College of Engineering and Physical Sciences

School of Informatics & Digital Engineering, Computer Science

Final Report

Explained Computer-Aided Diagnostic System for Ocular Disease Recognition

Duwa Khan

210044641@aston.ac.uk

Supervisor: Shereen Fouad

April 2024

Abstract

In recent years, image classification has seen widespread application across various industries, yielding impressive results. However, the deployment of deep learning for classification in the medical industry poses significant challenges, notably in trust and transparency. This study addresses these issues by training five different CNN-based image classification algorithms, focusing on identifying the most effective model for a multi-ocular disease diagnostic system that ensures high interoperability. To enhance the reliability of the fundus image classification model, various Explainable AI (XAI) techniques were implemented, including LIME, SHAP, and Grad-CAM. These methods were strategically used to ensure that the system's predictive accuracy is trustworthy and transparent, making it suitable for clinical applications.

Keywords: Healthcare, Computer-aided Diagnostic System, Image Classification, Deep Learning, Explainable AI (XAI), Fundus Images

Contents

Figures

Tables

List of Symbols and Abbreviations

AI	Artificial Intelligence
XAI	Explainable AI
CNN	Convolutional Neural Network
GANs	Generative Adversarial Networks
LIME	Local Interpretable Model-agnostic Explanations
Grad-CAM	Gradient Class Activation Mapping
SHAP	Shapley Additive Explanations

1 Introduction

1.1 Motivation

In the past few years, there has been a growing usage of deep learning (DL), a branch of artificial intelligence (AI) that allows computers to imitate human behaviour, in the healthcare industry, particularly for diagnosing medical conditions using images (Jiang et al., 2021). DL can identify patterns efficiently and process vast amounts of data effectively, resulting in quicker and more precise disease diagnosis (Parikh and Helmchen, 2022).

However, in the implementation of DL models, particularly convolutional neural networks (CNNs), for medical image diagnosis, interpretability of the decision-making process of these models arises as a common challenge. The "black box" nature (Shrikumar *et al.*, 2016) inherent to neural networks often explains this ambiguity. In the healthcare domain, this lack of clarity raises substantial concerns about the trust and transparency of machine learning-enabled AI tools.

To address these challenges, this study proposes developing a targeted computer-aided diagnostic system tailored for ocular disease recognition. The focus will be on employing multi-class classification techniques to analyse and compare the accuracy of diagnoses generated by 5 CNN algorithms (ResNet-50, VGG16, VGG19, Xception, InceptionV3). Additionally, various explainable AI (XAI) techniques will investigated to interpret and understand the results produced by these models. This approach aims to shed light on the decision-making processes of the models, validating their accuracy and enhancing transparency in the diagnostic results, as a result, fostering confidence in the clinical use of medical AI applications for precise and reliable patient diagnoses.

The publicly available Ocular Disease Recognition (ODIR5K) dataset (Peking University, 2019) features a multi-class representation of ocular diseases of 5,000 patients with age, colour fundus photographs (CFPs), and diagnostic keywords. The patients can be classified into eight classes including Normal (N), Diabetes (D), Glaucoma (G), Cataract (C), Age-related Macular Degeneration (AMD), Hypertension (H), Pathological Myopia (M), Other diseases/abnormalities(O). This dataset is chosen since it includes fundus photographs, which are used to document disease progression and assist in early detection and treatment of eye diseases. The early detection of ocular diseases is crucial in the prevention of vision impairment and blindness as it can help with the timely management of the disease, hence a reliable DL-based diagnostic system for ocular disease recognition would be very beneficial if clinically successful as it can potentially reduce the workload of specialist doctors with DL opening doors to automated ocular disease screenings.

XAI techniques, LIME, Grad-CAM and SHAP will be applied. A comparative analysis will assess their effectiveness and contribution to enhancing trustworthiness and informed decision-making in a medical environment.

1.2 Research Questions and Objectives

1.2.1 Research Questions:

Primary Question: " What methods can be used to improve the understandability of machine learning models in computer-based medical diagnosis systems to promote transparency and confidence?"

Secondary Question: "Can a comparative analysis of various convolutional neural networks (CNNs), in conjunction with explainable AI techniques like LIME, Grad-CAM and others, effectively address challenges identified in prior studies, leading to improved accuracies and the establishment of a more reliable computer-aided diagnostic system for ocular diseases for potential clinical use?"

1.2.2 Objectives

1) Construct distinct image classification models for each identified deep learning algorithm, encompassing various CNN architectures.
2) Conduct a comprehensive evaluation to assess the results and accuracies of these models.
3) Undertake an in-depth comparative analysis, contrasting the performance of each deep learning algorithm to identify respective strengths and weaknesses and obtain the best performance model.
4) Integrate LIME, GradCAM, and SHAP into the best-performing model to enhance interpretability.
5) Validate the outputs generated by XAI techniques for correctness and reliability.
6) Conduct a subsequent comparative analysis of the XAI techniques to determine superior accuracy and reliable validation.

The insights derived from these analyses will be pivotal in understanding how the implemented strategies contribute to transparency, trust, and the overall reliability of computer-aided diagnostic systems, significantly contributing to the refinement and advancement of diagnostic systems, particularly in the context of ocular eye diseases.

1.3 Organisation

The report will consist of ten sections, starting with an introduction that summarises the project's information, goals, and research questions. The literature review in the background research section will be thorough, focusing on the distinctiveness of the study and proposing ways to apply the findings to the dataset for better results. After that, the paper will discuss how data is collected and analysed, the strategies for implementing deep learning models, and a detailed explanation of the architecture of each model. Descriptions of the XAI methods utilised will be given. The methodology section will outline the process of implementation, model accuracies, evaluation metrics for the CNN architectures, and a discussion of the results. Visuals from the XAI research will be displayed and analysed. Before discussing limitations, the code implementation will be assessed. A section on project management will follow. The final part will provide a detailed evaluation of the results, perspectives, and suggestions for upcoming studies while acknowledging any limitations faced. The appendices will contain the system's code, a project diary, and an ethical declaration form.

2 Background Research

In previous studies, the research associated with the ODIR5K (Peking University, 2019) dataset has prominently featured deep learning architectures and diverse classification methodologies for computer-aided diagnostic systems. Despite this, a noticeable gap in the literature lies in the exploration of multiple XAI techniques and a comparative analysis of interpretability within these studies. All studies mention the issue of class imbalance and provide techniques and suggestions on how to resolve this.

2.1 Literature Review

Khan et al. (2022) carried out binary classification on the ODIR5K dataset and implemented transfer learning, which is a machine-learning method where knowledge gained from one problem is used to solve a similar problem

(Weiss, Khoshgoftaar and Wang, 2016). It is valuable to use where there is a limited amount of training data for the new task, thus by transferring knowledge from a related task with abundant data, the model can generalise a lot better than if it was trained from scratch on the initial small dataset. A pre-trained VGG19 model is utilised to achieve an accuracy of 98.10%, however, the study missed the integration of explainable AI techniques, hindering the interpretability of its model. The imbalanced dataset was rectified with a combination of transfer learning and data augmentation and the research suggested the use of generative adversarial networks (GANs) to produce synthetic fundus images to resolve the class imbalance further and explore image segmentation to further refine accuracy in image classification of the study dataset.

Another paper (Hassan *et al.,* 2023) delved into the biological traits of patients within the study dataset, emphasising age and gender, proposing FAG-Net for age and gender estimation using fundus images. With an accuracy of 91.87%, the study employed saliency maps to detail biological traits and define causal regions. The paper made future suggestions for using saliency maps in medical image classification and proposed the implementation of more sophisticated deep-learning models with attention mechanisms.

Addressing the class imbalance in the study dataset, a distinct study implemented both multiclass and multilabel classification techniques, utilising an InceptionResNet architecture augmented with a DKCNet block, to generate discriminative feature attention maps, achieving an accuracy of 96.08%. For enhancing the interpretability of the model, Grad-CAM was used. The images with various artefacts, such as "low-quality image," "optically invisible disk," "lens dust," and "image offset,", to effectively reduce the rate of false positives. To combat the class imbalance, a series of random sampling techniques were employed. The paper made future suggestions of using synthetic fundus image generators to tackle the class imbalance and the further implementation of explainable AI techniques for lesion detection and localisation (Bhati *et al.,* 2023).

In the study by Wang et al. (2020), EfficientNet was implemented for feature extraction through multi-label classification, employing a pre-trained model initialised via transfer learning. This model, stripped of its top layer for enhanced feature extraction, was coupled with a custom multi-label classifier. While the study reached an accuracy of 73%, it did not address interpretability. The researchers noted that the 'O' label (representing other diseases/abnormalities) within the ODIR5K dataset encompasses a variety of uncommon diseases, with some labels suffering from limited data availability. This scarcity of data posed challenges to improving the neural network model's performance. The paper highlighted these limitations, suggesting an expansion of the study's dataset size, particularly for rarer conditions, and proposed the incorporation of demographic factors such as age, gender, and family history to refine fundus disease detection further.

Kinger and Kulkarni (2022) developed an ocular diseases classification model, incorporating LIME to explain the image classifications for accountability, transparency, and debugging in the healthcare domain. With an accuracy of 92.81%, the study underscored the importance of XAI in healthcare and urged the further exploration of XAI techniques and diverse deep neural network algorithms for comprehensive comparison and optimisation.

A multi-class classification implementation by Dipu, Shohan and Salam (2021) achieved an accuracy of 97.93% accuracy with an ImageNet pre-trained VGG-16 model. However, no explainability techniques were implemented. No future suggestion is made in the paper.

2.2 Discussion

Drawing inspiration from these studies, the present research aims to incorporate multi-class classification instead of mult-label and similar deep-learning architectures and use multiple XAI techniques to overcome their limitations in achieving accuracy by building a reliable computer-aided diagnostic system that can be used in the healthcare industry.

In the pursuit of enhancing multi-class classification, the research will use suitable but different deep-learning algorithms to those previously used while addressing class imbalances with the study dataset. Techniques such as over-sampling, under-sampling, and data augmentation will be leveraged to attain a balanced representation of classes. The study dataset will be preprocessed to ensure the images are of uniform size, resolution and format. Since transfer learning will be used, Keras, an interface for artificial neural networks, supported by Tensorflow will provide the pre-trained ImageNet dataset models. ImageNet is a natural image dataset comprised of over 15 million human-annotated images belonging to 1000 classes, it was created by academics for the development of computer vision algorithms (Brownlee, 2019). This dataset is used to provide the weights for transfer learning models, even though the images may not be similar to the study set, Dipu, Shohan and Salam (2021) yielded high accuracy with the use of various transfer learning models. The aim will be to take the pre-trained network, replace the last few layers with new ones suited to the task at hand, and fine-tune the model on the smaller dataset.

2.2.1 CNN Model Selection

The exploration of both VGG-16 and VGG-19 CNN architectures aims to contribute to the diversity of the model set. While VGG-19 may introduce potential overfitting issues due to its increased complexity, a strategic solution is proposed. Transfer learning will be employed to specifically address the class imbalance. VGG-16 was chosen for its strong performance in previous implementations but noted a lack of transparency; thus, this study will aim to elucidate its decision-making processes to further enhance trust in its high accuracy. Additionally, ResNet50, Xception, and InceptionV3 will be utilised. The combination of ResNet and InceptionV3 has previously yielded promising results (Bhati et al., 2023); hence, Xception is introduced in this study to provide new insights into the dataset.

Efficient Net will be intentionally excluded based on insights from prior studies (Wang et al., 2020), which reported suboptimal results. This decision highlights the importance of making informed architectural choices, aligning with the overarching goal of developing effective and efficient deep learning models for the given multi-class classification task.

2.2.2 Explainability Techniques Selection

For an in-depth exploration of XAI techniques, as highlighted by Kinger and Kulkarni (2022), this study will implement both model-agnostic and model-specific explainability methods. Previous studies have predominantly focused on model-specific methods such as Saliency Maps and Attention Maps, which provide explanations tailored to specific types or instances of machine learning models. These methods, while insightful, are inherently model-bound and may not be easily applicable to other models. In contrast, model-agnostic methods offer explanations that are independent of the neural network's architecture, focusing instead on the model's inputs and outputs to generate explanations.

The investigation will include model-specific methods like GradCAM, which is particularly designed for convolutional neural networks (CNNs). GradCAM produces visual explanations in the form of heat maps, pinpointing crucial areas in an image that influence the model's predictions.

On the model-agnostic front, methods such as LIME (Local Interpretable Model-agnostic Explanations) and SHAP (SHapley Additive Explanations) will be utilised. LIME facilitates the understanding of individual predictions by generating local approximations of the model's decision boundary. Meanwhile, SHAP values quantify the contribution of each feature to a model's prediction, offering a cohesive framework for assessing feature importance.

To conduct a comprehensive comparative analysis of these XAI methods this study will adopt the criteria proposed by Sun et al. (2018). Their research provides a standard set of metrics for DL engineers to interpret results more effectively when using various explainer methods. In this context, a model for automated skin lesion classification will be developed and evaluated using LIME, GradCAM, and SHAP, with comparisons drawn based on the following criteria:

1. **Consistency**: Evaluates the reproducibility of results under identical experimental conditions. For example, does running LIME multiple times yield similar explanations?
2. **Fidelity**: Measures the extent to which the explanation method accurately reflects the model's actual decision-making process. The quality of an XAI method's fidelity depends on both the truthfulness of the method to the model and the model's accuracy.
3. **Sensitivity**: Assesses the XAI method's responsiveness to changes in input data or its predictive class. This involves using augmented data to test the adaptability of the model, as explanations should vary to reflect any modifications in the data.
4. **Clinical Relevance**: Determines the alignment of XAI method explanations with clinical considerations and expert knowledge, particularly critical for medical applications.

This research aims to encompass these criteria in the comparative analysis of explainability methods. Figure 1 below presents a visual comparison of the XAI techniques employed in this study.

