Jimma Institute of Technology

School of Biomedical Engineering

Bioimaging Engineering Graduate Program

Multiple Retinal Disease Classification system from Retinal Images Using Deep Learning

By: Kedir Jundi Aliyi

A Thesis Submitted to the School of Graduate Studies of Jimma Institute of Technology, In Partial Fulfillment of the Requirements for the Degree of Master of Science in Biomedical Imaging.

Main Advisor: Taye Tolu (Ph.D.)

Co-Advisor: Selamawit Haddush (MSc.)

April, 2024

Jimma, Ethiopia

Declaration

I declare this thesis entitled "**Multiple Retinal Disease Classification system from Retinal Images Using Deep Learning**" is my work and that has not been presented for a master of degree in any other university.

Name **Signature**

_____ _____

(Done by)

Date: _____

Approval sheet

This Msc thesis entitled "**Multiple Retinal Disease Classification system from Retinal Images Using Deep Learning**" by **Kedir Jundi** is approved for the Degree of Master of Science in Biomedical Engineering, Bioimaging specialization.

Name	**Signature**
Melkamu Hunegnaw, PhD	
(External Examiner)	
(Internal Examiner)	
Taye Tolu Mekonnen, PhD	
(Main Advisor)	
(Co-Advisor)	
(Chairman)	

Date: _____

Abstract

Retina is a metabolically active part of the eye, in which eye specific and systemic based diseases manifest themselves in it. Retinal imaging with fundus camera is primarily technique to diagnosis common causes of blindness and health problems of the eye, for its safety and cost effectiveness. Automated retinal disease classification to types and sub-types from fundus camera is essential for early detection and management of disease treatment plan, which is less effective and tiresome with the manual methods. Different machine and deep learning methods have been proposed to automate such manual diagnosis procedures. However, the results reported from previously proposed automation techniques are inconsistent, less accurate, and did not consider Ethiopian context data. This thesis proposed the use of deep learning based classification of the four main causes of blindness (diabetic retinopathy, hypertensive retinopathy, glaucoma, and occlusion) to their respective types, as well as the subtypes of diabetic retinopathy and occlusion, using retinal fundus images. The model was developed using the Keras framework, trained, and validated using fundus photography images acquired from the "paper pointed dataset", "the Kaggle online repository" and local data acquired from Blue Vision Clinic. Google Collaborator and Python programming languages were used for model training and graphical user interface development. The ResNet50 based transfer learning model was used, and the result was then compared to benchmark finding. ResNet50 based transfer learning model showed a better result in type and subtype classification. With the proposed method, 97.2%, 96.8%, and 95.3% accuracy and of 93.2%, 92.2% and 93% F1-scores was achieved for disease type, diabetic retinopathy subtypes and occlusion sub-types respectively. These findings highlight deep learning based retinal disease classification as an efficient and accurate alternative to manual methods, contributing to improved disease management and treatment.

Keywords: ***Classification, Deep Learning Model, Retinal Diseases, Transfer Learning.***

Acknowledgement

First and foremost, I would like to express my gratitude to the Almighty God for his unwavering support throughout this research work. I want to extend my heartfelt gratitude to my advisors, Dr. Taye Tolu (Ph.D.) and Selamawit Haddush (M.Sc.), for their efforts in advising, supporting, and sharing expertise with me so that I could complete this task effectively. Furthermore, I would like to express my gratitude to my clinical collaborator, Mr. Biruk Abera (Ophthalmologist) for data collection. In addition to this, I'd like to express my gratitude to Mr. Mohammed Aliy for his helpful counsel and direction throughout this thesis work. Moreover, I want to thank all of the staff of Bio-medical Engineering school, as well as my friends and family for their assistance and motivation.

Contents

List of Figures

List of Tables

List of Acronyms

AI Artificial Intelligence

AV Artery-Vein

BRAO Branch Retinal Artery Occlusion

BRVO Branch Retinal Vein Occlusion

CAD Computer-aided diagnosis

CNN Convolutional Neural Network

CRAO Central Retinal Artery Occlusion

3D Three-Dimensional

2D Two-Dimensional

DME Diabetic Macular Edema

DCNN Deep Convolutional Neural Network

DNN Deep Neural Network

DR Diabetic Retinopathy

FA Fluorescein Angiography

FOV Field of View

GUI Graphical User Interface

HR Hypertensive Retinopathy

OCT Optical Coherence Tomography

OD Optic Disc

RAM Random Access Memory

RELU Rectified Linear Unit

ResNet Residual Network

RPE Retinal Pigment Epithelium

PDR Proliferative Diabetic Retinopathy

USA United States of America

VGG Visual Geometry Group

Chapter 1

Introduction

1.1 Background of the Study

Visual impairment and blindness are major health problems in our world. According to the study made on global data on visual impairment in 2002, the number of visually impaired people is estimated to be 161 million, of which 37 million were blind [1]. In developing countries,females and older age groups share higher numbers [2]. Ethiopia is not an exception to this health problem. Evidence confirms that Ethiopia has the world's highest rates of blindness (1.6%). A study conducted in different parts of the country, such as Merabete North Shoa, Jimma Zone, Bahirdar, and Gamo Gofa Zone, indicates that blindness is a major health problem that needs to be addressed [3, 4], For instance, the prevalence rate found in the study in Merhabette is about 1%. Moreover the following graph in the Figure 1.1 below summarizes common retinal disease statistics based on their frequent occurrence of a disease reported within referenced papers and presented as percentages.

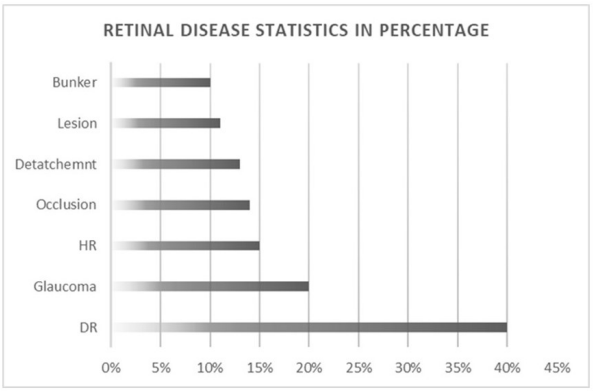


Figure 1.1 : *Blindness and prevalence [5, 6]*

1.1.1 Anatomy and physiology of Eye

The main structures of the eye include the cornea, iris, pupil, lens, retina, macula, optic nerve, and vitreous. The anterior structures, including cornea, iris, pupil and lenses are the front retinal parts and are mainly concerned with incoming light focusing. On the other hand, the retina is a vital structure in the eye's vision system and lines the back interior of the eye. This part of the eye enables the conversion of incoming light into a neural signal for further processing in the visual cortex of the brain for visual recognition and can be seen as an extension of the brain. Retina is a tissue lining and it is more accessible for examination. Moreover, it has been understood that an estimated 80% of all sensory information is thought to be of retinal origin, which indicates the importance of the retina as a gateway to interact with the environment. The optic nerves and macula are two other important parts contained within this retinal structure. The macula is a small central area in the retina that allows you to see fine details clearly. Whereas, as shown on the eye and retinal anatomy picture in Figure 1.2, the optic nerves are the nerve endings that connect the eye to the brain and carry the electrical impulses formed by the retina to the visual cortex of the brain. Additionally, blood vessels are another constituent part of retinal tissue.

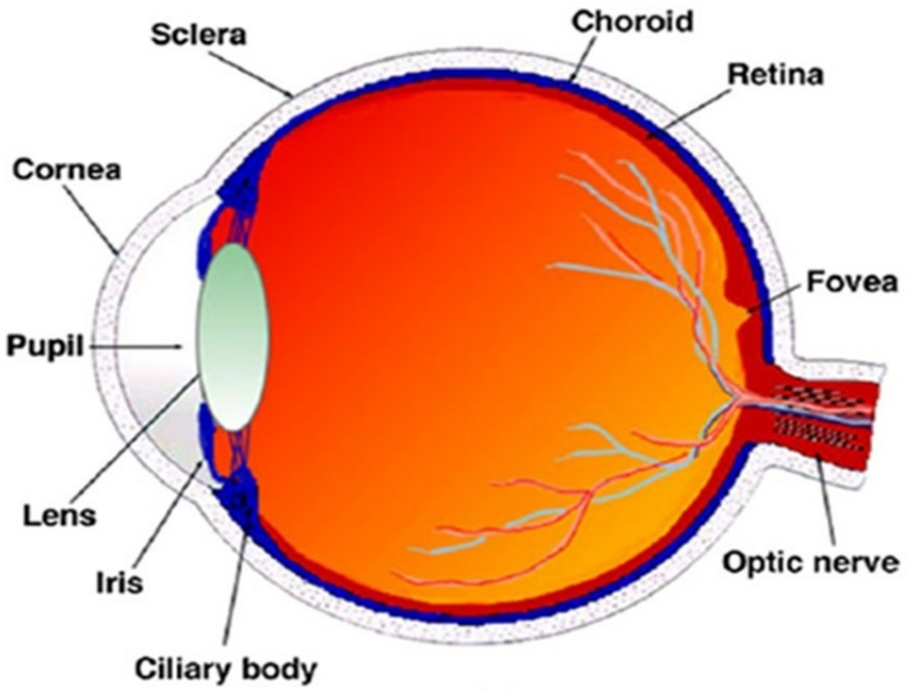


Figure 1.2 : *Eye and retinal anatomy [7]*

1.1.2 Retina and vision system

The retina, being a complex tissue layer consisting of various types of cells, is primarily made up of photoreceptor cells, interneurons, retinal pigment epithelium (RPE) cells, ganglion cells, and glial cells. Photoreceptors serve as the primary sensory cells in the visual system, converting incoming photons into neural signals via photon transduction, while ganglion cells play a crucial role in transmitting visual data from the retina to the brain. On the other hand, interneurons located in the inner nuclear retinal layer establish connections between the photoreceptor and ganglion cell layers. The cellular organization of the retinal tissue structure is illustrated in Figure 1.3.

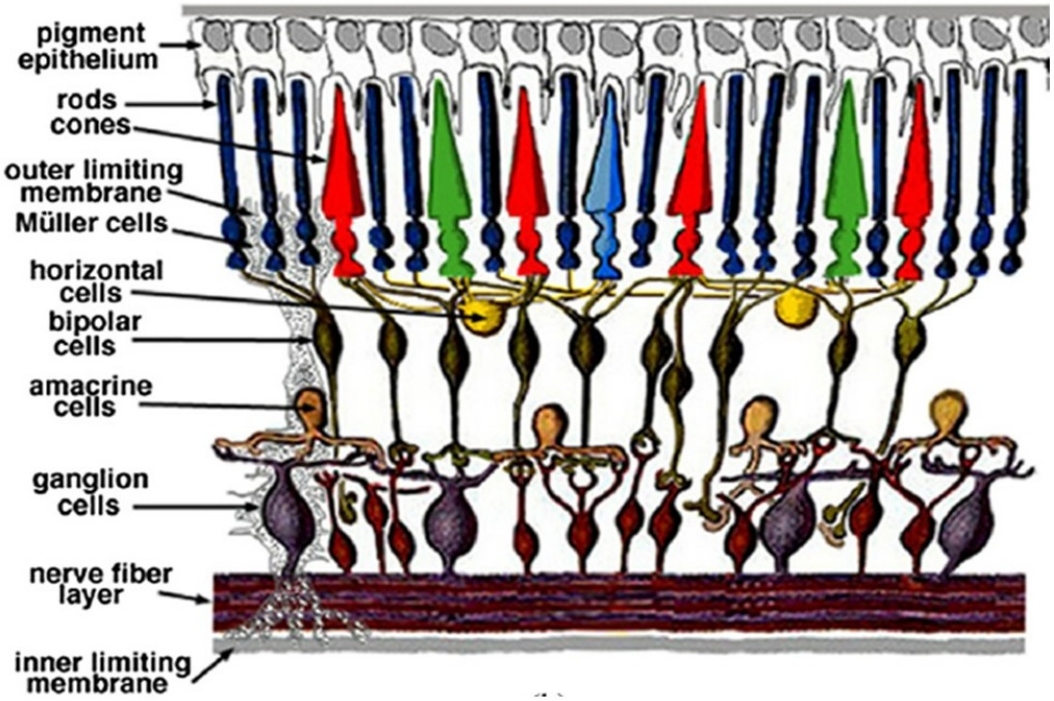


Figure 1.3 : *Retinocellular organization [7]*

The process of vision system starts with the focusing of reflected light from seen objects, combined with adjustments made by the anterior segment of the eye (i.e., the cornea, anterior chamber, posterior chamber, lens and vitrous chamber. Subsequently, this incoming light is converted into neural impulse by the retina, which is then transmitted to the brain for interpretation. Once the recognition is done, one can feel and identify the seen objects. The diagram depicted in Figure 1.4 illustrates the trajectory of light as it traverses various components of the eye within the front retinal regions, focusing incoming light (displayed on the right side) and converting it into electrical

signals at the retina (depicted on the left side) to facilitate image formation. Upon reaching the retina, incoming light interacts with the three primary types of cells: photoreceptor cells, bipolar cells, and ganglion cells, while other accompanying cells such as amacrine and horizontal cells regulate and finetune retinal signaling.

