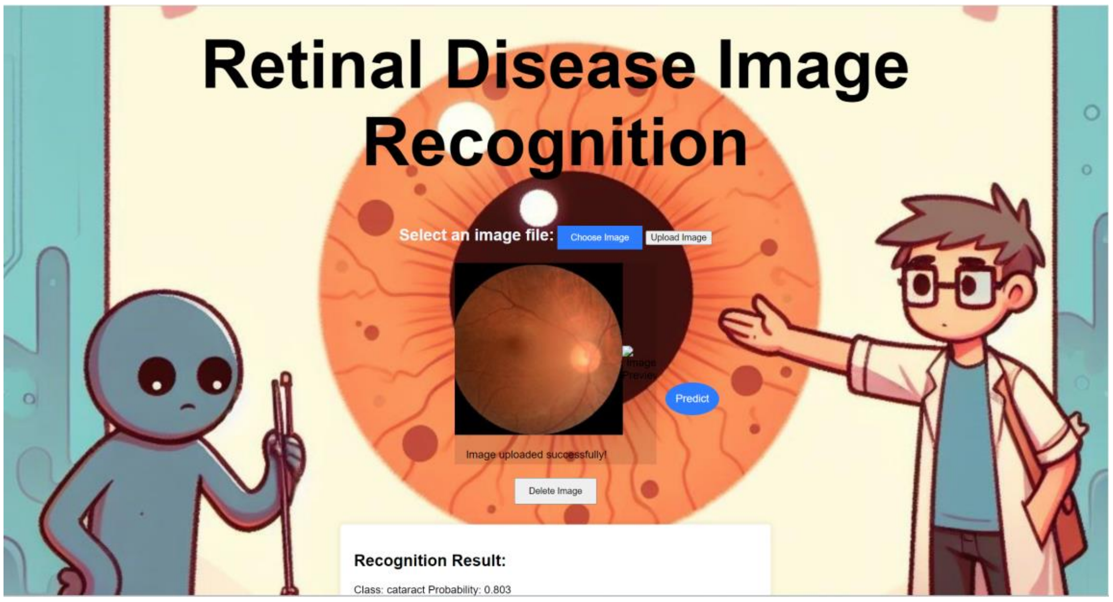


Image uploaded successfully!

Recognition Result:

Class: cataract Probability: 0.803

Appendices Fig 4 Demonstration of the retinal image recognition