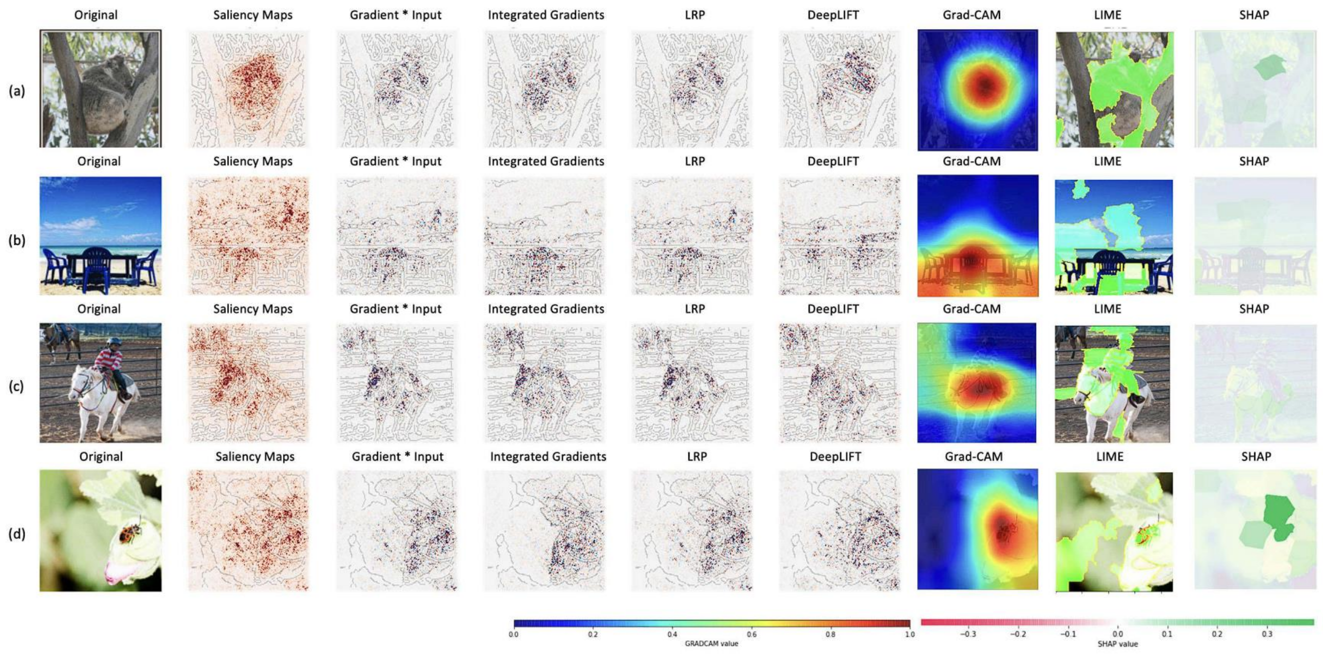


Figure 1 Comparative results of model-specific (e.g., Grad-CAM) and model-agnostic methods (Das and Rad, 2020)

3 Methods of Detecting Ocular Eye Diseases

3.1 Use of Deep Learning for Ocular Detection

Transfer learning will be applied to five pre-trained CNN-based models to develop a multi-class image classification model. The performance of each model will be compared, with the best-performing model further analysed using three explainability methods.

3.1.1 Convolutional Neural Network for image classification

CNNs excel in image classification and object recognition due to their ability to effectively process image data (Brownlee, 2021). They utilise convolutional layers to apply filters to input images, capturing essential features such as edges, textures, and shapes. These features are instrumental in the classification process.

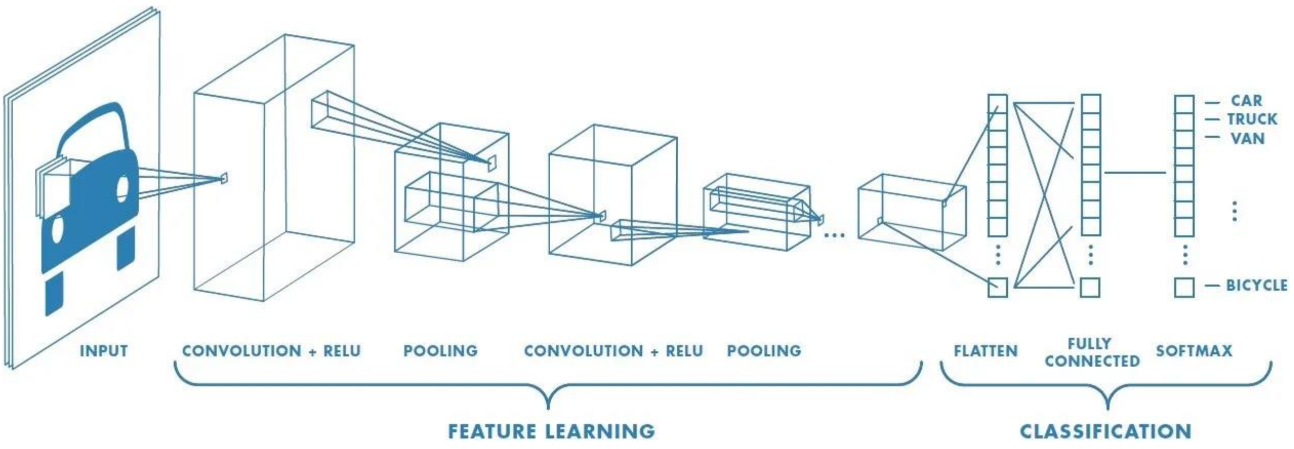


Figure 2 CNN For Image Classification (Saha, 2023)

Figure 2 illustrates a diagram example of image classification using the CNN algorithm.

The process starts with the initial image input, which then goes through a series of convolutional layers that utilise filters to identify features. Following each convolutional layer, it is common to include a Rectified Linear Unit (ReLU) activation function to introduce non-linearity, allowing the neural network to understand complex patterns.

Following each convolutional layer, there is a pooling layer that decreases the representation's size, resulting in a reduction of parameters and computations in the network, ultimately aiding in preventing overfitting. This procedure also guarantees that the recognition of features stays reliable even if changes occur in dimensions and orientation, enhancing the overall accuracy of the image categorisation system. The process of feature learning concludes with a flattening step, which transforms the 3D result from the last pooling layer into a 1D feature vector. The vector is inputted into a sequence of fully connected layers where the "learning" occurs through adjustments of weights.

In the end, a SoftMax layer categorises the image into different classes like car, truck, van, and bicycle by giving probabilities to each class, and the class with the highest probability is the model's output. This structure allows the CNN to analyse unprocessed image data, understand characteristics, and forecast the image's content in a step-by-step approach.

3.1.2 Transfer Learning

Transfer Learning is applied using the following steps below:

1. Obtain layers from a previously trained model on the ImageNet dataset.
2. Freeze them, to avoid destroying any existing knowledge they contain during future training rounds.
3. Add some new trainable layers on top of the frozen layers. These new layers will learn to turn the old features into predictions on the new dataset.
4. Then train the new layers on the study dataset.
5. Apply finetuning, which consists of unfreezing a few layers of the entire model obtained and retraining it on the study dataset at a low learning rate to achieve more improvements.

3.2 CNN Models

Within this paper, the TF-Keras Application models for the architectures listed are used for implementing the image classification system.

3.2.1 VGG16

Developed by Simonyan and Zisserman (2014), VGG16 is a deep CNN known for its efficacy in image classification. Its architecture, depicted in Figure 3, consists of 16 weighted layers, including several 3×3 convolutional layers and max-pooling layers, culminating in three fully connected layers and a SoftMax output layer. Its depth and small convolutional filters contributed to its top-5 test accuracy of 92.7% on ImageNet. However, the risk of overfitting on smaller datasets is a known issue, which can be mitigated by regularisation and data augmentation techniques.

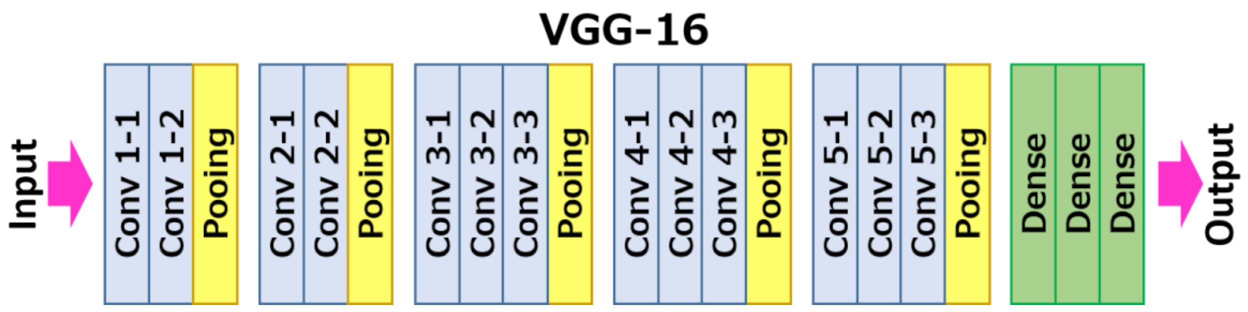


Figure 3 VGG16 Model Architecture (Hassan, 2023)

3.2.2 VGG19

VGG19, a continuation of VGG16, consisting of an extra three layers, possibly allowing it to grasp more intricate features and slightly enhancing its performance in specific assignments. Nevertheless, the improvements in performance could be limited and greatly influenced by the task and dataset. Additionally, the extra depth requires increased computational resources.

3.2.3 ResNet50

ResNet50, a residual network designed by He et al. (2015), consists of 50 layers and introduces skip connections that address the vanishing gradient problem, aiding in the training of deeper networks.

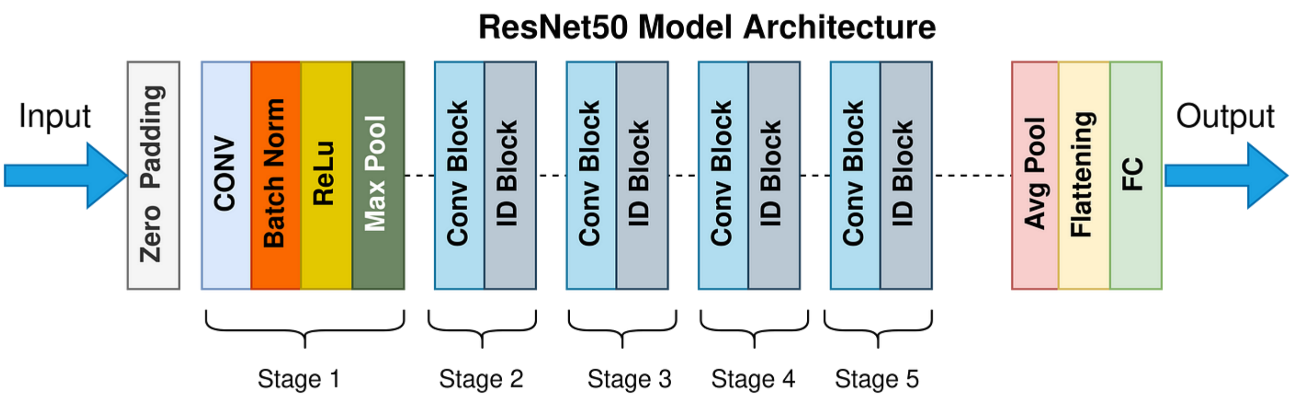


Figure 4 ResNet50 Model Architecture (Mukherjee, 2022)

Mathematical Explanation of How Residual Blocks Work in ResNet50:

Assume you have an input x. In a traditional neural network layer, you might pass x through a set of operations (like convolution, batch normalisation, and activation) to get an output, which we can call $H(x)$.

In a residual block, rather than learning $H(x)$ directly, the layers learn a residual function $F(x)$ with the idea that learning this residual is easier:

$F(x)=H(x)-x$ (1)

The output of the residual block then becomes:

$H(x)=F(x)+x$ (2)

3.2.4 InceptionV3

InceptionV3, created by Szegedy et al. in 2015, improves upon the original Inception design by incorporating factorised and asymmetric convolutions and utilising advanced grid size reduction techniques to improve both efficiency and accuracy in image recognition.

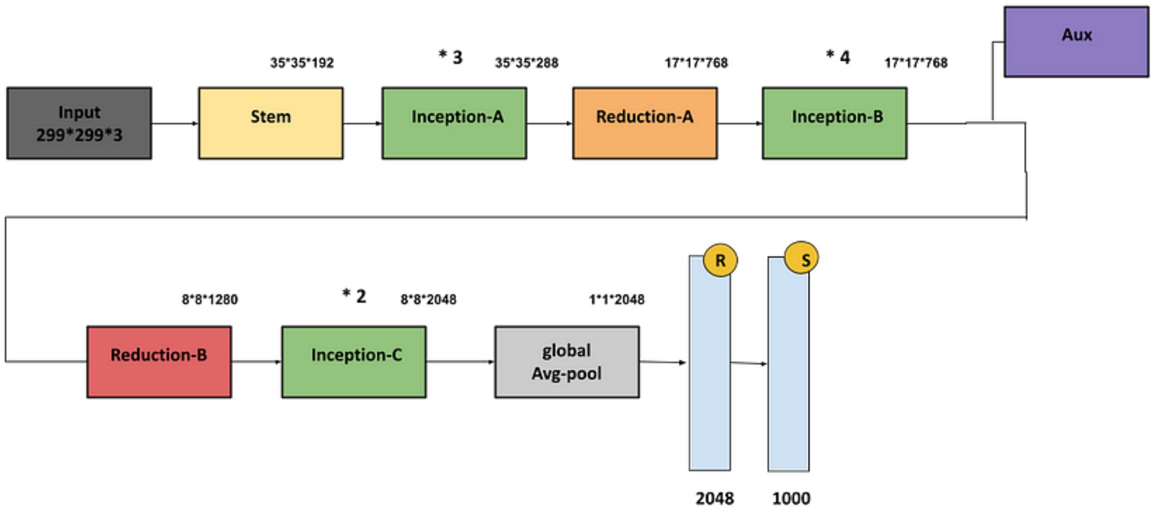


Figure 5 InceptionV3 Model Architecture

3.2.5 Xception

Xception, introduced by Chollet (2016), takes inspiration from the Inception architecture but pushes the idea of separable convolutions to its logical extreme. Xception stands for "Extreme Inception" and replaces Inception modules with depthwise separable convolutions. This approach essentially involves breaking down the convolution operation into two separate layers: one for filtering across the spatial dimensions (depthwise convolution) and another for combining the outputs across channels (pointwise convolution). The Xception architecture demonstrates that by rethinking the Inception philosophy and applying depthwise separable convolutions, one can achieve not only a more efficient use of model parameters but also improved performance on tasks such as large-scale image classification. The Xception model has been shown to outperform InceptionV3 in various benchmarks, highlighting the effectiveness of depthwise separable convolutions for deep learning models focused on computer vision. Figure 8 illustrates its model architecture.