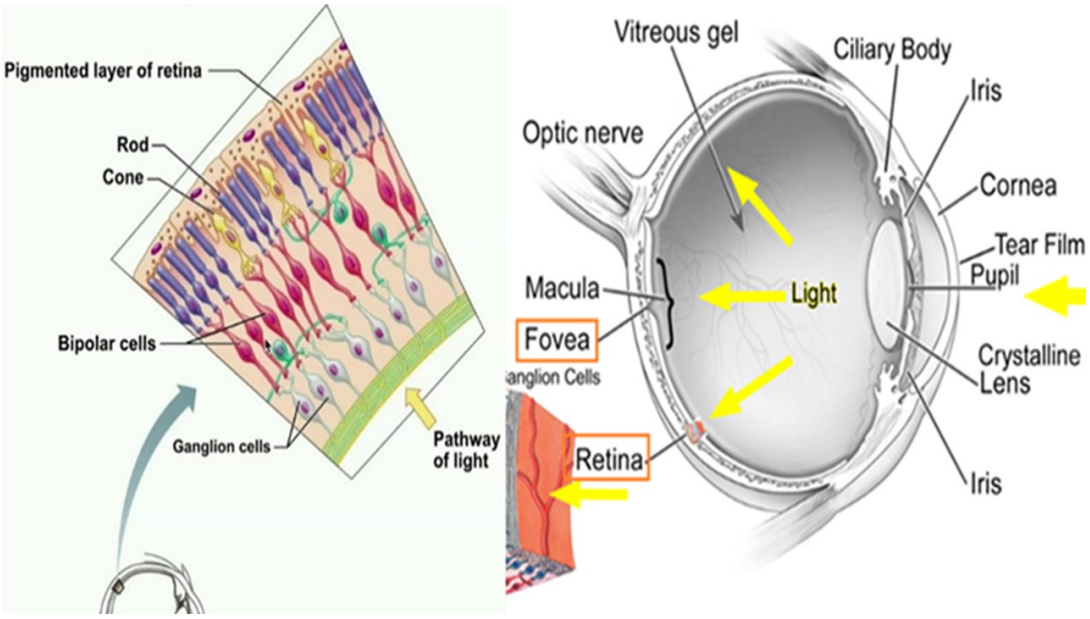


Figure 1.4 : *Vision system*

1.1.3 Retinal diseases

The retina is a metabolically active part of our eye and as a result, both eye specific and systemic based diseases manifest themselves in retinal examinations. These diseases originate either from within the eye, brain, or other systems, like the cardio-vascular system.

Systemic origin diseases

A number of systemic diseases affect the retina. Diabetic retinopathy from diabetes compilations, hypertensive retinopathy, and occlusions from cardiovascular diseases, which are common diseases, have been discussed below.

Diabetic Retinopathy (DR)

Diabetic retinopathy is a complication of diabetic mellitus and the second most common cause of blindness and visual loss in the U.S.A and mostly affects the working

age population [8]. One can classify diabetic retinopathy as either non-proliferative or proliferative. Non-proliferative DR includes mild, moderate, and severe, which are earlier stages of the disease, and proliferative DR is the sight-threatening stage. The optic nerve head, vessel diameters, bifurcation geometry, and vascular tortuosity are the indicators, and hemorrhages, soft exudates, hard exudates, aneurysms and microaneurysms as seen on figure 1.5 , are the damages of the DR disease condition. DR is not curable, but vision loss and blindness from this disease are preventable, and its early management protects eye sight.

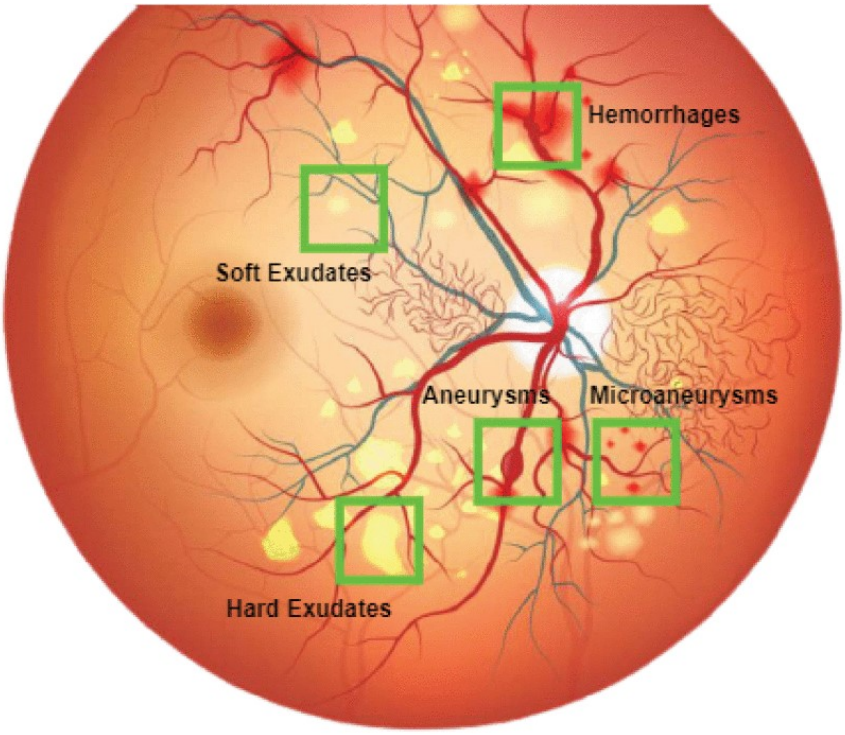


Figure 1.5 : *Diabetic disease conditions [9]*

Cardiovascular disease

Diseases of the cardiovascular system affect the retina mainly by the change they bring to the ratio between the diameter of retinal arteries and veins, known as the A/V ratio. Hypertensive retinopathy and occlusion are the two retinal problems of the cardiovascular system. Hypertensive retinopathy is a retinal problem due to hypertension. It has an effect that can invoke direct retinal ischemia, which causes retinal infarcts visible as cotton wool spots and choroidal infarcts visible as deep retinal white spots. The picture in Figure 1.6 indicates the effect on retinal blood vessels and related complica-

tions due to hypertensive retinopathy.

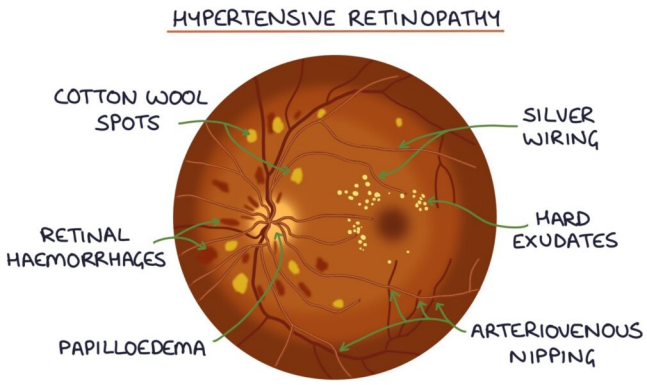


Figure 1.6 : *Hypertensive Retinopathy based disease conditions [10]*

On the other hand, systemic vascular disease can cause arterial and venous occlusions, known as central and branch arterial and venous occlusions. Retinal artery occlusion is a blockage of one of the arteries of the retina, which are blood vessels that carry oxygenated blood from the heart to our retina. A blockage in the main artery of our retina is called a central retinal artery occlusion [11]. Retinal vein occlusion, on the other hand, happens when a blood clot blocks the vein. It can be either a central retinal blood occlusion or a branch retinal vein occlusion. Sometimes it happens because the veins of the eye are too narrow. It is more likely to occur due to diabetes and possibly high blood pressure, high cholesterol levels, or other health problems that affect blood flow [12].As seen on the figure 1.7 , paled retina,attenuated blood vessels and cherry red spots are the expected damages from occlusion disease.

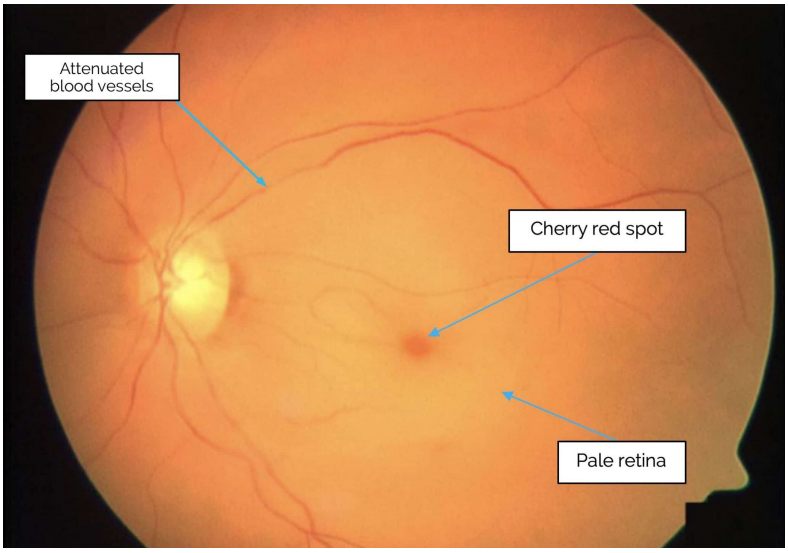


Figure 1.7 : *Occlusion disease conditions [13]*

1.1.4 Eye specific origin diseases

Eye specific origin diseases encompass a range of conditions that primarily affect the eye and its structures. Glaucoma, retinal detachment, lesions, retinal pigmentosa, retinal retnoblastoma, and macular bunker and coats disease are the eye-specific diseases. Glaucoma is characterized by gradual damage to the optic nerve, resulting in a loss of visual field. Early diagnosis and management minimize the risk of glaucoma. Glaucoma primarily acts on the retina by damaging ganglion cells and their axons. The hallmark of the disease is the cupping of the optic disc. The ratio of the optic to the disc cup is an indicator to assess the presence and progression of glaucoma.