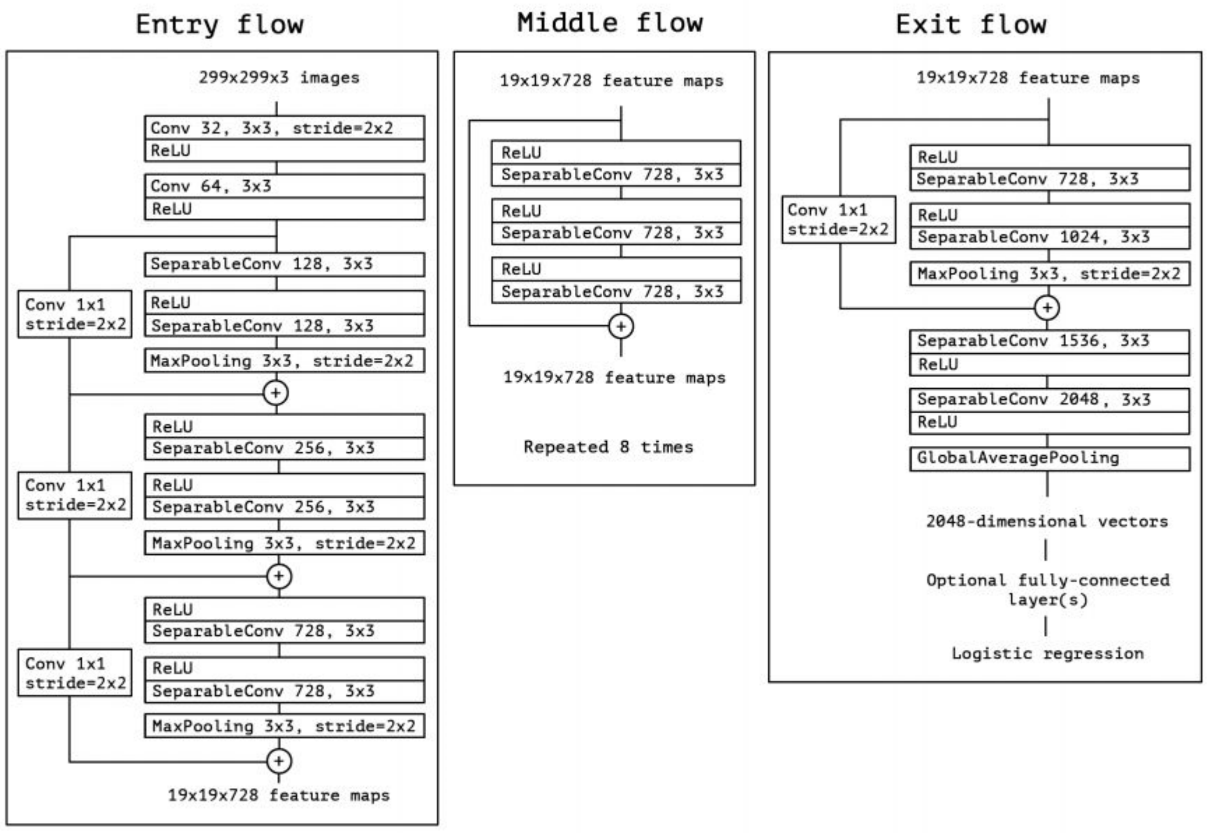
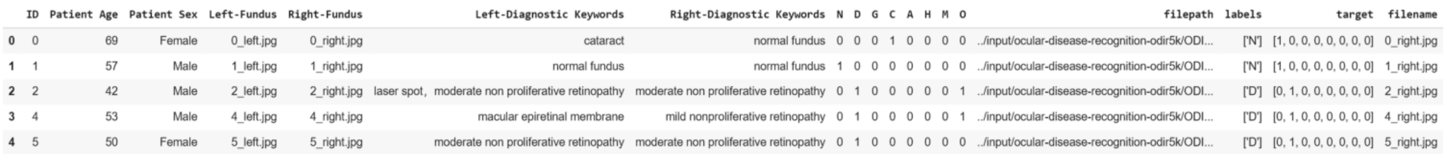


Figure 6 Xception Model Architecture (*XCeption model and depthwise separable convolutions*, 2019)

4 Data Collection and Analysis

4.1 Dataset

The ODIR dataset, sourced from Kaggle's "Ocular Disease Recognition" (2020), encompasses a comprehensive collection of patient data compiled by Shanggong Medical Technology Co., Ltd., originating from various hospitals and medical centres across China. The dataset is characterised by colour fundus images captured using different cameras, leading to a range of image resolutions. It is thoughtfully organised into pre-split training and testing folders, complemented by a detailed data frame file named 'full_df.csv', available in Excel format, along with pre-processed versions of the training images. Figure 7 illustrates the dataset structure, showcasing the first five rows of the 'full_df.csv' file. Classification labels for both eyes, accompanied by descriptive diagnostic keywords, are provided for each patient. The 'target' column, structured for multi-class classification and one-hot encoded, assigns a binary value of 1 or 0 to denote the presence or absence of a specific ocular condition, with all eight classes represented. Figure 3 also includes a selection of fundus images from the training set, exemplifying the dataset's utility for this study.



	ID	Patient Age	Patient Sex	Left-Fundus	Right-Fundus	Left-Diagnostic Keywords	Right-Diagnostic Keywords	N	D	G	C	A	H	M	O	filepath	labels	target	filename
0	0	69	Female	0_left.jpg	0_right.jpg	cataract	normal fundus	0	0	0	1	0	0	0	0	../input/ocular-disease-recognition-odir5k/ODI...	['N']	[1, 0, 0, 0, 0, 0, 0, 0]	0_right.jpg
1	1	57	Male	1_left.jpg	1_right.jpg	normal fundus	normal fundus	1	0	0	0	0	0	0	0	../input/ocular-disease-recognition-odir5k/ODI...	['N']	[1, 0, 0, 0, 0, 0, 0, 0]	1_right.jpg
2	2	42	Male	2_left.jpg	2_right.jpg	laser spot， moderate non proliferative retinopathy	moderate non proliferative retinopathy	0	1	0	0	0	0	0	1	../input/ocular-disease-recognition-odir5k/ODI...	['D']	[0, 1, 0, 0, 0, 0, 0, 0]	2_right.jpg
3	4	53	Male	4_left.jpg	4_right.jpg	macular epiretinal membrane	mild nonproliferative retinopathy	0	1	0	0	0	0	0	1	../input/ocular-disease-recognition-odir5k/ODI...	['D']	[0, 1, 0, 0, 0, 0, 0, 0]	4_right.jpg
4	5	50	Female	5_left.jpg	5_right.jpg	moderate non proliferative retinopathy	moderate non proliferative retinopathy	0	1	0	0	0	0	0	0	../input/ocular-disease-recognition-odir5k/ODI...	['D']	[0, 1, 0, 0, 0, 0, 0, 0]	5_right.jpg

Figure 7 First 5 rows of data frame file for the ODIR dataset.

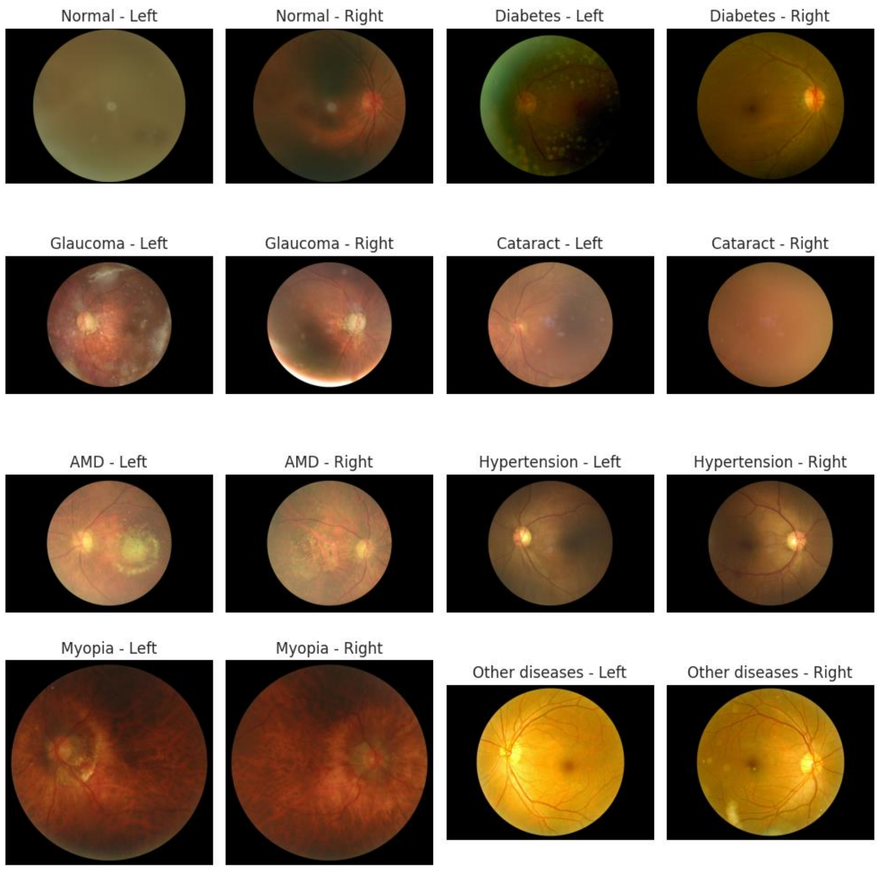


Figure 8 Colour Fundus Images from each class from the dataset.

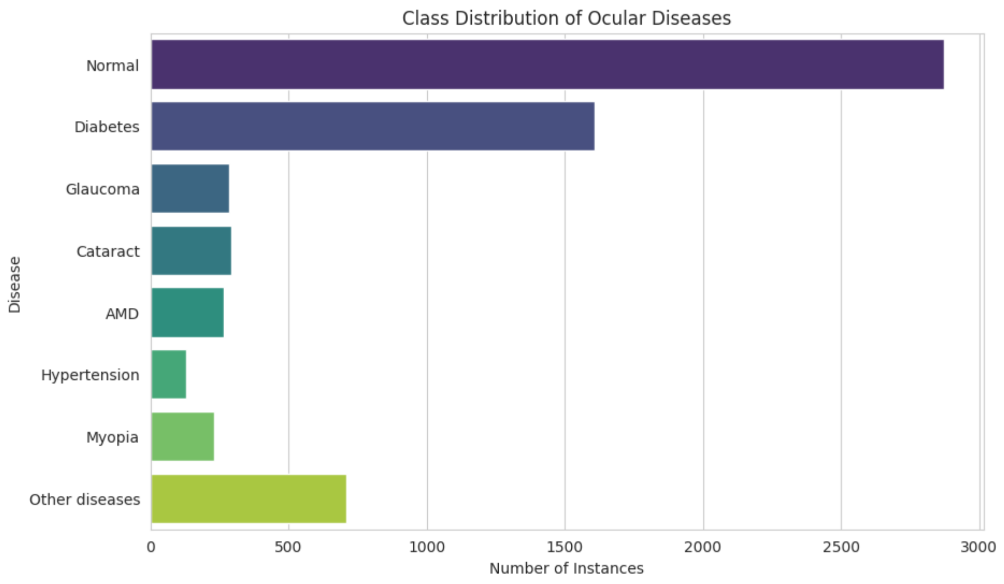


Figure 9 Bar chart representing the distribution of the dataset.

4.2 Challenges of the Dataset

The study dataset presents several challenges:

- **Unlabelled Test Data**: Only the training folder, which includes labelled images, will be used due to the absence of labels in the testing folder.
- **Class Imbalance**: There is a significant class imbalance, with the 'Normal' class outnumbering others such as 'Hypertension.' This will be addressed using random sampling, transfer learning, and data augmentation.
- **Varied Image Resolutions**: The diversity in image resolutions necessitates effective preprocessing to ensure uniform data quality for good model performance.

5 Explainable AI

5.1 Explainable AI Methods

Below, each XAI technique's process in providing explanations is detailed.

5.1.1 Lime

LIME was introduced in 2016 by Ribeiro, Singh and Guestrin (2016), it is an algorithm that can explain individual predictions of any classifier model. LIME uses a superpixel algorithm as part of its process to explain the predictions of machine learning models, particularly for image data. The algorithm divides an image into superpixels, which are groups of pixels that share similar attributes. LIME interprets the model's behaviour by considering these superpixels as features. LIME can identify the most influential features of an image by manipulating superpixels and analysing the model's predictions based on these changes. This method enables a clearer explanation of intricate models, especially in situations where visual data is essential.

5.1.2 SHAP

The technique of Shapley Additive Explanations, which utilises Shapley values from game theory, was first presented by Lundberg and Lee in 2017 to explain individual predictions made by machine learning models. It views every attribute in the data set as a "participant" in a competition where the forecast generated by the model serves as the reward.

By calculating the contribution of each feature to the prediction (the Shapley value), SHAP provides a detailed breakdown of how each feature influences the model's output for a particular instance. This is achieved by evaluating the model with subsets of features, both with and without a specific feature, to understand its marginal contribution across all possible combinations. The aggregation of these contributions gives a comprehensive view of the feature's importance towards the model's prediction. This method allows for highly detailed and interpretable explanations, especially useful for complex models where direct interpretation is challenging.

5.1.3 Grad-CAM

Grad-CAM is a technique tailored for interpreting the decisions of convolutional neural networks (CNNs), especially in visual tasks such as image classification. It works by computing the gradients of any target concept like a class label in the output layer concerning the feature maps of a convolutional layer. These gradients are pooled to derive a set of weights that highlight the importance of each feature map for predicting the target concept. A weighted combination of these feature maps creates a "heatmap" of the same size as the convolutional output, visually indicating the crucial areas in the input image for the model's prediction. By overlaying this heatmap on the original image, Grad-CAM provides a visual explanation that points to the areas in the image most influential for the classification decision. This method is especially useful for gaining understanding into what the model perceives and deems significant during prediction. (Selvaraju et al., 2019)

6 Methodology

6.1 Experiment Process

In this project, I conducted a series of five experiments to identify the most effective classification model using the ODIR-5K dataset. The implementation was carried out using Python and TensorFlow. Initially, I began the development in Visual Studio Code, utilising the computational resources available on my local laptop. However, these resources were insufficient for the task at hand. Consequently, I transitioned to using Google Collab, which provided the necessary computational power needed for more computer vision-based intensive tasks.

All pre-trained models used in this study were sourced from the Keras 2 API documentation (Team, no date), ensuring that I utilised well-established and robust models as the foundation for our classifier.

6.1.1 Data Preprocessing

The data was first split into training, test and validation sets using the train_test_split function from the sklearn.model_selection library. The allocated proportions were 20% for testing, 64% for training, and 16% for validation. Table 1 presents the class distribution across each set.

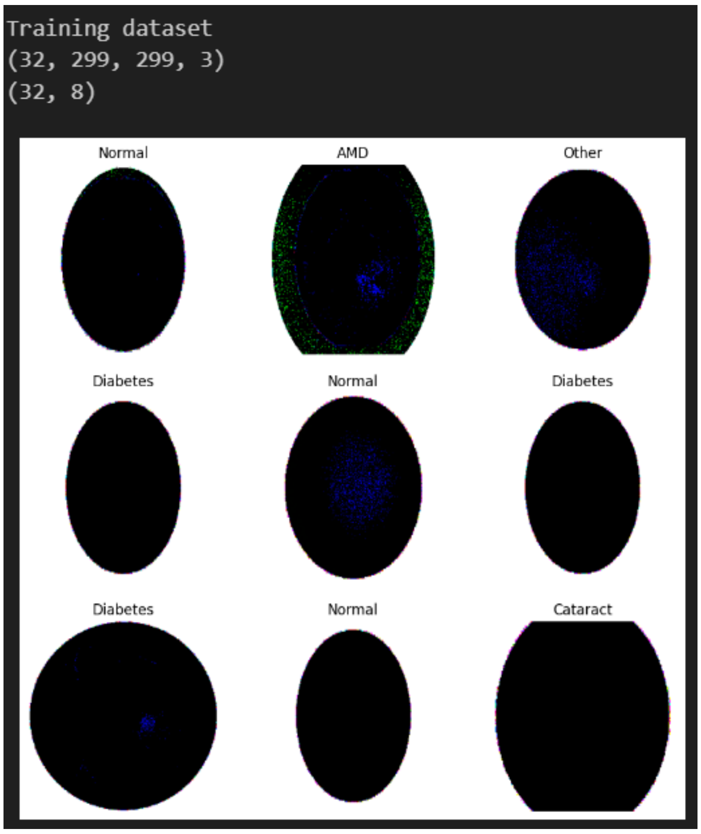
A unified preprocessing pipeline was utilised across all models, with adaptations for image resolution and specific preprocessing functions tailored to the architecture of each pre-trained model. This foundational preprocessing step involved converting images to a binary grid of pixels and transforming the labels into a binary format, necessary due to the one-hot-encoded target column's string data type in the data frame.