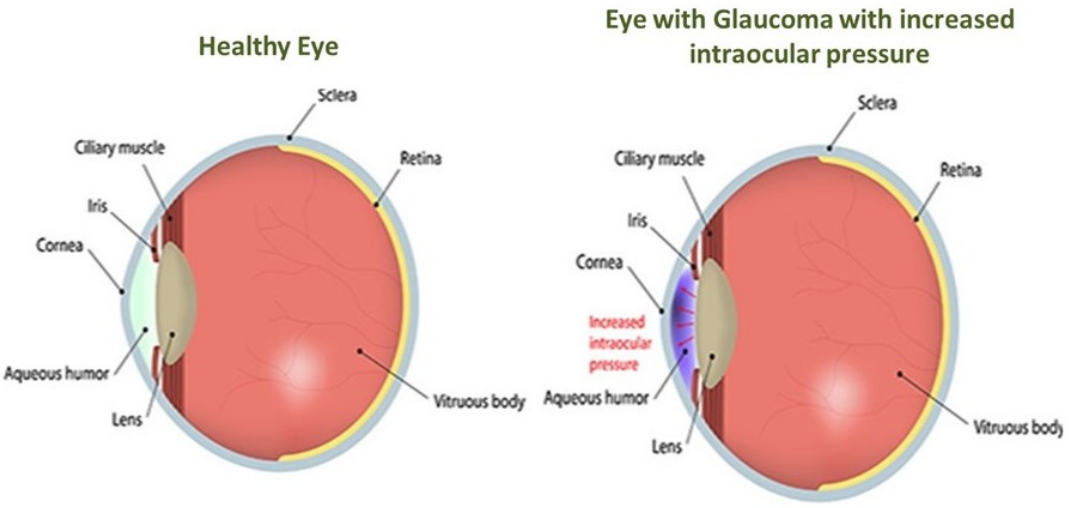


Figure 1.8 : *Glaucoma image [14]*

Retinal detachment is a medical emergency where the retina pulls away from its normal position and requires quick treatment to avoid vision loss. Symptoms include the sudden appearance of floaters, flashes in one eye or both, and reduced peripheral vision. On the other hand, retinal imaging can significantly increase a patient's chances of successfully treating certain types of cancer (lesions). High-resolution images of the retina can reveal early signs of cancer in the eye, such as dark spots that indicate a melanoma. When these symptoms are noticed early, patients can undergo diagnostic testing and seek treatment promptly to prevent cancer from spreading to other parts of the body. Retinal pigmentosa is a hereditary disease that causes black pigmentation and gradual degeneration of the retina. Its symptoms include difficulty seeing at night and decreased side vision. As peripheral vision worsens, people may experience "tunnel vision". However, complete blindness is uncommon. The onset of symptoms

is generally gradual and often occurs in childhood [15]. Retinoblastoma is a cancer that begins in the retina and commonly affects young children but can rarely occur in adults. It is a rare form of eye cancer and a common form of cancer affecting the eye in children. It may occur in one or both eyes [16]. A macular bunker is a scar that forms over the part of the eye responsible for color, central vision, and detail (the macula). This scar can cause a wrinkle, resulting in blurred vision [17]. Coats' disease is a rare congenital, nonhereditary disorder causing full or partial blindness, characterized by abnormal development of blood vessels behind the retina. Coats' disease can also cause glaucoma. It can have a similar presentation to that of retinoblastoma [18].

1.2 Retinal Disease Diagnosis Methods

1.2.1 Retinal imaging techniques

Retinal imaging is the process of capturing images of the retina, which involves analyzing the acquired images to diagnose and study various anomalies. Different imaging techniques allow for the external viewing of the retina and its tissue, including brain tissue. The history of retinal imaging dates back to Jean Merry, a French physician who first observed retinal vessels in a cat by submerging them in water[19]. However, this method was not practical for human use, leading to the invention of the ophthalmoscope. The challenge with early ophthalmoscope imaging was the risk of infectious diseases for physicians. Consequently, the development of photographic eye imaging, specifically the fundus camera in 1910, gained popularity. Another significant advancement was the invention of fluorescein angiographic imaging. This technique utilized a fundus camera equipped with narrow band filters to capture images of a fluorescent dye injected into the bloodstream, which binds to leukocytes. Fundus photography provides a 2D representation of the semi-transparent, 3D retinal tissue. Stereo fundus photography initiated the exploration of the 3D shape of the retina, which was subsequently superseded by confocal scanning laser ophthalmoscopy and optical coherence tomography (OCT). OCT, in particular, has opened up new possibilities for investigating neurodegenerative diseases and serves as a potentially useful biomarker for assessing central nervous system neurodegeneration.

1.2.2 Fundus photography imaging

Fundus imaging is a technique that produces a two-dimensional (2D) representation of the three-dimensional (3D) surface inside the eye, as shown in Figure 1.9. This imaging system consists of a specialized low-power microscope and a camera that are connected. During the imaging process, the patient positions their chin on a chin-rest and rests their forehead against a bar. Meanwhile, the operator adjusts and aligns the camera before activating the shutter release to generate a flash and capture the image. The resulting image is an enlarged and upright depiction of the fundus, with the typical field of view ranging from 30° to 60°, and a magnification of 32.5, depending on the optical properties of the system. It is possible to modify these parameters by utilizing zooming or auxiliary lenses, which enables a wider field of view (FOV) by capturing multiple images at different fixation points. Conversely, improving image quality can be achieved by administering mydriatic eye drops to dilate the pupils prior to imaging. This dilation enlarges the FOV of the fundus and enhances the overall image clarity. Retinal imaging using a fundus camera is cost-effective and safe for documenting retinal abnormalities. It is commonly used in the assessment of diabetic retinopathy, glaucoma, occlusions, and hypertensive retinopathy, serving as a screening tool to identify early stages of retinopathy before symptoms appear. Fundus imaging is also routinely applied in various other eye conditions, such as monitoring changes in the optic nerve or detecting growth or alterations in pigmented retina lesions like choroidal nevi through serial photographs.

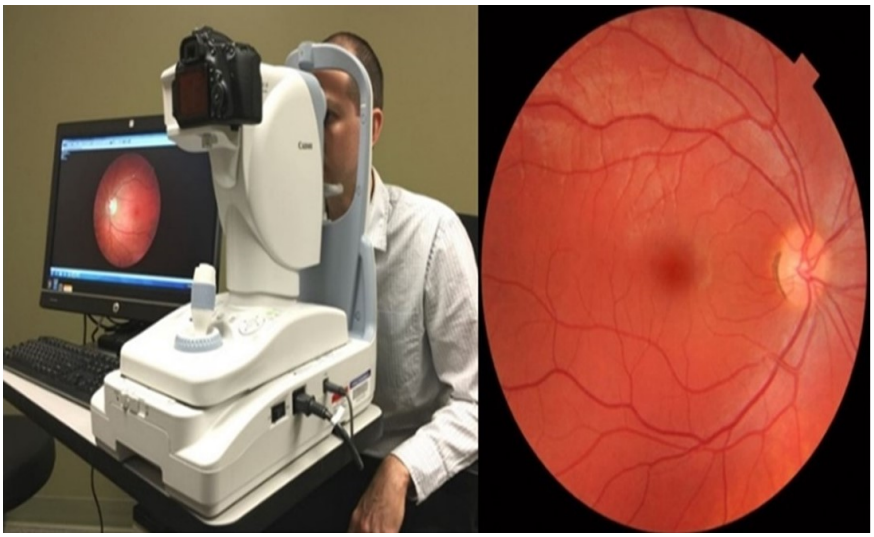


Figure 1.9 : *Fundusphotography imaging[20]*

1.2.3 Diagnosis approaches

The acquired retinal image are subject to manual subjective examination of the disease of interest and may be assisted by the device specific quantification and measurement of disease landmarks in some cases.

1.2.4 Retinal disease classification

Various diseases can be diagnosed and classified from retinal images through fundus camera imaging. The most common and leading cause of blindness includes classfication of diabetic retinopathy, hypertensive retinopathy, glaucoma and occlusion to their types. These diseases can be diagnosed with fundus camera. On the other hand, diabetic retinopathy can be classified into four stages; mild, moderate, proliferative and severe. Occlusions can be divided into two categories as branch and central. The figure 1.10 presents sample images, acquired with fundus camera, from each class of the labelled sets.

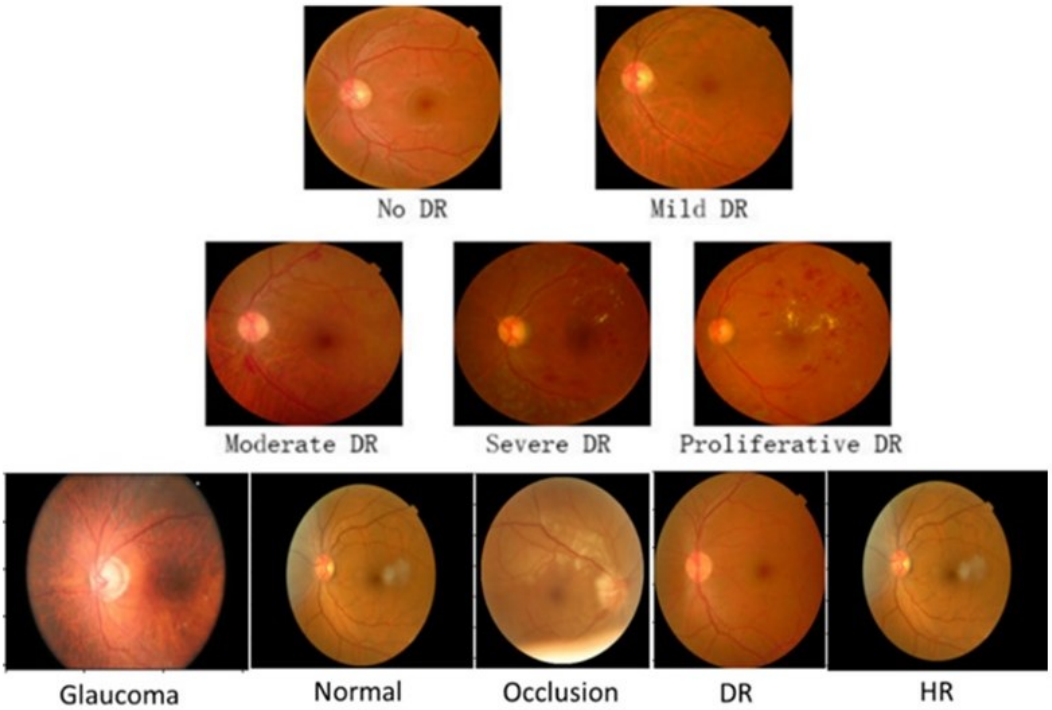


Figure 1.10 : *Typical sample class images*

1.2.5 Manual diagnosis drawbacks

The conventional approach entails physicians performing examinations by visually inspecting retinal images that have been scanned. This particular method is characterized by its time-consuming nature, susceptibility to observer bias, and tiresomeness of the process due to the large images involved. Moreover, the manual process of examining retinal images for the purpose of diagnosing and identifying diseases necessitates the expertise of trained professionals, which presents an additional challenge due to the potential scarcity of such experts, particularly in developing countries like Ethiopia.

1.3 Problem Statement

The manual diagnosis of eye diseases from retinal images captured by fundus camera is a time-consuming, tedious, and subjective process that requires highly trained experts to differentiate between similar features.

Computer aided disease diagnosis methods are showing promising results in automating disease diagnosis. But still there are challenges due to unrecognized deep learning technique tuning and performance assessments. Related works on disease diagnosis from retinal images through deep learning have shown good progress, yet still challenges in the tuning technique and performance assessment many works indicate inconsistent results in multiple disease classification diagnosis from retinal images.

Beside researchers promising work on proposing different methodological approaches for automation, most of their work heavily relies on public data availability than other concerns like prevalence. After-all,there is a gap in research focusing on enhancing the robustness and generalization of deep learning models across diverse patient populations to ensure consistent performance in real-world clinical settings [22]. Therefore, a lack of a fast, accurate, and consistent retinal disease classification from retinal images with datas included from Ethiopian context, hinders the management and treatment of important eye diseases and overall affects health service delivery.

1.4 Objective

1.4.1 General objectives

The general objective of this study is to develop an automatic multiple disease diagnosis and classification system using deep learning technique from retinal images.