Given that all selected models were pre-trained on the ImageNet dataset, they required the implementation of ImageNet's original preprocessing routines. This practice is critical in transfer learning to ensure that the fundus images are pre-processed identically to the ImageNet data. Maintaining this consistency is crucial, as it allows the pre-trained models to correctly interpret the new data. It aligns the pixel value distributions of the fundus images with those of ImageNet, which is essential for effective knowledge transfer and optimal model performance. Figures 10 and 11 provide examples of the ImageNet preprocessing function applied to the colour fundus images of the datasets. The images have also been resized differently for the models shown. For VGG16, VGG19 and ResNet50, the input data was resized to [224,224] and for Xception and InceptionV3 the input data was resized to [299,299].

To tackle the class imbalance, simple data augmentation techniques, like horizontal and vertical flips were applied to the training data only and a combination of random oversampling and under sampling was applied. I used two different thresholds when random sampling, so if the distribution of class was less than half the size of the largest class size which is Normal, these classes were oversampled, but only up to 80% of the size of the largest class. Classes greater than 80% of the largest class size were downsampled to that size. Figure 12 displays the distribution of classes after sampling was applied.

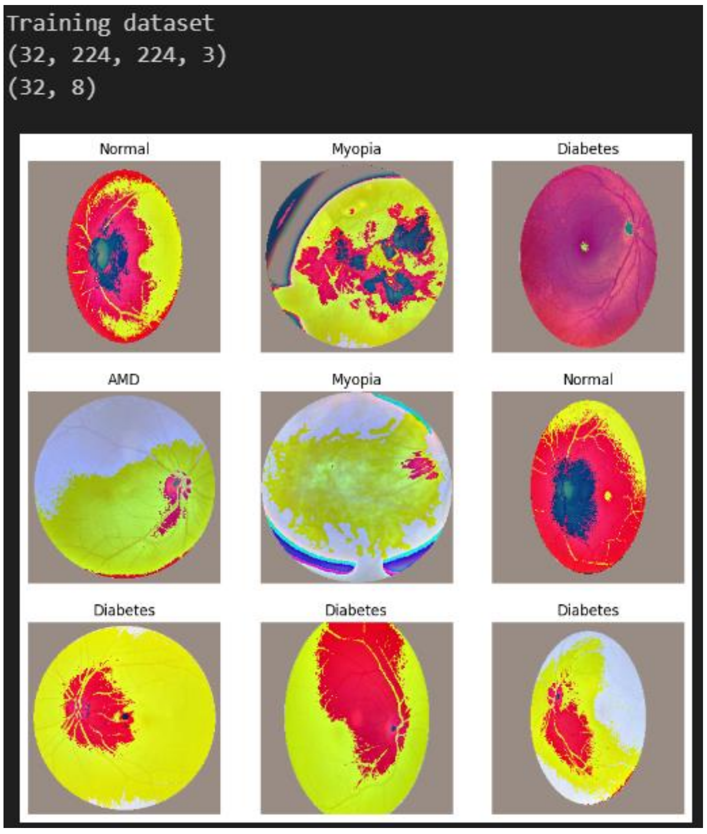
Table 1 Class Distribution of Train, Test, and Validation Data

Classes	**Training Cases**	**Testing Cases**	**Validation Cases**
Normal	1870	554	449
Diabetes	1036	325	247
Glaucoma	430	153	125
Cataract	183	62	58
AMD	171	57	48
Hypertension	169	50	45
Myopia	153	50	29
Other Disease	78	28	22



```
Training dataset
(32, 299, 299, 3)
(32, 8)
```

Figure 11 Result of Xception Preprocessing Function



```
Training dataset
(32, 224, 224, 3)
(32, 8)
```

Figure 10 Result of VGG19 Preprocessing Function

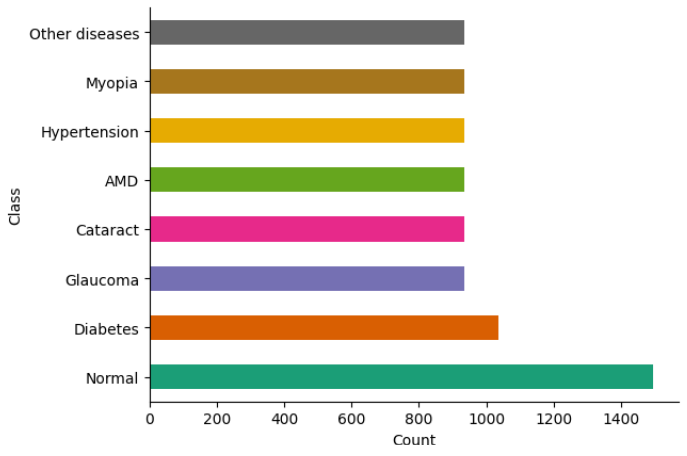


Figure 12 Class Distribution After Random Sampling

6.1.2 Hyperparameter configuration

Hyperparameters are essential in machine learning models, as they govern the learning process and significantly influence the models' performance. Tables 2 and 3 outline the configurations of the hyperparameters I applied to two versions of the image classification models I developed: the base model and the model utilising random sampling. During the hyperparameter tuning process, I adjusted parameters individually to observe their effects via the model loss and accuracy plots. This approach ensured the model was generalising effectively without overfitting. Additionally, I prioritised parameters that enabled faster training times, an important consideration due to the computational resource limitations encountered on Google Colab. The following hyperparameters were kept the same for all 5 models.

Table 2 Hyperparameters for base model

Configuration	**Value**
Optimisation Function	Adam
Epoch	10 (Complete training), 80 (Finetuning)
BatchSize	32
Learning Rate	0.0001
Dropout	0.2
ReduceLROnPlataeu	monitor=‘val_loss’, factor=’0.5’, patience=’3’, min_lr=’1.00E-07’
EarlyStopping	monitor=‘val_loss’, patience=‘5’
ModelCheckpoint	save_best_only=True

Table 3 Hyperparameters for resampled training data model

Configuration	**Value**
Optimisation Function	Adam
Epoch	80 (Complete training), 80 (Finetuning)
BatchSize	32
Learning Rate	0.0001
Dropout	0.2
L2 Regularisation	kernel_regularizer=l2(0.001)
ReduceLROnPlataeu	monitor=‘val_loss’, factor=’0.5’, patience=’3’, min_lr=’1.00E-07’
EarlyStopping	monitor=‘val_loss’, patience=‘5’
ModelCheckpoint	save_best_only=True

Given that this project addresses a multi-class classification problem, categorical_crossentropy was selected as the most appropriate loss function. This choice is predicated on its effectiveness in dealing with multiple classes where each instance is exclusively assigned to one class out of many. Categorical_crossentropy, also known as SoftMax loss, measures the difference between the predicted probabilities and the actual distribution of the classes. It computes the probability that a given input belongs to each of the classes, to minimise the loss by adjusting the model weights to predict a probability distribution as close as possible to the true distribution.

6.1.3 Model Architectures

In developing the image classification model, I used pre-trained models from Keras that were originally trained on the ImageNet dataset. These models served as the foundational architecture, with the top layer omitted and average pooling applied to adjust for the specific classification task. To ensure that the learned features from ImageNet were preserved, all pre-trained layers were set to non-trainable, effectively freezing them. This step was crucial for maintaining the integrity of the features that these layers had already learnt, which are generally applicable across a wide range of visual recognition tasks.

Subsequently, the model was customised for the image classification challenge of the study dataset by adding new layers designed for the task at hand. This customisation included:

- A dense layer consisting of 512 units with ReLU (Rectified Linear Unit) activation, was chosen for its ability to introduce non-linearity into the model, making it capable of learning more complex patterns.
- A dropout layer for regularisation to prevent overfitting. By randomly setting a fraction of input units to 0 at each update during training, dropout helps to make the model more robust and less prone to memorising the training data.
- A final dense layer equipped with a SoftMax activation function to output predictions across the 8 classes. The SoftMax layer is essential for multi-class classification tasks as it converts the model's raw predictions, into probabilities by taking the exponential of each output and then normalising these exponentials.

This architectural setup for each model is illustrated in Figures 13-17, depicting the standard structure adopted for all base model implementations.

During the later stages of training, the last 4 layers of the model were unfrozen to allow for fine-tuning. This fine-tuning process is instrumental in adapting the model more closely to the specific requirements of the new task. By making these layers trainable, they can update their weights in response to the new data, potentially capturing nuances that are specific to the task at hand. To safeguard against disrupting the pre-learned features too drastically, the model was recompiled with a significantly reduced learning rate for this phase. Such a cautious approach ensures that updates are subtle, enhancing the model's performance on the new task by refining rather than overwriting the detailed aspects of its learning.

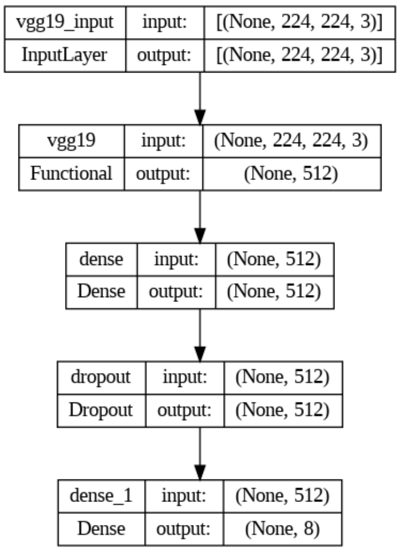

vgg19_input | input: | [(None, 224, 224, 3)]
InputLayer | output: | [(None, 224, 224, 3)]
vgg19 | input: | (None, 224, 224, 3)
Functional | output: | (None, 512)
dense | input: | (None, 512)
Dense | output: | (None, 512)
dropout | input: | (None, 512)
Dropout | output: | (None, 512)
dense_1 | input: | (None, 512)
Dense | output: | (None, 8)

Figure 15 VGG19 Model

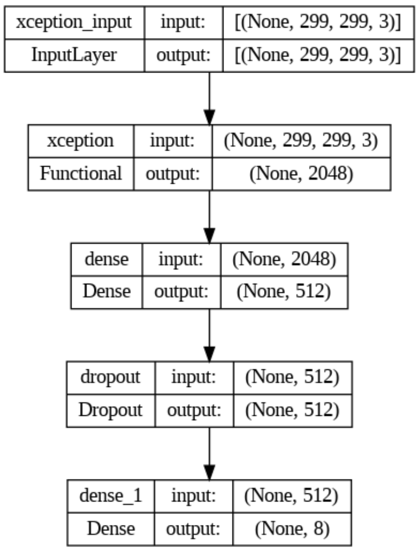

xception_input | input: | [(None, 299, 299, 3)]
InputLayer | output: | [(None, 299, 299, 3)]
xception | input: | (None, 299, 299, 3)
Functional | output: | (None, 2048)
dense | input: | (None, 2048)
Dense | output: | (None, 512)
dropout | input: | (None, 512)
Dropout | output: | (None, 512)
dense_1 | input: | (None, 512)
Dense | output: | (None, 8)

Figure 14 Xception Model

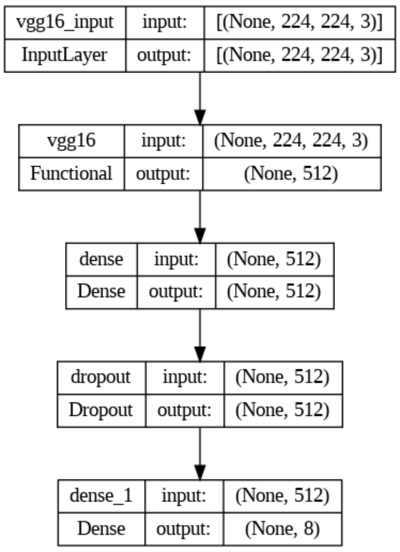

vgg16_input | input: | [(None, 224, 224, 3)]
InputLayer | output: | [(None, 224, 224, 3)]
vgg16 | input: | (None, 224, 224, 3)
Functional | output: | (None, 512)
dense | input: | (None, 512)
Dense | output: | (None, 512)
dropout | input: | (None, 512)
Dropout | output: | (None, 512)
dense_1 | input: | (None, 512)
Dense | output: | (None, 8)

Figure 13 VGG16 Model

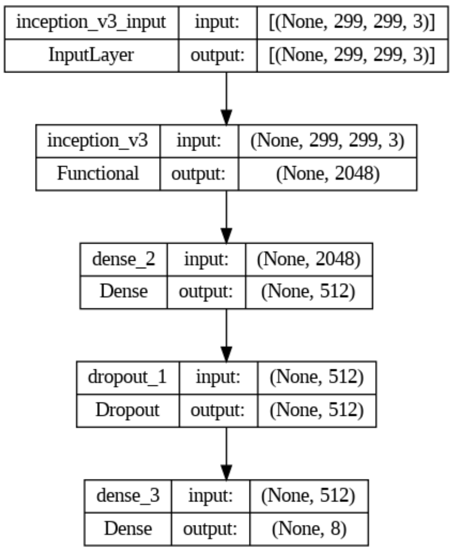

inception_v3_input | input: | [(None, 299, 299, 3)]
InputLayer | output: | [(None, 299, 299, 3)]
inception_v3 | input: | (None, 299, 299, 3)
Functional | output: | (None, 2048)
dense_2 | input: | (None, 2048)
Dense | output: | (None, 512)
dropout_1 | input: | (None, 512)
Dropout | output: | (None, 512)
dense_3 | input: | (None, 512)
Dense | output: | (None, 8)

Figure 17 InceptionV3 Model

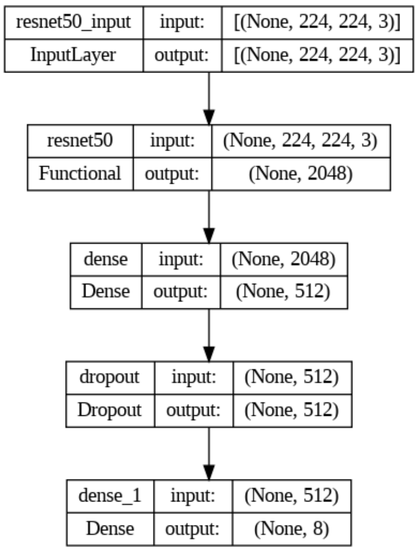

resnet50_input | input: | [(None, 224, 224, 3)]
InputLayer | output: | [(None, 224, 224, 3)]
resnet50 | input: | (None, 224, 224, 3)
Functional | output: | (None, 2048)
dense | input: | (None, 2048)
Dense | output: | (None, 512)
dropout | input: | (None, 512)
Dropout | output: | (None, 512)
dense_1 | input: | (None, 512)
Dense | output: | (None, 8)

Figure 16 ResNet50 Model

6.1.4 Training of the models

This section outlines the training process of the models under study. The training was conducted in two distinct phases: the base implementation phase and the sampling implementation phase. Both phases utilised specific hyperparameter configurations as detailed in Table 2 and Table 3, respectively.