1.4.2 Specific objectives

- To implement a state-of-the-art convolutional neural network (CNN) architecture for the classification of the four common eye complications using retinal images captured by a fundus camera.
- To develop a high-accuracy classification system for diagnosing and classifying the four common eye complications using the implemented CNN architecture and retinal fundus images.
- To evaluate the accuracy of the developed model in classifying eye complications by calculating metrics such as precision, recall, F1-score, sensitivity and overall classification accuracy.
- To develop a user-friendly graphical user interface for the system interface, allowing health care professionals to easily input retinal images and access the classification results.
- To compare the diagnosis time reduction to traditional manual methods, and assess the potential time savings by the automated approach.

1.5 Research Questions

- How can state of art deep learning technique can be implemented effectively to achieve high accuracy retinal disease classification model?
- How do the advancements in CNN architectures enable more robust and accurate classification system?
- What training strategies, hyperparameter and optimization methods should be

employed to ensure the deep learning based model generalizes well and achieves high classification accuracy?

- How does the the performance of developed model can be evaluated?
- what are the preferences and requirements for the system interface and how does the developed model meet those requirements?
- How does diagnosis using deep learning algorithms impact the time required for diagnosing retinal diseases compared to manual methods?

1.6 Significance of the Study

- Application wise computer aided classification of retinal diseases using deep learning significantly improves accuracy and also reduces the diagnosis time required for manual diagnosis techniques and consequently, these research finding can be used as decision support system to be used in ophthalmology health service delivery especially where cases are too much and experts are scarce.
- the contextual representative local data contributes valuable data for further studies within problem domain in developing universally diagnosing models.

1.7 Scope

The main aim of this thesis is to study and develop multiple disease diagnosis model from retinal images using ResNet50 based transfer learning techniques. Using Python programming language and CNN architectures, the four common eye complications, with data included from within the Ethiopian context, were diagnosed and classified into types and subtypes from retinal images captured by the fundus camera. In addition to that, the study will also consider developing a GUI interface for system deployment.

1.8 Organization of the Study

The rest of the thesis has been organized into four chapters. In Chapter 2, related works on the automatic classification of diseases have been reviewed. Chapter three explains the methodology and materials used to approach this study. The next chapter is about the results and the discussion of the results obtained. Finally, the conclusion and recommendation for future work of the study have been discussed in Chapter 5.

Chapter 2

Literature Review

2.1 Automatic Retinal Disease Type Classification Methods

A review of related works in computer aided diagnosis(CAD) based multiple disease diagnosis from retinal images with deep learning techniques is discussed in this chapter. The summary is indicated with related existing gaps.

2.1.1 Computer aided diagnosis of retinal disease

Computer aided diagnosis is a cost-effective, feasible, objective, and does not necessitate the presence of highly trained experts. CAD based diagnosis has become a major research subject in medical imaging and CAD based diagnosis from retinal imaging can help increase diagnosis accuracy, reduce subjective variability of disease type classification and subtypes, and reduce the workload of ophthalmologists.

CAD based disease diagnosis can be seen in classification, segmentation and object detection problems.from retinal images can be approached through segmentation, which is based on disease specific biomarkers to identify the disease and aid physicians in diagnosis[19]. The most common segmentation applications focus on OD/OC segmentation, lesion segmentation and blood vessel segmentation. Generally, segmentation is a disease specific approach to diagnosis. On the other hand, classification is a method of categorizing objects or data points to it's predefined categories based on its characteristics or attributes. Retinal disease classification can be seen as disease type,

classifying different retinal diseases, and as sub-types where specific disease classification is done based on the severity scale of the disease.

CAD based disease classification procedures employ either machine learning or deep learning techniques, which are generally included under artificial intelligence systems. Artificial intelligence is an umbrella term consisting of both machine learning and deep learning techniques [23].

2.1.2 Classical machine learning

Traditional machine learning is one technique to automate systems to ease the work load of humans. It solves the problems of segmentation, classification, and detection and relies on crafted features of an image (i.e., shape, texture, color, etc.). With this technique, the image pixels are used as input to the feature extraction process in order to get the feature vectors through quantifying the image content [24]. The feature vectors that resulted from feature extraction serve as an input to the machine learning models. Machine learning is limited by the largeness of the data, feature extraction and dimensionality problems while developing the algorithm.

2.1.3 Deep learning

Deep learning is a technique to automate systems that have networks capable of learning. from data. It differs from traditional machine learning in that an individual's feature selection is not expected, and the algorithm itself learns on its own which features are best for classifying the data. It is a model of multi-layered networks that mimics the human brain and learns from large amounts of data, which is good for handling feature extraction and dimensionality problems [25].

Deep learning algorithms use a hierarchical learning method. That is, information from the first layer becomes an input to the next layer. That is, if primarily only edge like regions have been detected in the lower level layers of the network. Then these edge regions define corners and contours. Merging corners and contours can lead to identifying object parts in the next layer [26]. In deep learning models, a series of hidden layers extract features from the input image, and the final output layer classifies the image and obtains the output class label.

Among deep learning techniques convolutional neural networks (CNN) is highly applicable in medical image classification problems [27]. The concept of CNN has passed through perceptrons which is the first artificial neuron. The progress then advanced with back propagation algorithm that enabled an iterative weight update. LeNet5 has been later in place which was used for bank cheque signature recognition at a time. LeNet5 allowed the spatial hierarchical features from input data.It is also the first model architecture in introducing convolutional layers, pooling layers and fully connected layers which are the typical CNN architecture. Then after many more models have devised with different advances. Figure 2.1 shows the graphical representation of the connection of layers in a neural network.

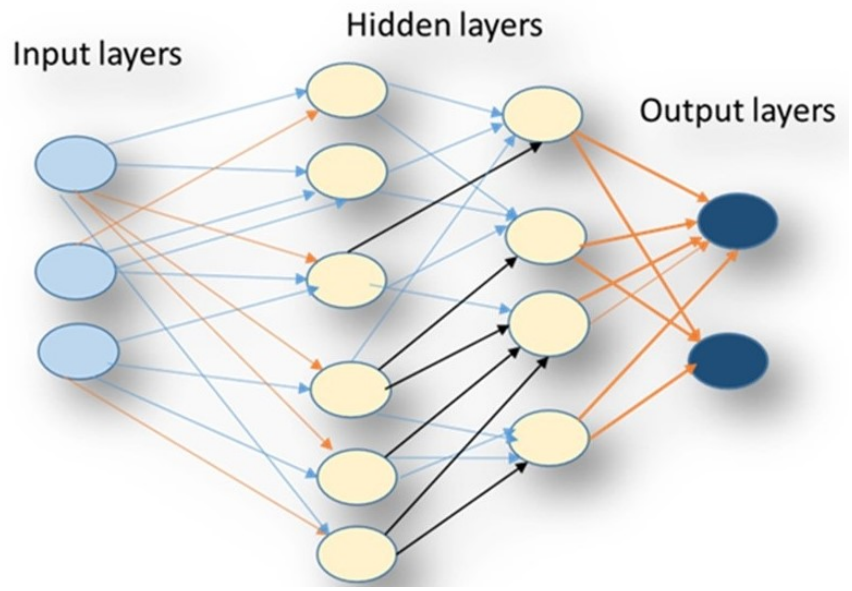


Figure 2.1 : *Typical CNN architecture*

Transfer learning

Transfer learning is a term in deep learning that refers to passing the weight values of a trained neural network to a new use case so that building and training the network from scratch will be avoided and the new network will know the learned features. Using pretrained models with less computational power and time can achieve this. Moreover, pretrained networks are quite competitive against training from scratch [26, 28]. Pretrained models can be used either as a feature extractor or as a classifier by directly freezing the completely convolutional base layer and optimizing the last fully connected layer, or the model can be used as a classifier by fine-tuning it. Currently,

there are different pretrained models that can be used for image classification, segmentation, and object detection (with weights trained on ImageNet) [28]. Figure 2.2 below depicts the most common transfer learning models for image classification.

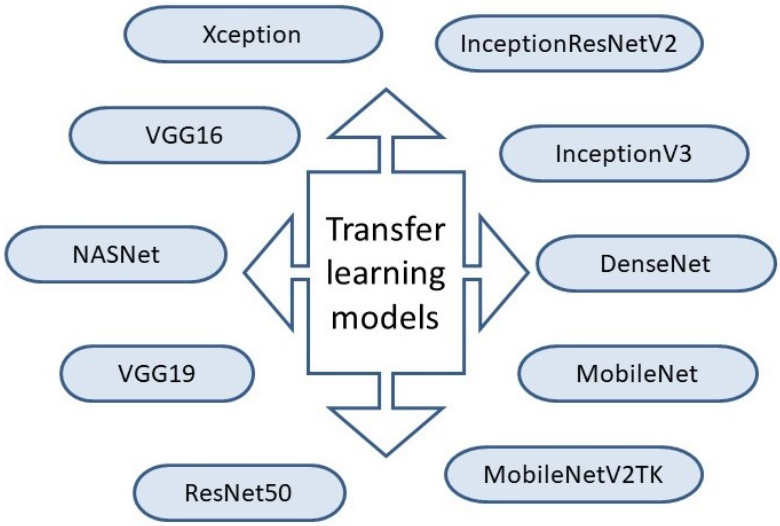


Figure 2.2 : *Transfer learning models*

Hyperparameters and parameters of deep learning models

When training a neural network, hyper-parameters and parameters are the two key things under consideration. Hyper-parameters are parameters that one sets manually before training a system to get the best result, and once they are set, they cannot change during the training. A typical set of hyperparameters for a neural network includes the number of hidden layers, initialization weight values, learning rate, decay rate, dropouts, and activation functions. Parameters are the ones that the model uses to make predictions. Parameters are weights and biases that the machine learns. These values differ for each experiment and epoch. Moreover, they are highly dependent on the type of data and the task.

2.2 Related works on retinal Disease type and sub-type Classification from Retinal Image

Computer aided classification systems capable of accurately classifying multiple diseases from retinal images have gained popularity among researchers. In recent years,

research has focused on the development of computer-aided systems to automatically classify multiple diseases from retinal images using deep learning algorithms that have been recorded.

One study by S.Pouyanfar et al,[29] focused on automatic diabetic retinopathy grading using deep learning and achieved an accuracy of 87%, specificity of 94%, and sensitivity of 87%, demonstrating the feasibility of applying deep learning in retinal image analysis. They used a CNN architecture and approximately 3,648 retinal images for diabetic retinopathy classification. However, this study was limited to classifying diabetic retinopathy subtypes.

Another study by R. Arunkumar and P. Karthigaikumar, [30] proposed a deep belief neural network and multi-SVM classifier to classify six diseases, including age related macular degeneration, diabetic retinopathy, macular edema, retinoblastoma, retinal detachment, and retinitis pigmentosa. However, glaucoma and hypertensive retinopathy were not considered in this study.

In a different study by S. Qummar et al. [31], a method was proposed for automatic classification of multiple diseases from retinal images using a CNN architecture. The paper also examined the effect of image resolution on accuracy and found an accuracy of 82.03% with 31x35 images and 78.91% with 46x53 images.

Another study by Chen et al. [32] proposed and found that the integration of multi-scale hollow CNNs is a feasible way to improve the classification of diseases from retinal images on small datasets. This study focused on optimizing the architecture and relied on publicly available data, without considering disease type classification.

On the other hand, a study of J. Y. Choi et al.[33] explored multi-categorical classification using the CNN architecture MatConvNet and examined the effect of multi-categorization on accuracy on small size data. The study found an accuracy of 30% with 10 categories classification and found 72% with 3 class categorization. While this study focused on the effect of multi-categorization, it did not consider development of classification system and found out as multicategorization affects the classification accuracy.

In addition to the above-mentioned related works and gaps in previous research on

disease diagnosis and classification from retinal images, others have discussed and summarized them as shown in the table below.

Table 2.1 : *Literature Review Table*

Author	Method	Finding
[34]	CNN from fundus images	Focused on Integration of multi scale shallow CNNs on small datasets. Does not consider common causes of blindness. Diabetic retinopathy grading.
[35]	CNN from fundus images	Found better accuracy for DR. Only diabetic classification. VGG19 Pre-trained CNN Architecture
[36]	DCNN (DenseNet121 pretrained network) on APTOS fundus images	Used DCNN and Diabetic retinopathy grading. found precision of 86%, recall of 87% ,F1score of 86% and quadratic weighted kappa of 91.96% and is limited to diabetic classification.
[37]	CNN from fundus images	HR classification using CNN. Is only HR subtype classification.
[38]	CNN from fundus images	Found better accuracy for DR. Only diabetic classification. Pre-trained CNN Architecture
[39]	CNN from fundus images	Used DCNN and Diabetic retinopathy grading. Found better accuracy for DR. Only diabetic classification.
[40]	CNN from fundus images	HR classification using CNN. Is only HR subtype classification.
[41]	CNN from fundus images	DNN. Classified DR into severe, proliferative, and non-proliferative. Is limited to only three DR stages.

Table 2.2 : *Continued Literature Review Table*

Author	Method	Finding
[42]	Used stacking method on fundus images.	Study considered the five class classification of DR and also focused on imbalanced dataset effect achieved 80.8% accuracy. Is only DR sub-type classification.

2.3 Major Gap Analysis and Summary

In summary, previous works show the significance of computer-aided diagnostic systems and deep learning approaches to automatic disease diagnosis from retinal images, which is a big step towards early diagnosis and prevention of exacerbations of the disease. A number of state-of-the-art methods have been developed in the past that have helped in the automatic segmentation and identification of retinal landmarks and pathologies.

However, most of the previous works concentrated on the diagnosis and classification of diabetic retinopathy,which is of course prevalent and major cause of blindness, as most of them relied on public data availability. Others developed a model to diagnose and classify selected diseases based on the availability of datasets in their context rather than the prevalence of those diseases. To be specific there is a gap in research focusing on enhancing the robustness and generalizability of deep learning models across diverse patient populations to ensure consistent performance in real-world clinical settings. After all, there has not been such a developed system using datasets from our country.

In this work, a classification approach has been introduced to automatically diagnose diseases from retinal images, which are common and leading causes of blindness.

Chapter 3

Methodology

3.1 Introduction

This chapter provides details of the materials and methods utilized to obtain the best retinal disease classification from fundus images using deep learning techniques for the readers of this study. Data collection, data preprocessing, model architecture, model training, testing and deployment, as well as research evaluation metrics and materials, are all covered in this area. The next part organizes and presents all of the above described procedures.

3.2 Research Design

This work was based on experimental research and the systematic development of disease diagnosis and classification from retinal fundus images using deep learning. In this research, there are independent variable hyper-parameters and dependent variables that includes the training and validation accuracy and metrics used for performance evaluation.

The dataset consists of retinal fundus images that have been collected from online and offline sources. Online data has been collected from different publicly available online datasets, "The Kaggle dataset [48]" and "The 39 category(paper pointed dataset(Cen et al., 2021))" [49] for the disease types and subtypes.
The data was split into training sets, validation and test sets. The training and validation sets were used for the development and optimization of the models that can be

used to classify the four disease types and the subtypes classifying models belonging to DR and occlusions. In addition to the hold out validation, 5 fold cross validation was used in order to systematically develop high performing model. The performance evaluation of the model was conducted using an independent test dataset once training completed, with testing performed on the model validated through hold-out validation sets. In conclusion, a graphical user interface that is simple to use is designed for the most effective classification trained model. The overall methodology of the system can be summarized as indicated in Figure 3.1 below.

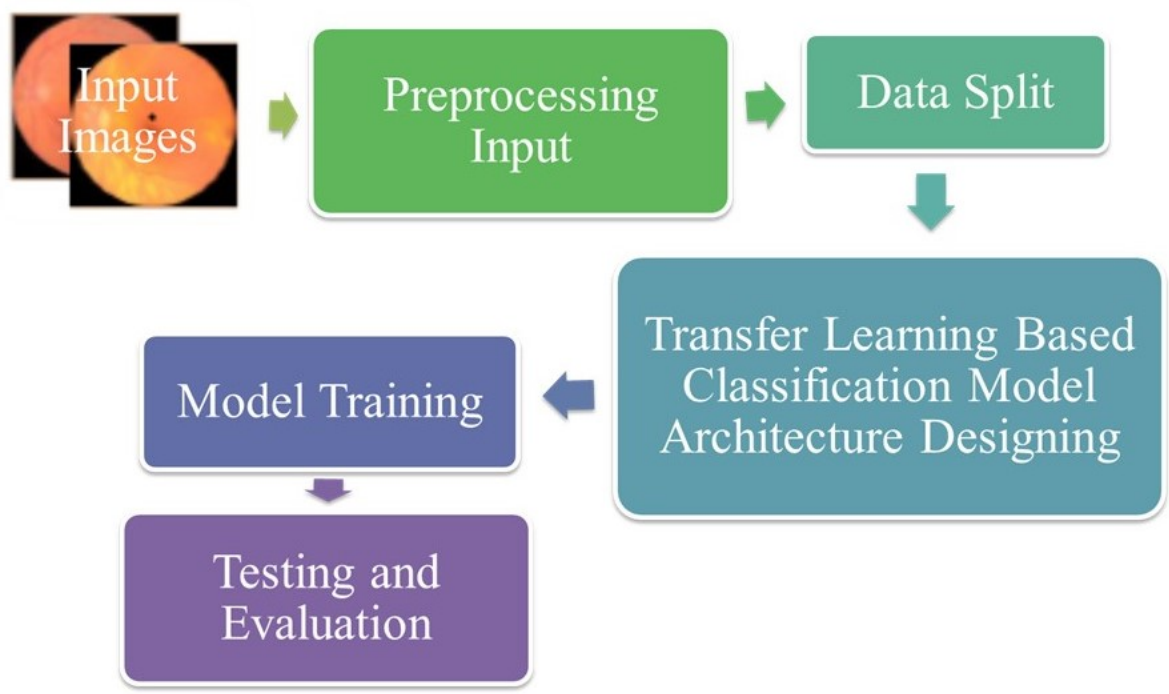


Figure 3.1 : *Overall methodology in block diagram*

3.3 Datasets

For this research, both local retinal images and images from online datasets were used for training, validation, and testing of the system. This helps the system learn different features from a variety of images. Data from Ethiopia is especially important, as relevant data is not readily available online.

3.3.1 Online training datasets

Online data has been collected from different publicly available Online datasets, "The Kaggle dataset [48]" and "The 39 category(paper pointed dataset(Cen et al., 2021))" [49] for the disease types and subtypes.

For the purpose of disease diagnosis, expert clinicians provided the ground truth labels based on fundus camera imaging of diverse age groups. The original publications setting out these datasets provide a comprehensive explanation of the specific diagnostic criteria and procedures utilized in assigning the ground truth labels [48, 49].

Table 3.1 : *Online data size*

Class Types	Disease Type	Data Size
Disease type	Normal	580
	Diabetic Retinopathy	584
	occlusion	576
	Glaucoma	434
	HR	359
DR Subclasses	No DR	1801
	Mild	489
	Moderate	1031
	Proliferative	328
	Severe	393
Occ Subclasses	CRAO	147
	BRAO	386
	Normal	430

3.3.2 Local datasets

Additionally, 9,314 local retinal fundus images were collected from databases with the help of an ophthalmologist at the Blue Visual Clinic in Addis Ababa, Ethiopia. These are images captured specifically for the disease types and subtypes under study, using Kowa VX-10 alpha digital fundus camera from from diverse age groups mostly above 20 years old.

Table 3.2, below shows the number of local data acquired from the Blue Visual Clinic, providing details on the number of images collected for each disease types and subtype.

Table 3.2 : *Local data size*

Class Types	**Disease Type**	**Data Size**
Disease type	Normal	440
	Diabetic Retinopathy	550
	occlusion	380
	Glaucoma	420
	HR	180
DR Subclasses	No DR	898
	Mild	1016
	Moderate	993
	Proliferative	914
	Severe	835
Occlusion Subclasses	CRAO	641
	BRAO	522
	Normal	450

3.4 Dataset Preprocessing

The preprocessing steps in the study were kept minimal to maintain a better generalization of the image conditions. A basic preprocessing step using the built in preprocessing function of Keras, ImageDataGenerator, was performed [50]. The ImageDataGenerator was set to perform a rotation of 90, 180, and 270 degrees, horizontal and vertical, for all the images in the dataset to increase the total number of data points without affecting the image's features with the selected augmentation techniques. The fundus photography images of the dataset have been resized to 224×224 to match the transfer learning model input image dimension. Figure 3.2 shows the sample result from data augmentation.

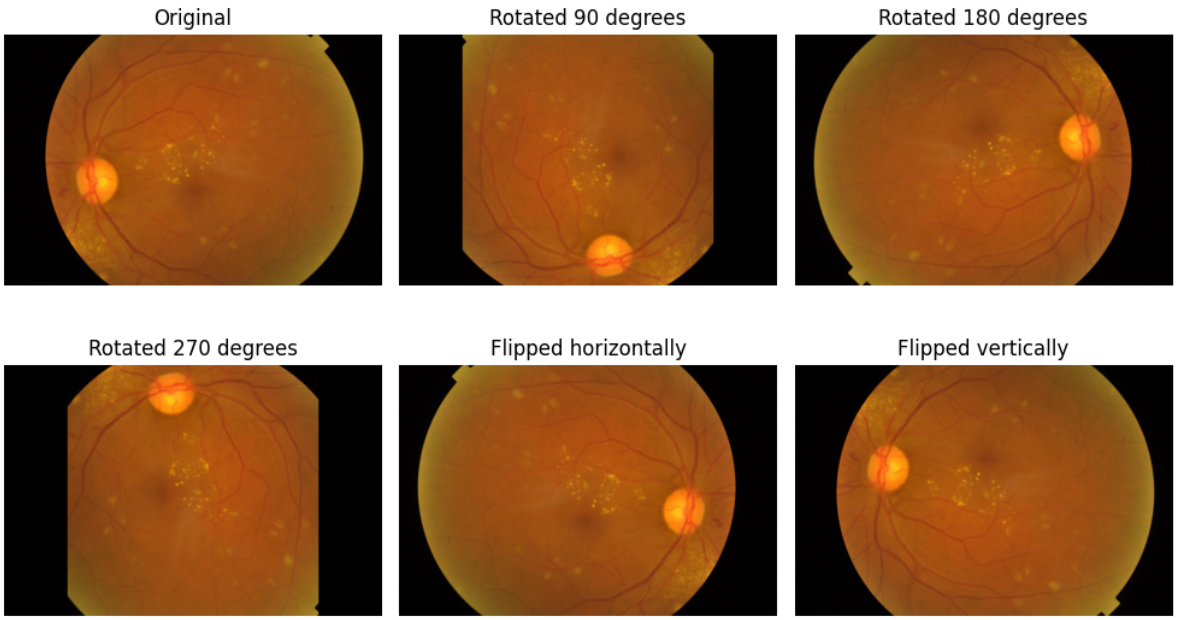


Figure 3.2 : *Sample augmentation*

3.5 Proposed Method for Disease Type and Subtype Classification from Retinal Images

In classification problems, the increase in the number of classes poses a challenge to correctly classify an instance into one of them compared to making the same classification with fewer classes P. D. Moral et al, [51]. On the other hand, according to the study by kim et al. [52] conducted on, two class classification and multi class nine class clas-

sification using VGG19, Inception V3 and ResNet50, ResNet resulted in higher classification accuracy in-terms of multiple class classification whereas, VGG19 and inception V3 outperformed in binary classification. Therefore, for this study, ResNet50 transfer learning based classification of selected retinal diseases into types and sub-types has been considered in order to systematically develop high accuracy classification model.