VGG16

Figure 18 represents the base model accuracy and loss graphs for VGG16, the accuracy graph illustrates the model’s training and validation accuracy converging as the number of epochs increases, detailing a stable learning process. The loss graph, however, shows that the training loss decreases significantly over epochs, but the validation loss decreases to a lesser extent and starts to plateau. This means that the model beginning to overfit the training data, and overfitting means that the model has learnt the training data too well, this was expected in the base classifier model since there is no class balancing to penalise the over-representation of the majority class.

As a result, I used a combination of oversampling and under-sampling for the sampling VGG16 model, to prevent overfitting by resolving the class imbalance. However, as illustrated in Figure 19, the generalisation of the model to unseen becomes worse as the training accuracy was consistently higher than the validation accuracy in the accuracy graph. The loss graph displays a similar trend for training loss as the base model, but the validation loss is higher throughout the training process compared to the base model. This elevated validation loss might be due to the model adopting representations that are overly specific to the resampled dataset and not as applicable to the broader, original data distribution.

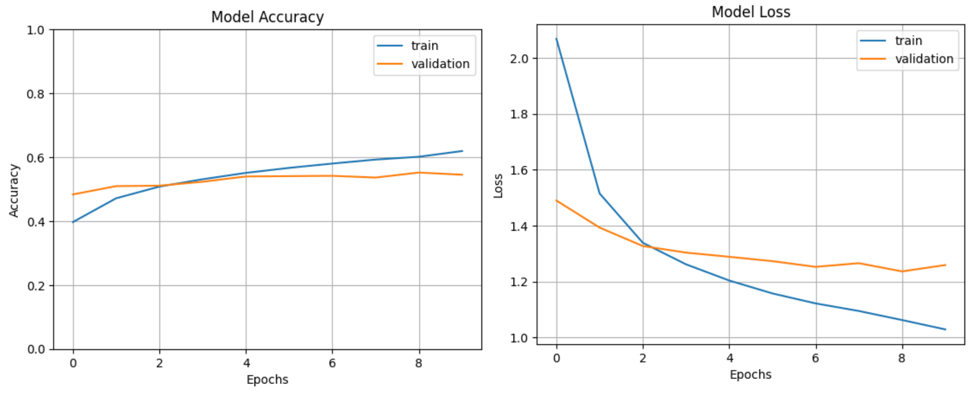


Figure 18 VGG16 Base Model Accuracy and Loss Graphs

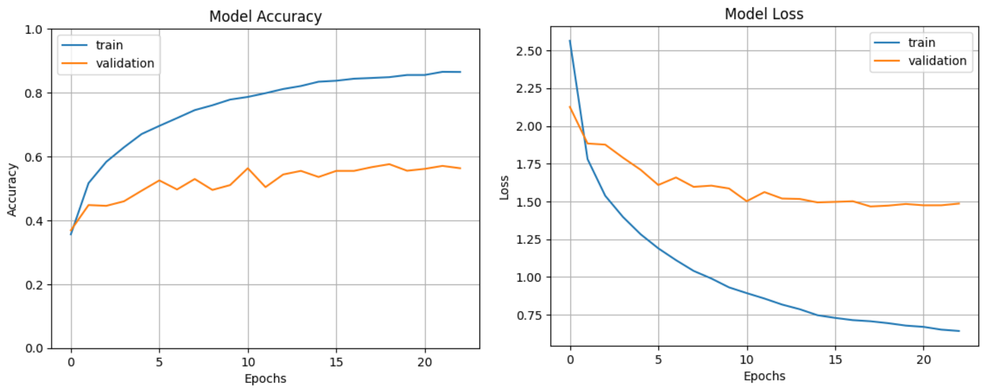


Figure 19 VGG16 Sampling Model Accuracy and Loss Graphs

VGG19

For the base models, the VGG19 demonstrates a convergence of training and validation accuracies like that of the VGG16, suggesting an alike capability in learning from the training set. Notably, the model loss for VGG19, as depicted in Figure 20, exhibits a sharper decrease in validation loss and presents less fluctuation compared to the VGG16 model. This sharper and smoother decline in loss may be attributed to the VGG19 architecture's additional layers, which could have provided it with an enhanced ability to learn patterns within the training data more swiftly.

When comparing the sampling models, the VGG19 (Figure 21) requires more epochs to display similar trends in both the loss and accuracy graphs as seen for the VGG16. This observation suggests that while the VGG19 may benefit from its deeper architecture in terms of learning efficiency, it also necessitates a longer training period to fully leverage this advantage when class balancing techniques are applied.

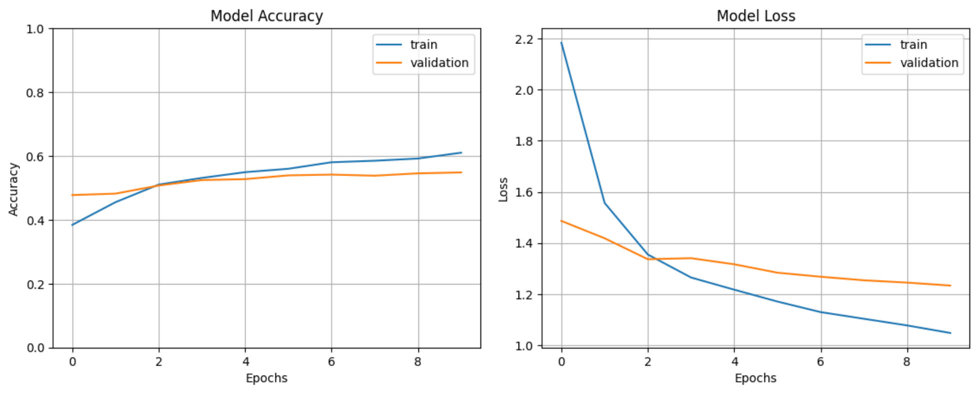


Figure 20 VGG19 Base Model Accuracy and Loss Graphs

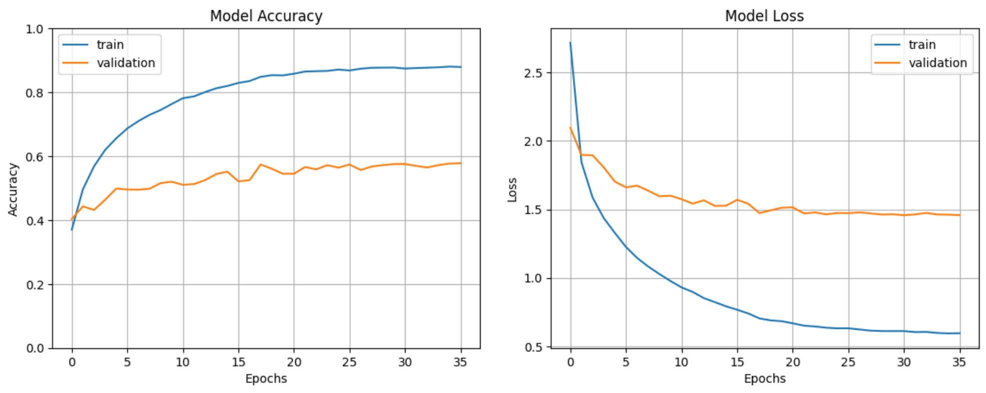


Figure 21 VGG19 Sampling Model Accuracy and Loss Graphs

ResNet50

For the base model, the accuracy graph of ResNet50 (Figure 22) exhibits a consistent improvement in training accuracy, whereas the validation accuracy plateaus early and remains comparatively lower, suggesting a greater degree of overfitting than that observed in the VGG models. The loss graph further corroborates this, with a more pronounced and rapid decrease in training loss, while the validation loss diminishes at a notably slower pace, indicating substantial overfitting.

This discrepancy in learning dynamics between ResNet50 and the VGG models may stem from the use of a distinct preprocessing function tailored to the pre-trained version of ResNet50. Such preprocessing adjustments can significantly affect the model's ability to generalise, as they directly influence the representation of input data that the model learns from.

With the application of oversampling techniques, as presented in Figure 23, the ResNet50 model continues to exhibit substantial overfitting, showing no significant improvement in the alignment between training and validation performance metrics. Although the model was trained for fewer epochs compared to the VGG19, this advantage is negated by the wider gap between the training and validation accuracy and loss, which indicates poorer generalisation to unseen data.

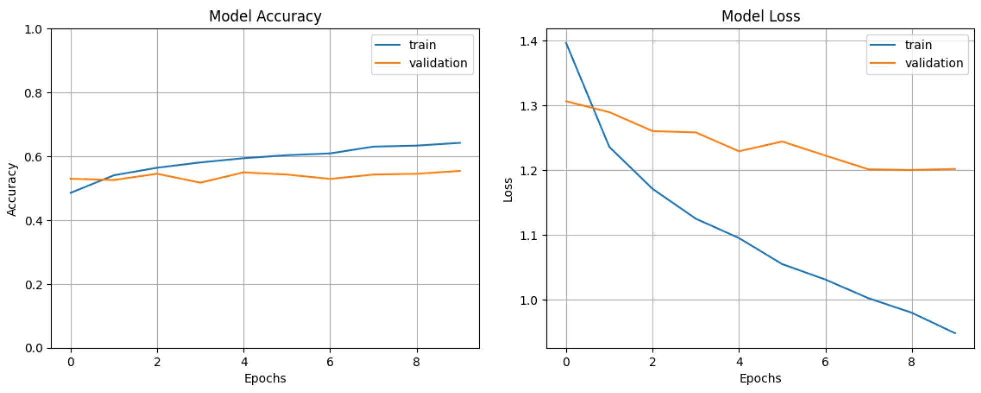


Figure 22 ResNet50 Base Model Accuracy and Loss Graphs

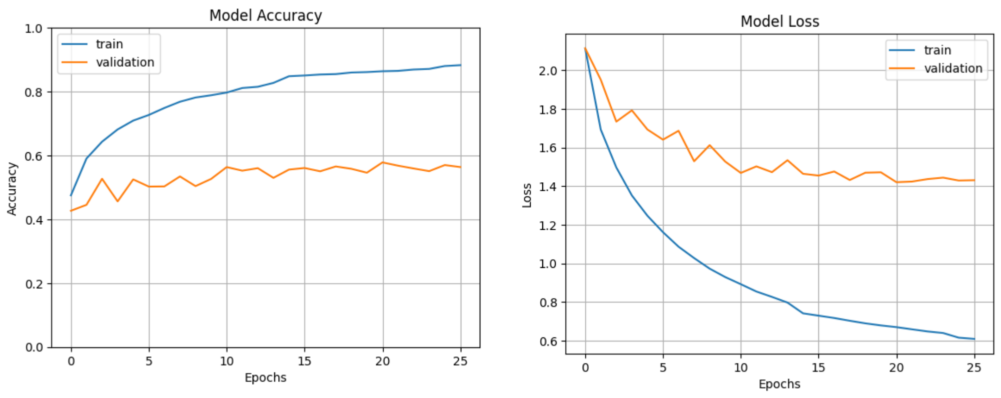


Figure 23 ResNet50 Sampling Model Accuracy and Loss Graphs

Xception

In the base model evaluation, Xception exhibits performance parallel to that of ResNet50, as reflected in the accuracy and loss graphs of Figure 24. It's noteworthy that the image preprocessing for Xception was distinct, involving resizing to [299,299] pixels rather than [224,224], along with a different preprocessing function. These divergent preprocessing steps could potentially contribute to the observed generalisation challenges.

For the sampling model, Xception mirrors the trends seen with ResNet50. A key distinction, however, lies in the training duration; Xception's model required 30 epochs before the training was halted by an early stopping callback. This suggests that while Xception may have similar learning patterns, the extended training period indicates a persistent struggle to converge, which ultimately triggered the early stopping mechanism.

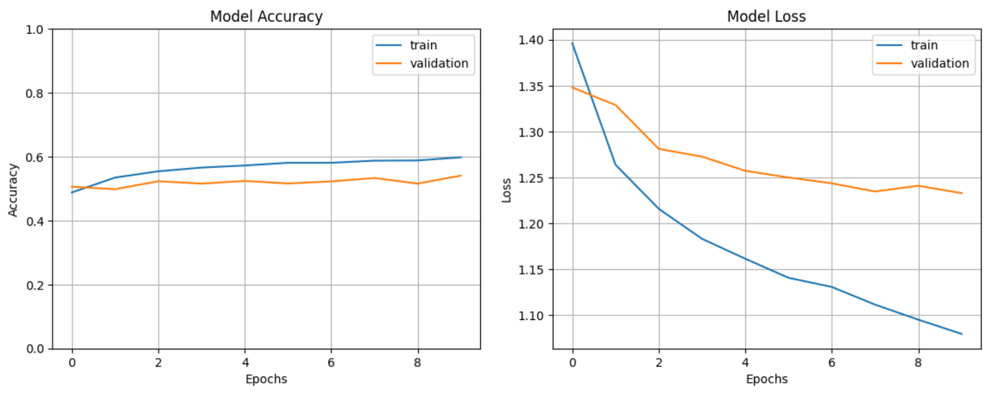


Figure 24 Xception Base Model Accuracy and Loss Graphs

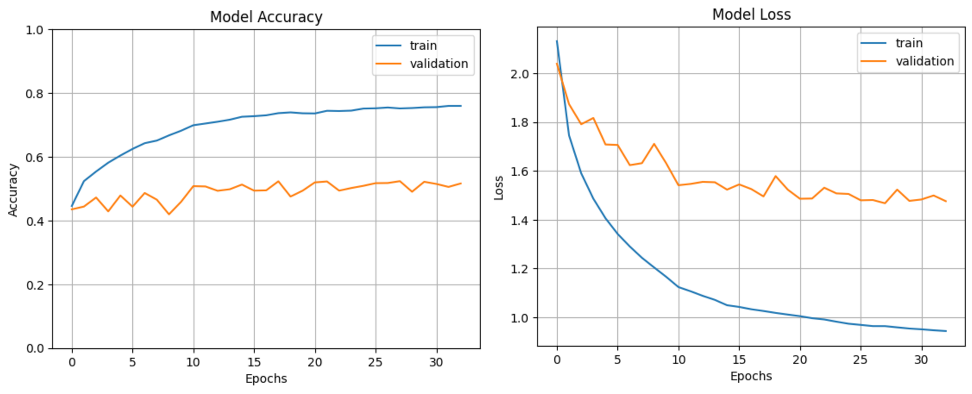


Figure 25 Xception Sampling Model Accuracy and Loss Graphs

InceptionV3

InceptionV3 and Xception models give very similar results for the base and sampling setups. The main difference is that InceptionV3's graphs look a bit more up and down. Since Xception is an updated version of InceptionV3 and they both process images in similar ways, it makes sense that Xception's graphs have smoother lines.