3.5.1 Transfer learning with CNN

Classification is a way of categorizing which class a given object belongs to. There are a lot of deep learning models at present. These models are deep-belief networks, convolutional neural networks, and recurrent neural networks. In this work, a convolutional neural network is proposed to be used. Convolutional neural networks (CNN) are deep learning architectures that have recently been employed successfully for image recognition and classification tasks [52]. It contains a convolutional layer, max pooling, and a fully connected layer. Convolutional layers apply a convolution operation to the input images, passing the result to the next layer. Pooling layers reduce the dimensions of the data by combining the outputs of neuron clusters at one layer into a single neuron in the next layer. The final fully connected layers connect every neuron in one layer to every neuron in another layer. The soft-max classifier is generally used to recognize the classes in deep-learning algorithms and classify diseases accordingly. Generally, the input image is then fed to the CNN model. Since there will be a limited number of data points, transfer learning will be applied by retraining a model trained for another task by selecting the best performing model, and data augmentation has been performed. The soft max classifier classifies the disease types and subtypes. Lastly, evaluation of the model has been performed.

3.5.2 Transfer learning based model architecture

The ResNet50 transfer learning model, which is 50 layers deep, handles vanishing gradients, and is a powerful model in classification tasks, has been used as convolutional base in this study with additional custom layers and softmax for classification.

ResNet50 model

It was first introduced by He et al. in 2015 in a paper entitled Deep Residual Learning for image recognition [53]. It is known that deep convolutional neural networks are highly useful in solving image classification problems. But, before ResNet, training a very deep neural network was difficult due to the problem of vanishing gradients. As the gradient is back propagated to earlier layers and has repeated multiplication, the gradient becomes extremely small. For this reason, as the network goes deeper, its performance gets saturated, or it will start degrading rapidly.

The major advantage of using ResNet is that it allows training extremely deep neural networks successfully without being affected by vanishing gradients. This is achieved by introducing an "identity shortcut connection" that skips one or more layers. And add the input of a layer to the output of the layer so as to feed the next layer of the network. Identity short-cut connections add neither extra parameters nor computational complexity [52].

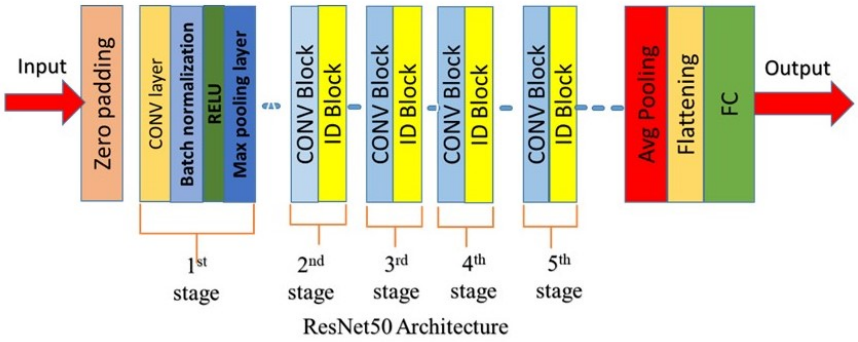


Figure 3.3 : *ResNet50 model architecture*

Generally for this study, pretrained ResNet50 has been used as convolutional base followed by additional custom layers with flattening layer, Dense layer with Relu activation, drop out of 0.2 and softmax on the top for tuning and making classification.

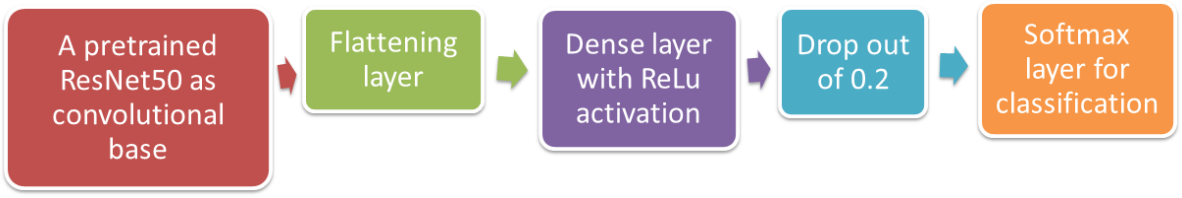


Figure 3.4 : *ResNet50 based designed model architecture*

3.5.3 Hyperparameters of the model

In this research, different hyper-parameters were fine-tuned to increase the performance of the system. These include choosing the right optimizer, adjusting the learning rate, choosing the appropriate activation function, and choosing the proper loss function.
As an optimizer, the ADAM [53] optimizer was chosen for its best performance in terms of both speed of convergence and accuracy. The number of epochs used was 50, the learning rate was set to 0.00001, and the activation function used was ReLu. The loss function for multiclass classification was categorical-cross-entropy.

3.6 Performance Evaluation Metrics

After building a model and training the network, its performance must be evaluated so as to know the true result. There are different ways to evaluate a model. The first one is by using a confusion matrix and getting the TP, TN, FP, and FN rates of the predicted values. The second is by calculating and getting the precision, recall, F1-score, and overall accuracy values of a model. For instance, for a binary classifier of classes A and B, a model can be evaluated using a confusion matrix. Figure 3.5 indicates the TP, TN, FP, and FN components for class A. From the table, we can see that the green-shaded part indicates the TP and TN parts [53].

- A TP (True Positive) value indicates that what is predicted is true.
- A TN (True Negative) value indicates that the predicted class is truly negative.
- A FP (False Positive) value indicates that a thing is predicted as if it is part of the class while it is not.
- FN (False Negative) the prediction indicates that it is not part of the class while it is.

Once the confusion matrix is ready, the classification report containing the accuracy, precision, recall specificity, and F1-score can be done. So, given a class prediction

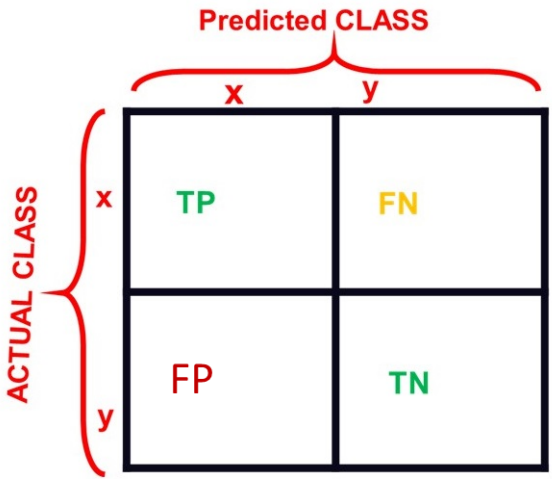


Figure 3.5 : *Confusion matrix instance*

from the classifier, the precision is the one that answers the question, "How likely is it to be correct?" It is calculated using equation (3.1), and recall or sensitivity will indicate the answer to "Will the classifier identify it?" It is calculated using equation (3.2); the F1-score is the harmonic mean of precision and recall. It is calculated using equation 3.3.. The model is said to be performing well if we have a high F1-score, and specificity is the one that determines the proportion of actual negatives that are correctly identified. To measure the proportion of actual negatives that are correctly identified, the specificity of the model was calculated using equation (3.4). Finally, the model's performance was measured using the accuracy metrics in equation (3.5). The performance metrics were calculated using the equations given below:

$$precision = \frac{TP}{TP + FP} \tag{3.1}$$

$$Recall = \frac{TP}{TP + FN} \tag{3.2}$$

$$F1 - score = \frac{2 * precision * recall}{precision + recall} \tag{3.3}$$

$$Specificity = \frac{TN}{TN + FP} \tag{3.4}$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.5}$$

3.7 Graphical User Interface (GUI)

In this study, tkinter and Python programming language, was used to develop the final interface of the system. It incorporates the "browse image button" and "load image button" to browse and load the digital retinal image, respectively, and the "browse weight button" and "load weights buttons" to browse and load trained model weights. The "test image button" will be used to operate the classification with loaded model weights and the image. The "notification box" helps to give feedback on each of every single click. Furthermore, the "test button" helps to make classification tests on the image, and the result will be displayed on the space labeled "Your result shown here".Figure 3.6 shows a snap shot of the general layout of the developed GUI.

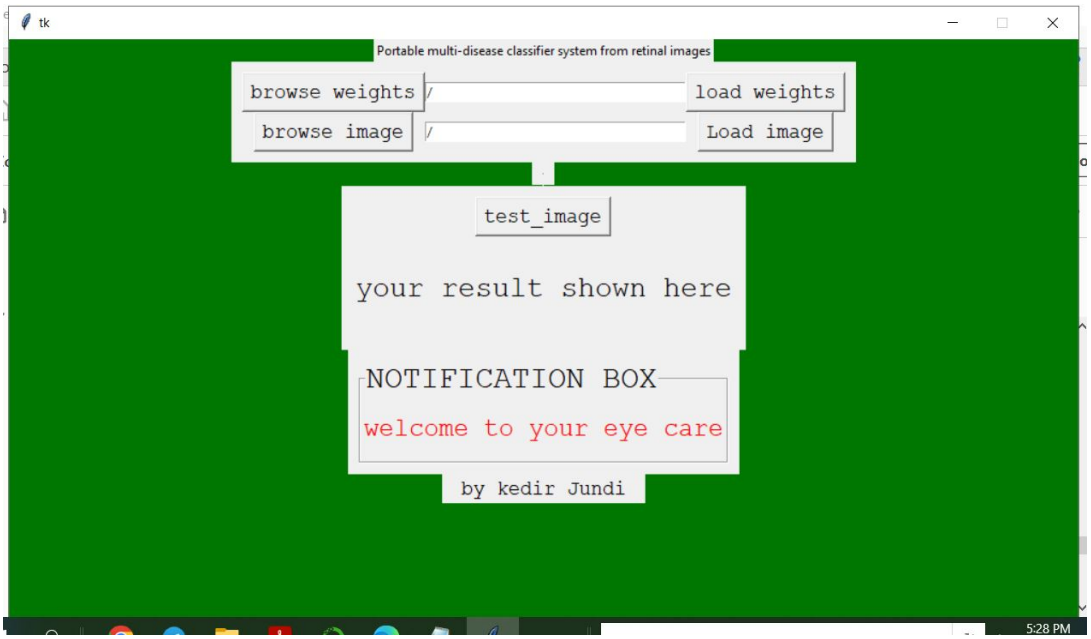


Figure 3.6 : *GUI template*

3.8 Materials

Software:

- Python 3.7.3 software with different modules like keras, tensor flow, sklearn and others for writing and training the system.
- Jupiter notebook python IDEs were used to write and execute the code.

- Tkinter and Jupyter notebook to develop the graphical user interface of the system (GUI).
- kaggle dataset data repository and 39 category dataset.

Hardware:

- fundus camera
- ASUS Intel(R) Core(TM) i5-8250 CPU, @ 1.60GHz 1.80GHz Laptop having 8GB RAM, 64 bit operating system, x64-based processor, Windows 10.

Chapter 4

Result and Discussion

4.1 Preprocessing Result

Images augmentations were made with built in keras function, ImageDataGeneration, using rotation and flipping operations to meet class wise data balancing and data boosted as well. Accordingly, the class wise data has been limited to an average of 2276 for disease type, 2669 for DR sub-type and 2576 for occlusion sub-type classifications respectively. Image has augmented two times, three times, and four times, depending on the true data size in data repositories. Moreover, Table 4.1 shows the overall total data size distribution.

Table 4.1 : *Total dataset after augmentation*

	Types	**True size**	**Generated**	**Total**
Disease type	DR class	1634	1134	2768
	Occlusion	739	1436	2175
	Glaucoma	817	1634	2451
	HR	614	1842	2456
	Normal	1016	1016	2032
	No DR	2699	-	2669

Table 4.1 – ***Total dataset after augmentation***

	Types	**True size**	**Generated**	**Total**
DR Subclasses	Mild	1505	1194	2669
	Moderate	2024	675	2669
	Proliferative	1242	1457	2669
	Severe	1228	1441	2669
Occ Subclasses	CRAO	788	1576	2364
	BRAO	908	1816	2724
	Normal	880	1760	2640

Generally, about 11,882, 13,345 and 7728 images have been used for disease type, DR sub-type and Occlusion sub-type respective classification.