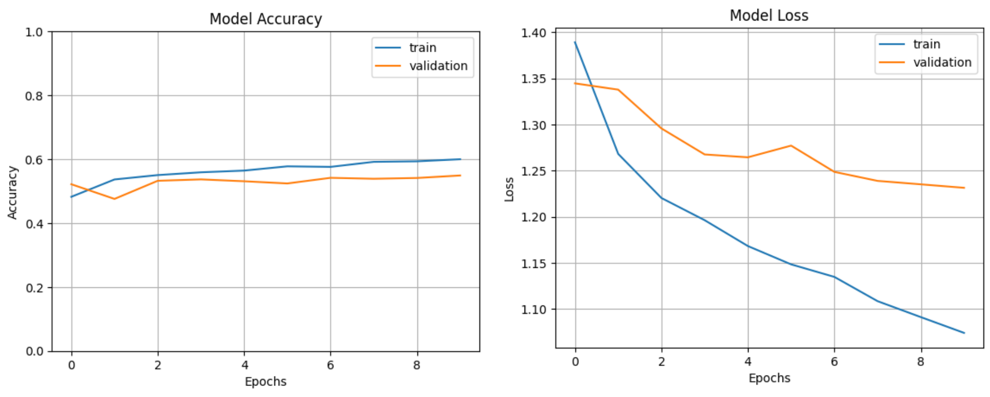


Figure 26 InceptionV3 Base Model Accuracy and Loss Graphs

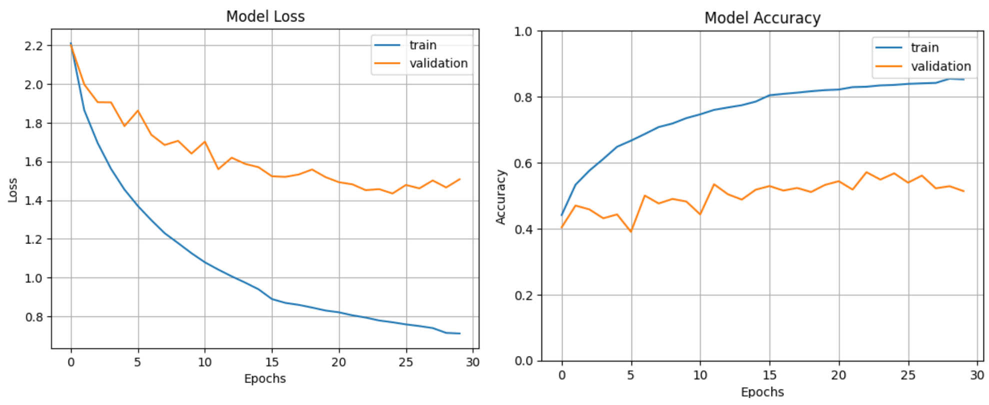


Figure 27 InceptionV3 Sampling Model Accuracy and Loss Graphs

In summary, from this comparative analysis of training, the base model implementation training process, had less overfitting and more generalised plots, so the sampling models will not be further evaluated.

6.2 Evaluation Metrics

To analyse the results provided by the classification models, I utilised evaluation metrics such as accuracy, precision, recall and F1 score. The formulas of these evaluation metrics are shown below:

- Accuracy = (TP+TN) / (TP+TN+FP+FN) (1)
- Precision = TP / (TP+FP) (2)
- Recall = TP / (TP+FN) (3)
- F1 = 2 * Precision * Recall / (Precision + Recall) (4)
- Balanced Accuracy Score= $(1/N)\sum^{N}_{i=1}(TP/(TP+FN))$ (5)

Here,

- TP = True Positive (The total number of images that are correctly detected to be positive)
- FP = False Positive (The total number of images that are predicted to be positive but are negative)
- TN = True Negative (The number of images that are accurately predicted to be negative)
- FN = False Negative (The number of images that are incorrectly predicted to be negative)
- N = Number of Classes

The balanced accuracy score evaluates each class individually, considering true positives and false negatives, and averages the accuracies. This method guarantees that every class, no matter how big or small, has an equal impact on the overall metric, resulting in balanced accuracy being a more informative and just measurement for imbalanced datasets.

ROC AUC (Receiver Operating Characteristic Area Under the Curve) is computed to assess the model's capability to differentiate between various classes in a multi-class classification scenario. It is a valuable measure for assessing model effectiveness beyond just accuracy, particularly in datasets where classes are unevenly distributed. The model's ability to predict 0 classes as 0 and 1 classes as 1 improves as the AUC value increases.

6.3 Results Analysis

Table 4 Comparative Analysis of Image Classification Models Implemented

Models in Order of Performance (Descending)	**Model Accuracy on Test Dataset**	**Precision (macro-average)**	**Recall (macro-average)**	**F1-Score (macro-average)**	**Balanced Accuracy Score**	**AUC-ROC Score**
VGG16	61.14%	0.68	0.51	0.56	51.39%	84.73%
VGG19	58.05%	0.60	0.50	0.53	49.91%	84.46%
ResNet50	56.88%	0.65	0.43	0.47	43.41%	83.44%
Xception	58.41%	0.65	0.46	0.50	45.63%	83.32%
InceptionV3	54.65%	0.63	0.39	0.41	39.12%	80.48%

The best-performing model is VGG16, as it demonstrates superior performance across all evaluated metrics, it achieves the highest Model Accuracy (61.14%), Balanced Accuracy Score (51.39%), and Area Under the Receiver Operating Characteristic Curve (AUC-ROC) score (84.73%), indicating not only its ability to correctly classify a higher percentage of the test dataset but also its robustness in handling imbalanced classes and in distinguishing between class labels. VGG19 follows closely, showing similar strengths, particularly in its ability to differentiate between classes as evidenced by its AUC-ROC score (84.46%). Xception and ResNet50 present competitive, though slightly inferior, performances, with challenges in balanced accuracy, which suggests difficulties in inequitable performance across classes. InceptionV3, however, falls behind the others, struggling significantly across all metrics—most notably in balanced accuracy (39.12%) and AUC-ROC (80.48%), which hints at substantial difficulties both in dealing with class imbalance and in distinguishing between class labels effectively.

Table 5 VGG16 Classification Report

Class	**Precision**	**Recall**	**F1-score**	**Support**
Normal	0.61	0.79	0.69	1108
Diabetes	0.62	0.57	0.59	650
Glaucoma	0.60	0.49	0.54	114
Cataract	0.69	0.62	0.66	124
AMD	0.76	0.45	0.57	100
Hypertension	1.00	0.25	0.40	56
Myopia	0.83	0.77	0.80	100
Other diseases	0.32	0.17	0.22	306
accuracy			**0.61**	2558
macro average	0.68	0.51	0.56	2558
weighted average	0.61	0.61	0.59	2558

From table 5, the VGG16 classification report for various classes provides insight into which classes the model was most and least effective at predicting. The performance metrics to consider include Precision, Recall, and F1-score. Precision measures the proportion of correctly identified positive instances among all instances predicted as positive. Recall, also known as sensitivity, measures the proportion of actual positives correctly identified. The F1-score is the harmonic mean of precision and recall, offering a balance between the two, especially in cases of class imbalance.

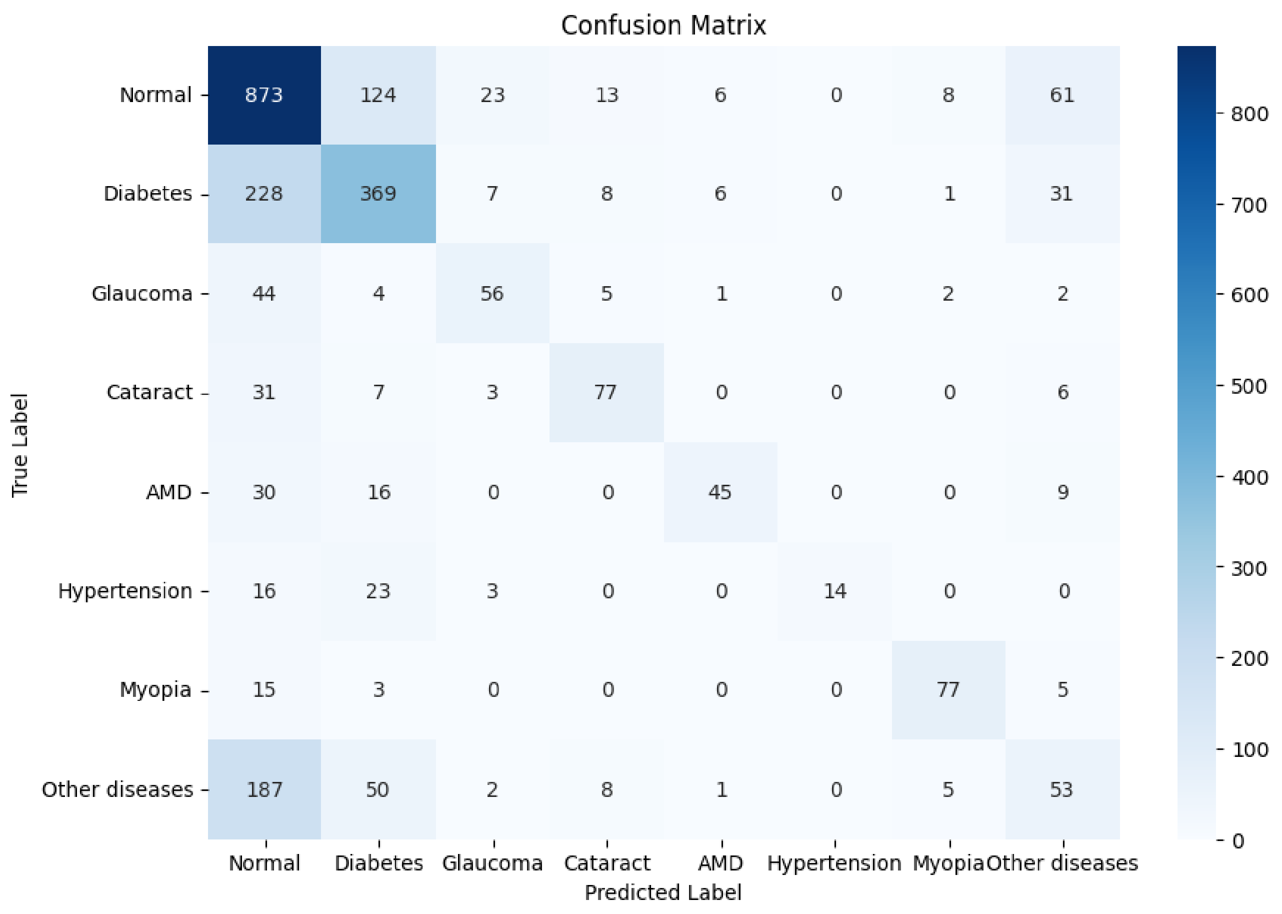
The best performing class for prediction is Myopia, as it shows the highest F1-score of 0.80, indicating a strong balance between precision (0.83) and recall (0.77). The high precision suggests that when the model predicts an image as Myopia, it is very likely correct.

Similarly, the high recall indicates that the model is good at identifying most of the Myopia cases in the dataset.

The worst performing class is other diseases with an F1-score of 0.22, this class has the weakest performance among all classes. The low precision (0.32) and recall (0.17) suggest that the model struggles significantly with correctly identifying and predicting cases of other diseases. This could indicate that the features of other diseases are not as distinct or that there is significant overlap with features of other classes, leading to confusion.

Despite Hypertension’s perfect precision of 1.00, the recall is only 0.25, resulting in an F1-score of 0.40. This suggests that while the model is highly accurate when it predicts Hypertension, it fails to identify a large portion of actual Hypertension cases, possibly due to the class's lower representation in the dataset or its features being common with other conditions.

Age-related Macular Degeneration (AMD) has the highest precision of 0.76 after Hypertension but suffers from a lower recall (0.45), indicating that the model, although very reliable in its AMD predictions, misses more than half of the actual AMD cases.



	Normal	Diabetes	Glaucoma	Cataract	AMD	Hypertension	Myopia	Other diseases
Normal	873	124	23	13	6	0	8	61
Diabetes	228	369	7	8	6	0	1	31
Glaucoma	44	4	56	5	1	0	2	2
Cataract	31	7	3	77	0	0	0	6
AMD	30	16	0	0	45	0	0	9
Hypertension	16	23	3	0	0	14	0	0
Myopia	15	3	0	0	0	0	77	5
Other diseases	187	50	2	8	1	0	5	53

Figure 28 VGG16 Confusion Matrix

Confusion Matrix of the VGG16 model details the counts of actual versus predicted labels, this helps to identify where the model is making correct predictions and where it is confused. The model frequently confuses 'Other diseases' with 'Normal', suggesting that the features of 'Other diseases' may not be distinct enough, or that the model requires better feature extraction techniques to distinguish this class. The confusion matrix validates that the model is not 100% accurate and could benefit from extra data for the other disease classes.

6.4 Visualisation of XAI Results

Appendix 1. All implementations of the explainability methods were conducted using readily available applications, with the code and resources accessible via my GitHub repository, detailed in Appendix 1.

Following the evaluation results discussed previously, various XAI techniques were applied to the VGG16 model. For each method, I ensured consistency by using the same image data and keeping key parameters, such as ‘num_samples’ for LIME and ‘max_evals’ for SHAP, constant. Each implementation was executed twice to yield two sets of results, allowing for a comparative assessment of reproducibility. Augmented image results were successfully generated only for Grad-CAM and SHAP. The ensuing discussion will juxtapose these outcomes, providing insights into the explanatory power of the methods for a "Normal" class image from the test set. This comparison will be structured around the criteria of consistency, fidelity, sensitivity, and clinical relevance as established in section 2.2.2, supplemented by a dedicated table for each model's evaluation on these fronts.

6.4.1 LIME

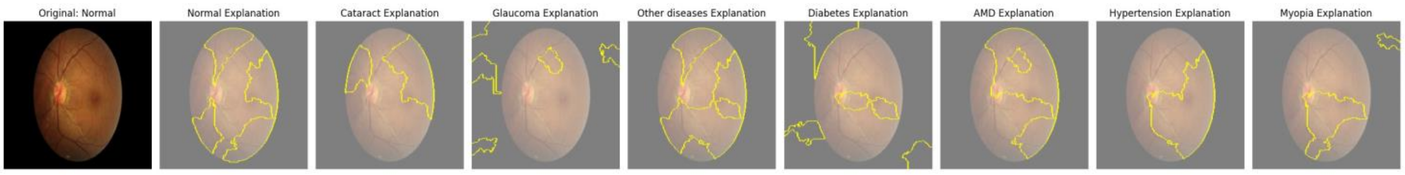


Figure 30 Lime for Normal Run 1

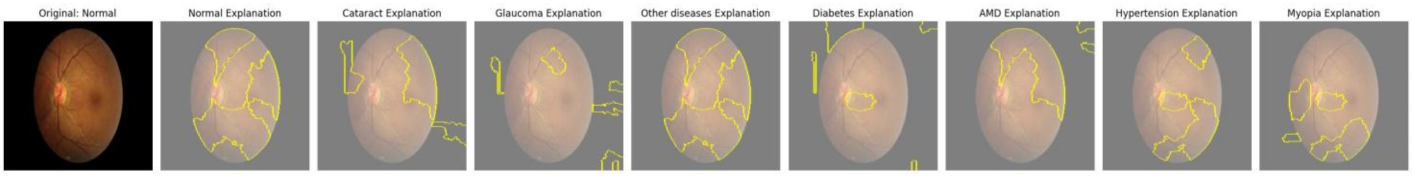


Figure 29 Lime for Normal Run 2

In my LIME implementation, I used segmentation masks derived from super pixel values to generate explanations for the 'Normal' test images shown in Figures 29 and 30. The two separate runs of the explainer model yielded different results for the true class predictions. Although the general patterns might appear similar at first glance, noticeable differences between the runs suggest a baseline level of consistency in the model. The fidelity of LIME is notable, demonstrating reliable performance in correctly identifying the 'Normal' class. According to the VGG16 classification report, the model achieves a precision of 0.61 and a recall of 0.79 for the 'Normal' class, indicating reasonable accuracy. However, alongside accurate explanations, some incorrect ones were also generated. For instance, explanations intended for the 'Other Disease' class were consistently applied to 'Normal' class images in both runs, despite the images not originating from the 'Other Disease' class. Unfortunately, I was unable to generate explanations for augmented images using LIME, leaving the tool's responsiveness to changes in input data untested. Additionally, without access to a clinical expert, the clinical relevance of the explanations provided by the explainer models could not be assessed.

6.4.2 Grad-CAM

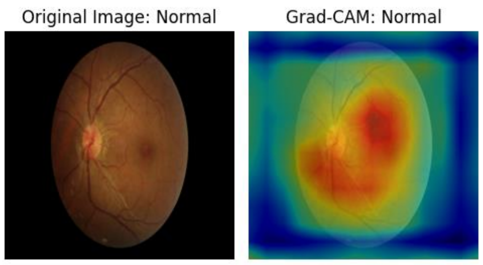


Figure 32 Grad-CAM for Normal Run 1

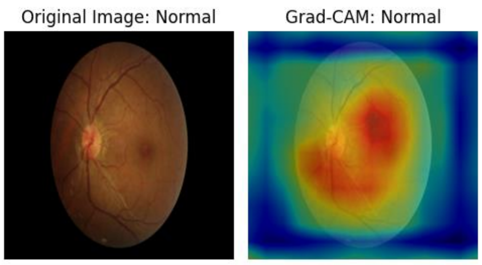


Figure 31 Grad-CAM for Normal Run 2

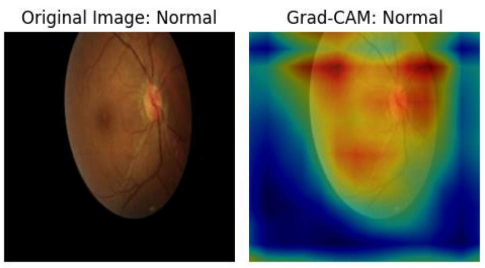


Figure 33 Grad-CAM for Augmented Normal

In the Grad-CAM implementation, it was not possible to produce explanations for multiple classes within a single image, like I was able to for the other explainability tools. Nonetheless, the Grad-CAM visualisations for the 'Normal' class showed a high degree of visual consistency and reproducibility across different runs and even with augmented images, indicating a robust model performance in these specific cases. The visual consistency suggests that Grad-CAM has good fidelity in highlighting the regions of interest that the model is focusing on for its predictions. However, since Grad-CAM was not able to provide a diverse range of class explanations for a given image, this poses a limitation to a fully comprehensive evaluation, particularly in understanding how the model might be handling potential false positives. This aspect is especially important when dealing with medical images where the distinction between classes may be subtle yet clinically significant.

6.4.3 SHAP

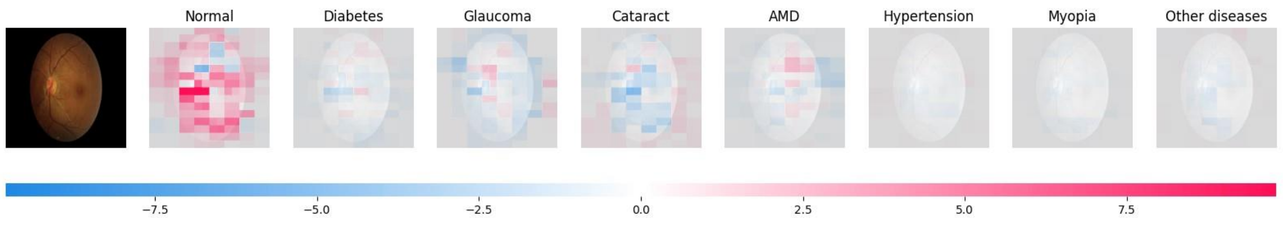


Figure 34 SHAP for Normal Run 1

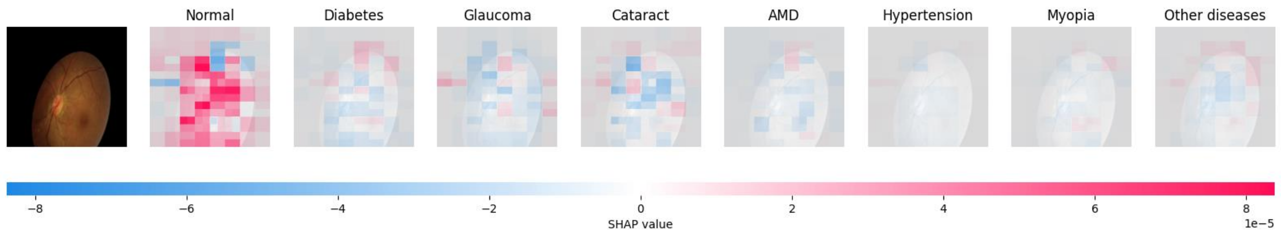


Figure 36 SHAP for Augmented Normal

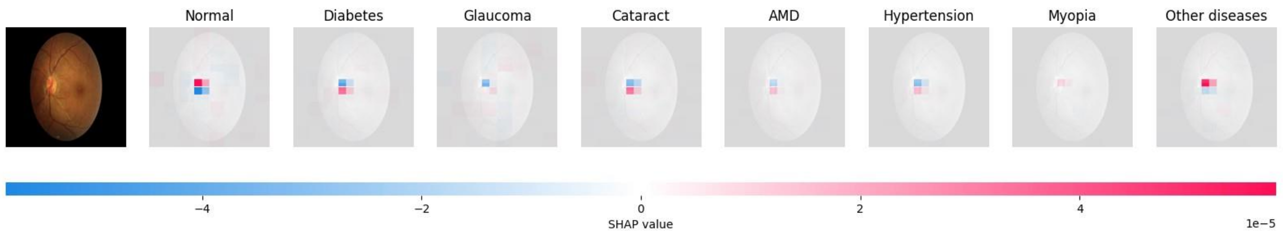


Figure 35 SHAP for Normal Run 2

For the implementation of SHAP, a Partitioner explainer was used, which is suitable to explain images. The explainer partitions the input features into groups that can be considered like super pixels used in LIME. These group of pixels are then used to compute SHAP values, providing an explanation for the model’s predictions by estimating the impact of each group on the output.

Among the three examined explainability techniques, SHAP yields the most visually striking and discernible results, with areas in red denoting the features most influential in the decision-making process. The reproducibility of the explanations generated by SHAP is notable, indicating that the method is robust to changes in the data.

Fidelity-wise, SHAP stands out by providing the most compelling explanations for the true class of the input image, particularly as seen in Figure 26 and Figure 24.

However, the performance of SHAP is not entirely consistent, as evidenced by Figure 25 where it falls short. The inconsistency observed in Figure 25 may represent a false positive since SHAP otherwise demonstrates effective explanations for the "Normal" class. This variation underlines the importance of a thorough investigation into the reliability of SHAP across a wider range of examples to fully ascertain its explanatory power and limitations.

7 Evaluation

This paper set out with clear objectives: to develop a computer-aided diagnostic system for ocular diseases and to enhance the transparency of its predictions through the application of XAI techniques. In reflecting upon these objectives, it can be affirmed that the paper has achieved its foundational goal of constructing a multi-class classification system and has explored the interpretability of the model predictions using various explainability techniques.

7.1 Achievement of Objectives

Through methodical application and comparative analysis, VGG16 emerged as the most accurate CNN model with a 61.14% accuracy rate. In terms of XAI, Grad-CAM demonstrated the highest degree of visual consistency, SHAP provided the most comprehensible explanations, and LIME displayed potential limitations, particularly in its application to augmented images.

7.2 Comparison to existing literature

However, comparing the achieved accuracy with existing literature reveals a notable discrepancy. For instance, Dipu, Shohan, and Salam (2021) reported a significantly higher accuracy rate of 97.23% on the same ODIR-5K dataset using a transfer learning VGG16 model. This disparity suggests that while the methodologies employed were conceptually robust, the execution or specific conditions of my study might have impacted performance adversely.

Ideally, the explainability implementation would benefit from a model with higher accuracy, as the modest results obtained limit the effectiveness of such features. Nevertheless, the best achievable outcome was realised, given that my attempts to mitigate class imbalance did not substantially enhance accuracy and led to issues with model overfitting.

After the initial model setup, I adjusted several hyperparameters, such as increasing the batch size from 16 to 128 and varying the dropout rate between 0.2, 0.5, and 0.8. These modifications are thoroughly documented in the commit history of my GitHub repository. Additionally, I varied the number of epochs, starting from 10 and extending up to 80. While random sampling improved the balanced accuracy score and recall for certain classes, overall accuracy remained low.

The 'Other diseases' class performed poorly, likely due to its heterogeneous features. Unlike other studies that removed image artifacts to refine class labels, this approach was not adopted in my work, which might have contributed to better outcomes if implemented. I also explored k-fold cross-validation to address class imbalance. This technique partitions the dataset into 'k' consecutive folds, each fold then used once as a validation while the k-

1 remaining folds form the training set. I only implemented this for ResNet50 and VGG16 to test if I would yield a higher accuracy, but I was not able to the test the cross-validation model on the test data to receive predictions.

Adding the appropriate layers to pretrained models was also a tedious task. As a model that is too complex for the dataset may lead to overfitting, where it performs well on training data but poorly on unseen data. Conversely, a model that is not complex enough might underperform due to insufficient capacity to capture the nuances of the data. Striking the right balance was crucial to maximise performance without compromising the model's ability to generalise to new data. This involved experimenting with different configurations and layers to find an optimal structure that enhances the model's predictive accuracy while maintaining its robustness against overfitting.

7.3 Limitations

The primary limitation I faced was the low accuracy of the image classification model, which did not improve despite applying class balancing strategies. Another significant challenge was the lack of computational resources, which notably impacted the accuracy as extensive hyperparameter tuning could have potentially enhanced the model’s performance. In this context, I considered using Keras Tuner, an automated hyperparameter tuning tool that optimises model configurations within specified limits to achieve the best possible performances. Keras Tuner systematically tests a set of hyperparameters, evaluating their effectiveness using a defined metric. However, when I attempted to use it for tuning ResNet50, the process demanded extensive computational power and ran for an extended period, straining the resources I was funding out of pocket.

Time management also proved problematic as I spent a considerable amount of time coding until the very end, which prevented me from recognising and addressing mistakes early in the process. Additionally, I encountered issues with data leakage among the train, test, and validation datasets, which initially resulted in misleadingly high accuracies. Fortunately, I identified and rectified this error in time. Another oversight was my failure to run LIME on augmented images to assess the explainability method’s responsiveness to changes in making predictions. This would have provided deeper insights into how well the explainability techniques adapted to new data scenarios, thereby enhancing the robustness and reliability of the diagnostic predictions.

8 Project Management

The section covers insights on the planning and management of this project.

8.1 SMART Objectives:

1. **Conduct a Comprehensive Literature Review**

- By the middle of December, identify and summarise at least 4 powerful deep learning architectures for multiclass classification in medical imaging through a comprehensive literature review, ensuring coverage of recent advancements.

2. **Conduct Dataset Preprocessing**

- By the end of January, implement class balancing techniques for the study dataset to address imbalances, ensuring equal representation of each eye disease category. Additionally, apply image augmentation where necessary to enhance dataset diversity and prepare it for deep learning model training.

3. **Apply Deep Learning Architectures to Study Dataset**

- Achieve a baseline accuracy of 90% by the end of February for each deep learning architecture identified in Objective 1 through thorough implementation and evaluation of the designated study dataset for eye disease classification.

4. **Conduct a Comparative Analysis Between the Deep Learning Architectures**

- Document and analyse accuracy, precision, recall, and F1-score for each optimised model by the beginning of March, conducting a comparative analysis to identify and document the models that outperform others in the context of medical image classification.

5. **Explore Explainable AI Algorithms**

- Apply LIME, Grad-CAM, and SHAP to the best-performing model and conduct a comparative analysis to evaluate their effectiveness in improving the interpretability and explainability of the models by the middle of March.

6. **Investigate the Impact of Explainable AI Algorithms and Conduct a Comparative Analysis**

- Document insights into how the XAI techniques used contribute to improving the understanding of model decisions, focusing on enhancing trustworthiness and facilitating informed decision-making in a medical context by the end of March.

7. **Write findings into the Final Report.**

- Document findings into the Final Report by the 22nd of April complete all sections and analyse all outcomes and limitations faced.

8.2 Project Infrastructure:

1. **Programming Language -** Python, due to its extensive libraries and frameworks for machine learning and image processing.
2. **Deep Learning Framework –** TensorFlow to implement and train the deep learning model. The framework offers a wide range of pre-built functions and modules for neural network development.
3. **Development Environment –** Google Colab.
4. **Version Control** – Git, to track changes in the codebase.
5. **Data Storage** – On personal laptop and OneDrive.
6. **Backup and Recovery -** Establish a backup and recovery strategy for critical data and code, minimising the risk of data loss or system downtime.

8.3 Project Characteristics:

Accuracy Requirements - A key characteristic is the high accuracy target of 90% for the detection and classification of eye diseases. This is based on what previous literature was able to achieve. This requirement emphasises the need for rigorous optimisation and evaluation of deep learning models.