Train-validation-test set split has done by keeping 10% of total datasize for testing, 20% of remained training set for validation sets and the remaining set to be used as training datasets.

4.2 Model Training Results

In this study, 5 fold cross validation and hold out validation methods were employed in order to assess the performance of each of the models during training.

4.2.1 Disease type classification training result

The disease type classification indicates the classifications of DR, HR, Glaucoma, Occlusion and Normal. The model's performance using 5 fold cross validation has been measured and presented as averaged train and validation accuracy with included train and validation accuracy scores for each of folds as in the Table 4.2.

Table 4.2 : *5-fold Disease Type Cross-Validation*

Folds	Training Accuracy	Validation Accuracy
Fold 1	93%	89%
Fold 2	95%	91%
Fold 3	94%	91%
Fold 4	95%	91%
Fold 5	96%	91%
Averaged Accuracy	94.6%	90.4%

Accordingly, the average training and validation accuracy obtained from five cross validation was 94.6% and 90.4% with individual folds training accuracy from 93% to 96% and for validation accuracy from 89% to 91%.

The result for hold out validation method represents training and validation accuracies and losses as shown with training curves in the Figure 4.1. For disease type classification, a better validation loss was achieved at the 47th epoch, which is a validation loss of 0.1490. Hence, the model has saved the weight value acquired at the 47th epoch for the classification task. As a result, 97.68% training accuracy and 94.433% validation accuracy were achieved.

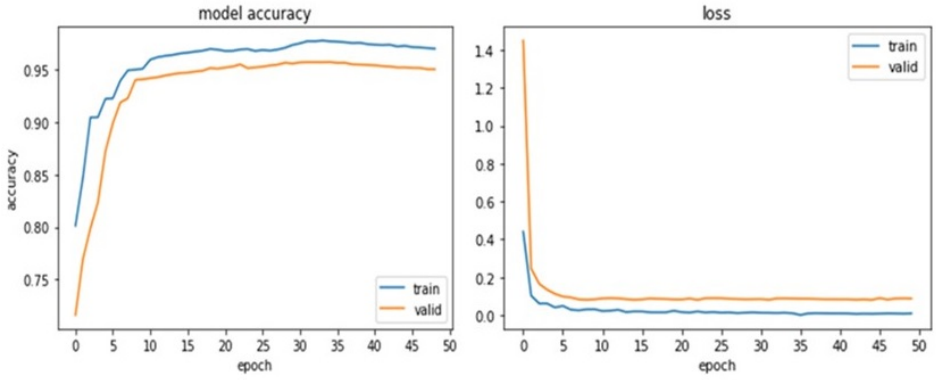


Figure 4.1 : *Disease type model training accuracy and loss curve*

4.2.2 DR Subclass classification

For this subclass, with holdout method better validation is achieved at 49th epoch with loss of 0.090. Hence, the model has saved the weight value acquired at the 49th epoch for the classification task. As a result, 98.68% training accuracy and 95.25% validation accuracy were achieved.

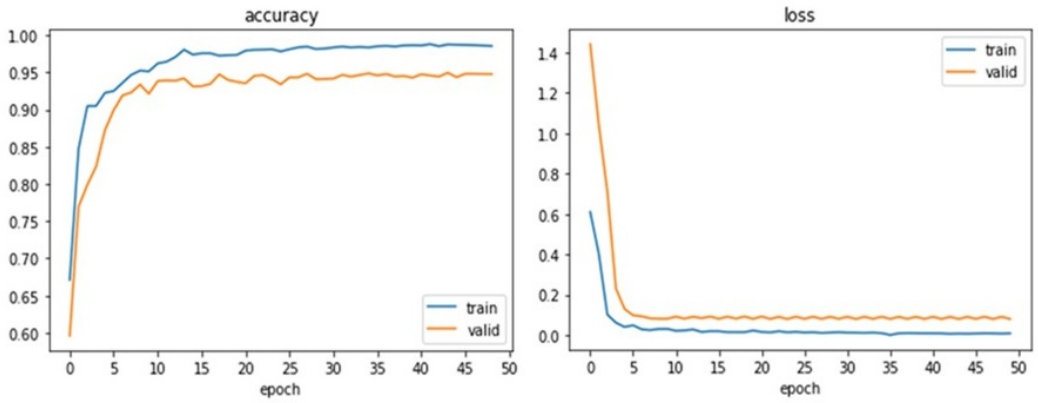


Figure 4.2 : *Model training accuracy and Loss for DR*

On the other hand, for models performance assessment using 5 fold cross validation, the average training accuracy and validation accuracy obtained were 94.2% and 91.8%, with individual folds training accuracy from 93% to 95% and validation accuracy ranging from 91% to 93%. The Table 4.3 below presents related findings.

Table 4.3 : *5 fold DR sub-type cross validation*

Folds	Training accuracy	Validation accuracy
fold 1	95%	93%
fold 2	94%	92%
fold 3	94%	91%
fold 4	95%	92%
fold 5	93%	91%
averaged accuracy	94.2%	91.8%

4.2.3 Occlusion subclass

By the same procedure, 5 fold and hold out validation were used on occlusion subtype datasets. Consequently, better training and validation accuracy and loss were recorded at 43rd epoch with hold out validation as shown in the figure 4.3 below. Hence, the model has saved the weight value acquired at the 43rd epoch for the classification task with 96.43% training accuracy and 95.54% validation accuracy.

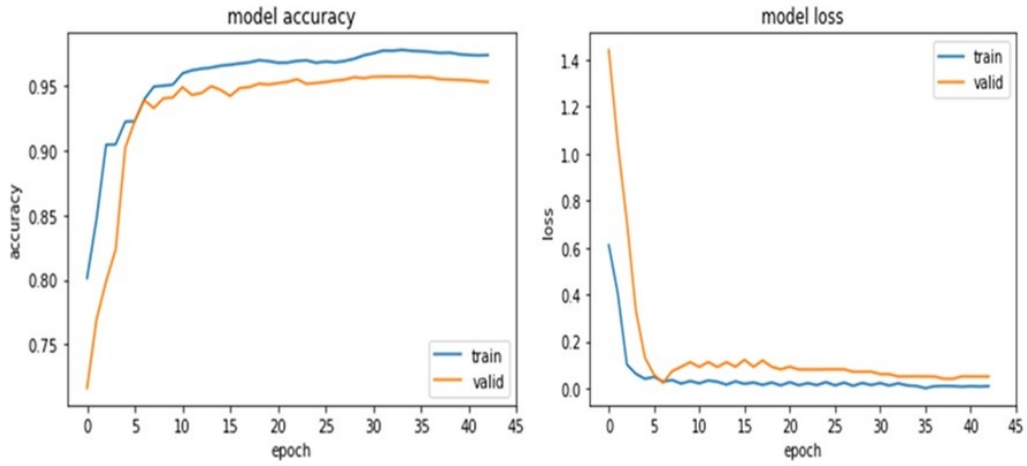


Figure 4.3 : *Occlusion subtype model training accuracy and loss curve.*

On the other hand, for 5 fold cross validation, 91.7% averaged training accuracy and 88.8% validation accuracy with individual folds training accuracy ranging from 90 to 93, and validation accuracy from 87% to 90% was achieved as in the table 4.4 below.

Table 4.4 : *5 fold occlusion sub-type cross validation*

Folds	Training accuracy	Validation accuracy
fold 1	93%	90%
fold 2	92%	89%
fold 3	90%	87%
fold 4	93%	90%
fold 5	91%	88%
averaged accuracy	91.7%	88.8%

4.2.4 Summary of training results

During the training of each models, the state of the model at each step of the training algorithm has been evaluated to give an idea of how well the model is learning using training datasets and also be evaluated on both a 5 fold cross and hold-out validation dataset to have an idea of how well the model is "generalizing". The training curves from hold out method and averaged training and validation accuracies from the 5 fold cross validation tell us the dynamics whether the model training under fits, over fits or is of good fit. The models training curve to good fitted when identified training and validation loss decreases to a point of stability with a minimal gap between the two final accuracy and loss values. After all, in both cross validation and hold out validation resulted in consistent and satisfactory results highlighting potential of the model for real world application.

4.3 Testing Phase Results

4.3.1 Disease type classification test result

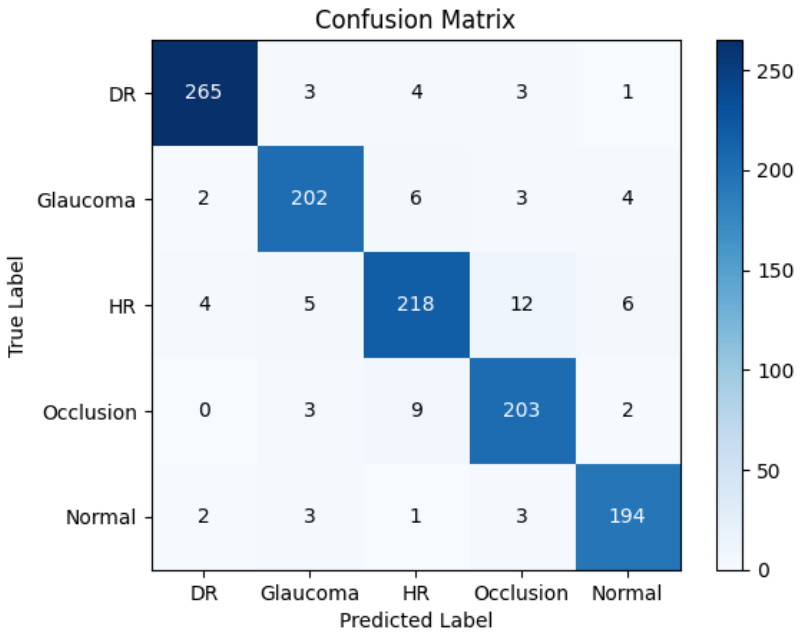


Figure 4.4 : *The four class confusion matrix*

Based on the confusion matrix done, the TP, TN, FP, and FN values were known, and from those values, the precision, recall, specificity, F1-score, and test accuracy were calculated. Table 4.5 below represents the overall result of the disease type classification.

Table 4.5 : *The Four class evaluation metrics result*

	Precision	Recall	Specificity	Accuracy	F1 score
DR	97%	96%	99%	98%	96%
Glaucoma	94%	93%	99%	97%	93%
HR	92%	89%	98%	96%	90%
Occlusion	91%	94%	98%	97%	92%
Normal	94%	96%	99%	98%	95%

4.3.2 DR-subclass classification test result

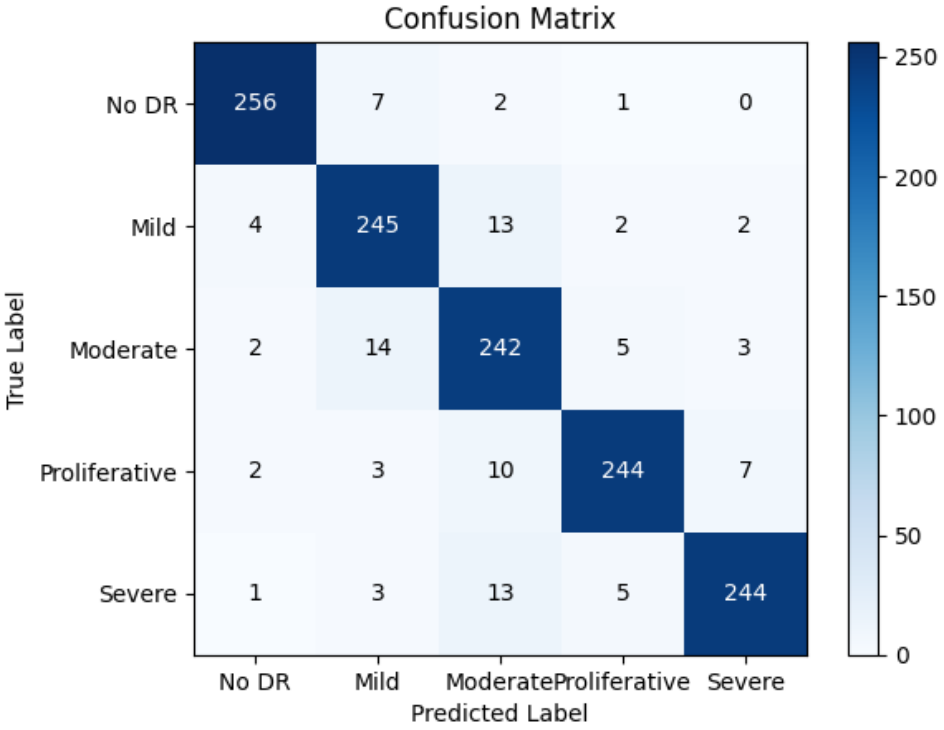


Figure 4.5 : *DR subclass confusion matrix*

From the results found using the confusion matrix, the precision, recall, specificity, F1-score, and test accuracy results were calculated and known through the classification report. Table 4.6 indicates the subtype classification results.