Complexity - The project involves the complexity associated with deep learning, requiring the implementation of multiple deep learning architectures, optimisation strategies, and explainable AI algorithms. The medical context adds an extra layer of complexity, considering the need for accuracy and interpretability in healthcare applications.

Continuous Improvement - The project is iterative, involving continuous improvement through phases such as literature review, model development, optimisation, and evaluation. Regular assessments and adjustments are integral to achieving the desired outcomes.

Multi-disciplinary Nature - The project lies at the intersection of computer science, machine learning, and healthcare.

Scope - The project focuses on the development of a small-scale computer-aided diagnostic system for medical image analysis, specifically targeting the detection and classification of eye diseases in fundus images.

Data Sensitivity - The use of medical imaging data involves sensitivity and compliance with privacy, and security regulations, such as HIPAA. The project must prioritise data protection and ethical considerations throughout the development process.

Time Sensitivity - The project has a defined timeframe from January to the end of April, requiring efficient planning and execution to meet the project objectives within the specified deadline.

8.4 Product Flow Diagram:

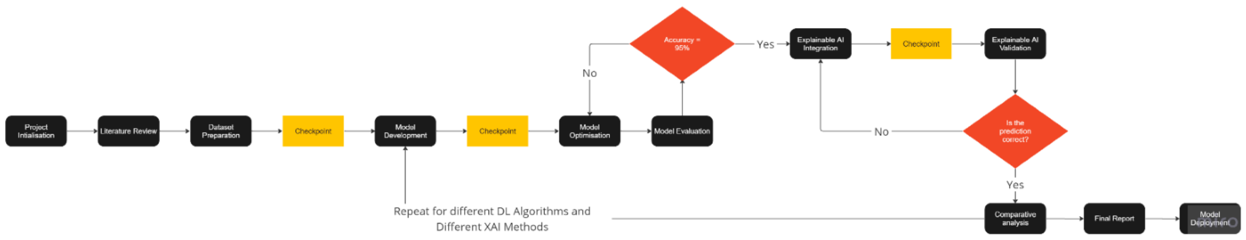
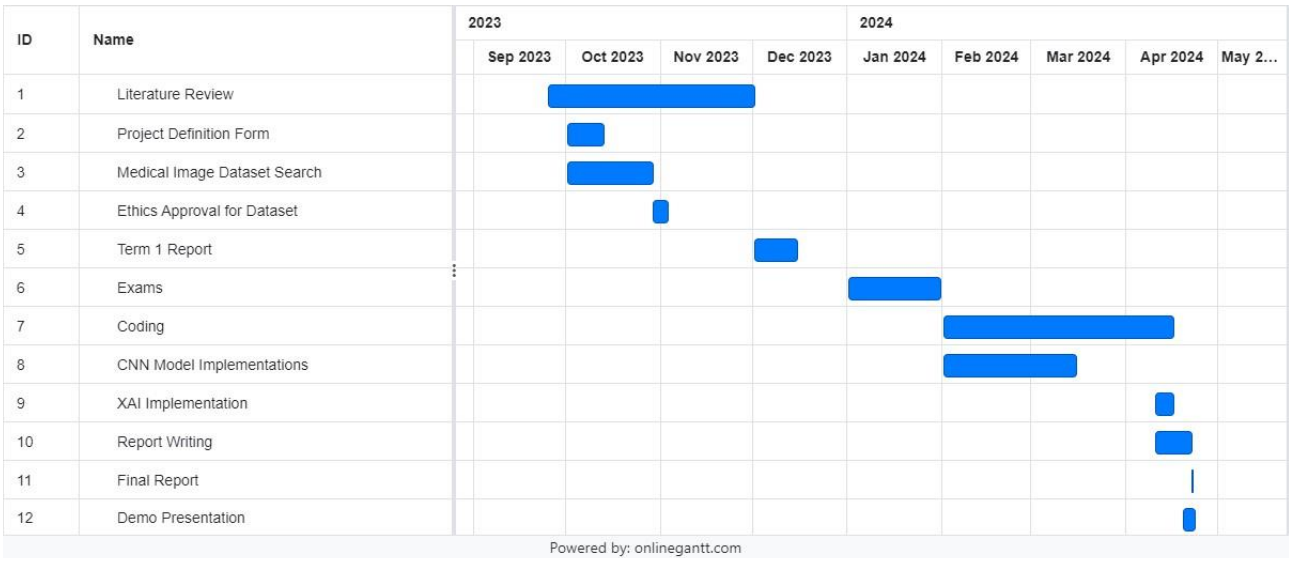


Figure 37 Product Flow Diagram

<u>**Activities Cost Estimation – Gantt Chart**</u>



ID	Name
1	Literature Review
2	Project Definition Form
3	Medical Image Dataset Search
4	Ethics Approval for Dataset
5	Term 1 Report
6	Exams
7	Coding
8	CNN Model Implementations
9	XAI Implementation
10	Report Writing
11	Final Report
12	Demo Presentation

Figure 38 Gantt Chart

Activity Risk Assessment

Activity 1: Literature Review:

- **Risk**: Incomplete or outdated literature.
- **Mitigation**: Regularly update the literature review, consult with experts, and verify the relevance of sources.

Activity 2: Dataset Preparation:

- **Risk**: Insufficient or poor-quality data.
- **Mitigation**: Thoroughly validate and preprocess data, perform data augmentation, and consider incorporating external datasets.

Activity 3: Model Development:

- **Risk**: Difficulty in selecting suitable architectures.
- **Mitigation**: Consult with experts, experiment with different architectures, and conduct pilot studies to identify the most effective models.

Activity 4: Model Optimisation:

- **Risk**: Overfitting or underfitting during optimisation.
- **Mitigation**: Regularly validate models on separate datasets, adjust hyperparameters cautiously, and use techniques like regularisation.

Activity 5: Model Evaluation:

- **Risk**: Misinterpretation of evaluation metrics.
- **Mitigation**: Clearly define evaluation metrics, seek expert input, and document the rationale behind metric choices.

Activity 6: Explainable AI Integration:

- **Risk**: Poor integration leads to inaccurate interpretability.
- **Mitigation**: Validate interpretability tools separately, ensure compatibility, and seek feedback from domain experts.

Activity 7: Comparative Analysis:

- **Risk**: Biases in model comparison.
- **Mitigation**: Use standardised evaluation metrics, conduct sensitivity analyses, and consider different perspectives for a comprehensive comparison.

Activity 8: Model Deployment:

- **Risk**: Technical challenges in integrating models into the diagnostic system.
- **Mitigation**: Perform thorough testing, use containerisation for consistency, and have a rollback plan in case of deployment issues.

Activity 9: Documentation & Reporting:

- **Risk**: Incomplete or unclear documentation.
- **Mitigation**: Establish documentation standards, conduct regular reviews, and seek feedback from project stakeholders

Software Development Lifecycle: Agile

I adopted an agile approach for developing the classification models, which is particularly beneficial for projects with tight deadlines, high complexity, and a need for innovative solutions. From the outset, I established the key evaluation metrics for these models, setting a clear framework for their design and implementation. These metrics included accuracy, loss, precision, recall, and AUC-ROC, which are critical for gauging the performance of the models.

I began the project with a preliminary phase, "Sprint Zero," dedicated to setting up the necessary infrastructure and tools for model development. This setup involved establishing development environments, data pipelines, and continuous integration/continuous deployment (CI/CD) systems. Additionally, this phase allowed for exploratory data analysis to gain insights into the data characteristics and challenges that the CNN models would address.

During each subsequent sprint, I focused on developing and refining CNN models. Starting with simple architectures to establish baselines, I progressively tackled more complex designs and integrated additional data sources. This iterative approach facilitated the testing of various hyperparameters, layer configurations, and activation functions to optimise model performance.

Instead of daily scrum meetings, I held bi-weekly discussions with my project mentor. These sessions were crucial for assessing my progress, troubleshooting implementation issues, and seeking guidance on any challenges encountered.

9 Conclusion and Future Work

This study aimed to develop a transparent, computer-aided diagnostic system for ocular disease recognition. Despite the advanced methodologies applied, the system faced significant challenges in achieving high accuracy, which was further compounded by computational resource limitations. The highest achieved accuracy of the image classification models was modest, reflecting the difficulty of balancing model complexity with performance on a complex dataset like ODIR-5K. Although the use of Explainable AI (XAI) techniques such as LIME, SHAP, and Grad-CAM enhanced the interpretability of the models, the overall effectiveness was constrained by the models' performance.

Limitations included the low accuracy of predictions and the substantial computational resources required for optimal hyperparameter tuning and model training, which were not fully accessible due to budget constraints. These factors hindered the potential of achieving better performance and more robust explanations of the model decisions.

Future Work should focus on:

- Enhancing Model Accuracy: Exploring more sophisticated data augmentation techniques and advanced neural network architectures could potentially improve model accuracy.
- Sourcing External Funding for Computational Resources: Utilising cloud computing platforms with the help of external funding may provide the necessary power for more extensive hyperparameter tuning and training deeper models.
- Extending Explainability: Applying XAI techniques to more complex models or developing new methods tailored to specific medical imaging tasks to improve both transparency and user trust.
- Clinical Validation: Collaborating with medical professionals to validate the model's predictions against clinical outcomes, thus ensuring its utility in practical scenarios.

In conclusion, while the project faced several setbacks primarily due to resource constraints, it laid a foundation for further research into the application of deep learning and explainable AI in medical diagnostics, particularly in the detection and classification of ocular diseases.

10 References

Bhati, A. *et al.* (2023) 'Discriminative kernel convolution network for multi-label ophthalmic disease detection on imbalanced fundus image dataset,' *Computers in Biology and Medicine*, 153, p. 106519. https://doi.org/10.1016/j.compbiomed.2022.106519.

Brownlee, J. (2019) *A gentle introduction to the ImageNet Challenge (ILSVRC)*. https://machinelearningmastery.com/introduction-to-the-imagenet-large-scale-visual-recognition-challenge-ilsvrc/.

Brownlee, J. (2021) *A gentle introduction to object recognition with deep learning*. https://machinelearningmastery.com/object-recognition-with-deep-learning/.

Chollet, F. (2016) *Xception: Deep Learning with Depthwise Separable Convolutions*. https://arxiv.org/abs/1610.02357.

Das, A. and Rad, P. (2020) *Opportunities and Challenges in Explainable Artificial Intelligence (XAI): a survey*. https://arxiv.org/abs/2006.11371.

Dipu, N.M., Shohan, S.A. and Salam, K.M.A. (2021) 'Ocular disease detection using advanced neural network based classification algorithms,' *Asian Journal of Convergence in Technology*, 7(2), pp. 91–99. https://doi.org/10.33130/ajct.2021v07i02.019.

Hassan, M. *et al.* (2023) *Futuristic variations and analysis in Fundus images corresponding to biological traits*. https://arxiv.org/abs/2302.03839.

Hassan, M.U. (2023) *VGG16 – Convolutional Network for Classification and Detection*. https://neurohive.io/en/popular-networks/vgg16/.

He, K. *et al.* (2015) *Deep residual learning for image recognition*. https://arxiv.org/abs/1512.03385.

Jiang, L. *et al.* (2021) 'Opportunities and challenges of artificial intelligence in the medical field: current application, emerging problems, and problem-solving strategies,' *Journal of International Medical Research*, 49(3), p. 030006052110001. https://doi.org/10.1177/03000605211000157.

Khan, M.S. *et al.* (2022) 'Deep Learning for Ocular Disease Recognition: an Inner-Class balance,' *Computational Intelligence and Neuroscience*, 2022, pp. 1–12. https://doi.org/10.1155/2022/5007111.

Kinger, S. and Kulkarni, V. (2022) 'Explainability of Deep Learning-Based System in Health care,' in *Smart innovation, systems and technologies*, pp. 619–633. https://doi.org/10.1007/978-981-16-9447-9_47.

Lundberg, S. and Lee, S.-I. (2017) *A unified approach to interpreting model predictions*. https://arxiv.org/abs/1705.07874.

Mukherjee, S. (2022) 'The annotated RESNet-50 - towards data science,' *Medium*, 18 August. https://towardsdatascience.com/the-annotated-resnet-50-a6c536034758.

Parikh, R.B. and Helmchen, L.A. (2022) 'Paying for artificial intelligence in medicine,' *Npj Digital Medicine*, 5(1). https://doi.org/10.1038/s41746-022-00609-6.

Peking University (2019) *ODIR-2019 - Grand Challenge*. https://odir2019.grand-challenge.org/dataset/ (Accessed: December 6, 2023).

Ribeiro, M.T., Singh, S. and Guestrin, C. (2016) *'Why should I trust you?': explaining the predictions of any classifier*. https://arxiv.org/abs/1602.04938.

Saha, S. (2023) 'A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way,' *Medium*, 8 April. https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53.

Selvaraju, R.R. *et al.* (2019) 'Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization,' *International Journal of Computer Vision*, 128(2), pp. 336–359. https://doi.org/10.1007/s11263-019-01228-7.

Shrikumar, A. *et al.* (2016) *Not just a black box: learning important features through propagating activation differences*. https://arxiv.org/abs/1605.01713.

Simonyan, K. and Zisserman, A. (2014) *Very deep convolutional networks for Large-Scale image recognition*. https://arxiv.org/abs/1409.1556.

Sun, J. *et al.* (2018) 'A comparative study of explainer modules applied to automated skin lesion classification,' *Department of Engineering Science, University of Oxford, UK* [Preprint]. https://ceur-ws.org/Vol-2796/xi-ml-2020_sun.pdf.

Szegedy, C. *et al.* (2015) *Rethinking the inception architecture for computer vision*. https://arxiv.org/abs/1512.00567.

Team, K. (no date) *Keras documentation: Keras 2 API documentation*. https://keras.io/2.16/api/.

Wang, J. *et al.* (2020) 'Multi-Label classification of Fundus images with EfficientNet,' *IEEE Access*, 8, pp. 212499–212508. https://doi.org/10.1109/access.2020.3040275.

Weiss, K.R., Khoshgoftaar, T.M. and Wang, D. (2016) 'A survey of transfer learning,' *Journal of Big Data*, 3(1). https://doi.org/10.1186/s40537-016-0043-6.

XCeption model and depthwise separable convolutions (2019). https://maelfabien.github.io/deeplearning/xception/#what-does-it-look-like.

11 Appendices

11.1 Appendix 1: Code

GitHub Repository Link - https://github.com/dduwa/ocular-disease-recognition

11.2 Appendix 2: Project Diary & Meetings with Supervisor

I created a digital project diary on Trello to keep organised, which includes dated transcripts of my meetings with my supervisor and details of weekly tasks I have assigned myself which have included deadlines and have attachments so that I can keep track of links and the files related to research and the project.

Trello Link: https://trello.com/b/Zr1JGrSA/project-diary-duwa-khan

11.3 Appendix 3: Ethics Declaration Form

Word document accessible via Aston university login - CS3IP Ethics Declaration Form - Duwa Khan.docx