Table 4.6 : *Evaluation metrics for DR subclass*

	Precision	Recall	Specificity	Accuracy	F1 score
No DR	97%	96%	99%	99%	96%
Mild	90%	92%	97%	96%	91%
Moderate	86%	91%	96%	95%	88%
Proliferative	95%	92%	99%	97%	93%
Severe	95%	92%	99%	97%	93%

4.3.3 Occlusion subtype classification

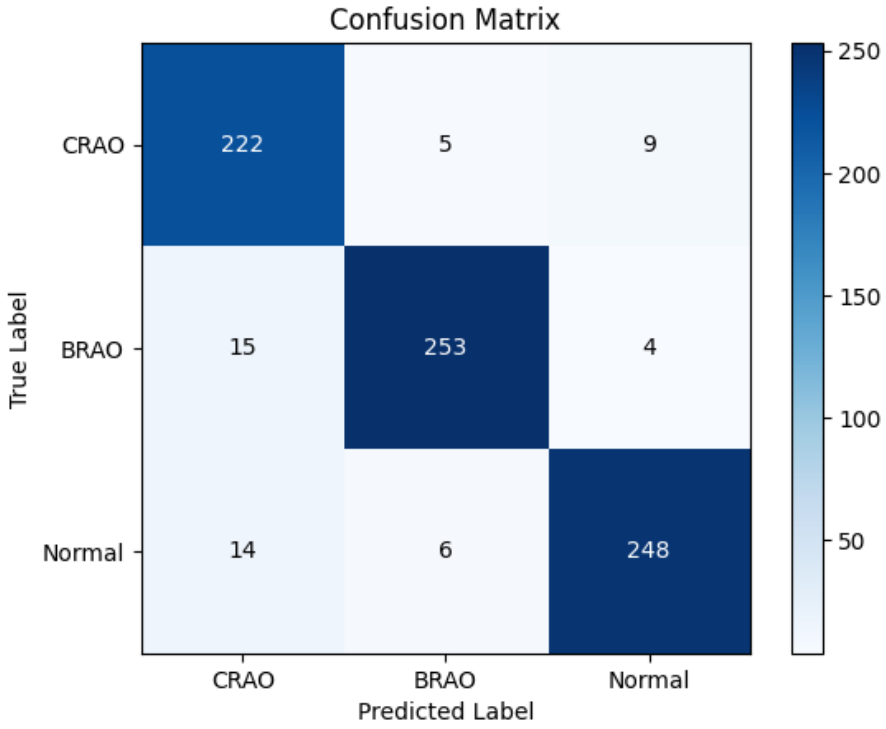


Figure 4.6 : *Occlusion subclass confusion matrix*

From the results found using the confusion matrix, the precision, recall, specificity, F1-score, and test accuracy results were calculated and known through the classification report. Table 4.7 indicates the subtype classification results.

Table 4.7 : *Evaluation metrics for Occlusion subclass*

	Precision	**Recall**	**Specificity**	**Accuracy**	**F1-score**
CRAO	88%	94%	95%	94%	91%
BRAO	96%	93%	98%	96%	94%
Normal	95%	93%	97%	96%	94%

Finally, the general performance of the developed models for disease type and subtype classification of a given image was assessed, with an averaged test precision, recall, f1 score, specificity and accuracy and the summarized as in the table 4.8 below.

Table 4.8 : *Evaluation summary table*

Class types	**Precision**	**Recall**	**Specificity**	**Accuracy**	**F1-score**
Four class	93.6%	93.6%	98.6%	97.2%	93.2%
DR Subtype	92.6%	92.6%	98%	96.8%	92.2%
Occlusion Subtype	93%	93.3%	96.6%	95.3%	93%

4.4 GUI

The developed GUI provided an interactive platform for users to interact with the system and has been tested with respect to response time and ease of use. It has been found to be easy to use and convenient for users. Once initialized, the result was achieved within 5 seconds. With “Browse Image”, “load weights” and “Test” buttons,

it is enabled to easily upload images and obtain test result. In addition the "notification box" pops up the feedback for each click in steps.

After all, some of individuals who got interacted with the system reported that they were able to easily load images and get test results without the need for an extensive technical knowledge. Furthermore, the snapshot illustrations are given in appendix part.

4.5 Discussion

The retina, being metabolically active part of our eye, is susceptible to various organ specific and systemic based diseases that can manifest within it. The majority of these diseases are incurable, necessitates routine screening and are the main causes of blindness [21]. Diabetic retinopathy, glaucoma, occlusion and hypertensive retinopathy are common eye health problems and causes of blindness nowadays [9]. These diseases primarily cause retinal complications and can be obtained from a retinal examination. The challenges posed by these diseases and the importance of the retina in disease diagnosis make multiple disease diagnoses from the retina a matter of high concern.

Retinal images acquired through a fundus camera has remained an irreplaceable imaging mode due to its flexibility, cost and ease of use. Once the retinal image is acquired, physicians look at the image and differentiate the related anomalies, which is a common technique to identify common eye health problems and their sub-types. Unfortunately, there is still a shortage of experts, especially in developing countries like Ethiopia, where the increasing number of patients makes disease diagnosis and management tedious and complicated. Moreover, the subjective nature of retinal image interpretation, which relies on expert performance, combined with the similarity feature of retinal images, often leads to misdiagnosis. Consequently, there is a need for a computer-aided diagnosing system.

In this study, a diagnosis of multiple diseases from retinal images into types and subtypes has been proposed based on a deep learning technique. For model training, validating and testing, fundus photography images were acquired from online datasets (Kaggle and the 39 category) and local data from Blue Visual Clinic. In the preprocess-

ing step, image resizing and data augmentation were applied to images multiple times based on the required data size. The aim of preprocessing the images is to balance the data and increase the dataset, enabling the classification model to take advantage of the improved data as the performance of deep learning models are highly dependent on the type and quantity of training data.

Data augmentation was performed by rotating the images 90 degrees, 180 degrees, 270 degrees and flipping vertically and horizontally more of bring class wise data balancing and also to increase the available data without affecting its features. As a result, the total number of data points increased two times, three times, and four times based on the original data size. To maintain a balanced number of data points in each class, the number of images in each class was limited to an average of 2376 for disease type, 2669 for DR sub-type and 2576 for occlusion sub-type. Next, all image data were resized to 224 x 224 to have a uniform image size and make them suitable and matched input to classifying model. A 5 fold cross validation and the holdout method validation was used, with dataset split into 70% training, 20% validation and 10% testing sets.

The main aim of this study was to classify retinal disease types into diabetic retinopathy, glaucoma, hypertensive retinopathy, occlusion, and normal. Additionally, to classify DR and occlusion into their subtypes. To achieve this, three different classification models were developed using the ResNet50 based transfer learning with soft-max as classifier. The best result was achieved by training a fine-tuned ResNet50 model, which used an ADAM optimizer, a learning rate of 0.00001, a Rectified Linear Unit (ReLU) activation function, a batch size of 32, and a categorical cross-entropy loss function for fifty epochs.

The learning performance of the model over 50 epochs was evaluated by plotting a learning curve for hold out validation as shown in Figure 4.1 to Figure 4.3 and presented as averaged accuracy for 5 fold cross validation, during each training phases.

For hold out validation two different metrics, accuracy and loss, were used to evaluate the model's optimization according to cross-entropy loss and its classification accuracy. A good result was obtained when the accuracy curve increased and the loss curve decreased with an increasing number of epochs for both the training and val-

idation datasets.Therefore, both metrics plot were created for each training task. By reviewing the learning curves shown in Figure 4.1 to Figure 4.3, the performance of the model was evaluated for all classification tasks. It can be concluded from the plots that the model learned well and had good generalizability performance for all classification tasks, as indicated in Figures Figure 4.4, Figure 4.5 and Figure 4.6.afterall,

For 5 fold cross validation the model resulted in an averaged training accuracy of 94.6% with individuals folds accuracy ranging from 93% to 96% in disease type classification model. For DR subtype an average training accuracy of 94.2% with subfold training accuracy range between 93% to 95%. For occlusion similar result with average training accuracy of 91.7% and individual training accuraries ranging from 90% to 93% has achieved.

Afterall, both cross validation and hold out validation yielded consistent and satisfactory results, indicating models potential for real world application.

Finally, testing was performed using separate test dataset and model evaluation metrics were then calculated. An averaged test accuracy of 97.2%, 96.8% and 95.3% were achieved for disease type, DR sub-type and Occlusion sub-type respective classification. These indicate that the result with ResNet50 based classification model was satisfactory, and outperformed similar studies [32, 43] within the problem domain in terms of evaluation metrics and comparison summary is provided as in the table 4.9 below.

Table 4.9 : *Result comparison table*

	Precision	**Recall**	**Specificity**	**Accuracy**	**F1 score**
pratt et. al [32]	-	30%	95	75%	-
Qummar et al [43]	63.85%	51.5%	86.72%	80.8%	53.74%
our method for DR	93.2%	93.4%	98.2%	93%	93.2%

Additionally, a provided user-friendly and easily applicable GUI solve the problem with the existing retinal disease diagnosis method and will assist physicians in making

diagnosis decisions and making better disease management plan.

On the hand, by considering the time taken by experienced ophthalmologist to make the manual diagnosis during local data acquisition, comparison with automated method resulted in significant time difference. On average, the automated approach significantly required less time for retinal disease diagnosis compared to the manual procedure. The manual diagnosis took considerably longer time due to the time required for interpretation and decision making. Therefore, the time saving facilitated by automated approach can enhance work flow and resource allocation, and indicates a need for continued research and implementation of the technique for better patient care.

4.6 Limitation

In this study, ResNet50 based transfer learning has been used as a deep learning technique. In order to improve the accuracy of the model even further it is important to investigate model ensemble techniques and also include more contextual data sources to develop universally classifying system. Again in terms of model evaluation, it is beneficial to conduct an extensive performance evaluation by incorporating other cross-validation techniques which is limited to 5 fold cross and hold-out validations in this study. and testing the model on multiple datasets.

Chapter 5

Conclusion and Recommendation

5.1 Conclusion

This study proposed an automatic disease type and subtype diagnosis using deep learning classification to types and subtypes from retinal images. ResNet50, which is a pretrained model on the ImageNet dataset with 1000 object categories, as convolutional base through transfer learning and with softmax as a classifier was used to classify retinal diseases into diabetic retinopathy, hypertensive retinopathy, glaucoma, and occlusion. Further classifies diabetic retinopathy into four subtypes (Normal, Mild, Moderate, proliferative and severe, and occlusion into two subtypes (branch and central occlusion).

The pre-trained ResNet50 was used as convolutional base as a feature generator, and the extracted features were fed to a soft-max classifier. The results obtained were then compared to the related bench mark finding. The overall results obtained were much better for type and subtypes classification. Moreover, average accuracy of 97.2%, 96.8%, and 95.3% were achieved for subtype classification, DR sub-type classification and occlusion sub-type classification respectively.

This developed system can be used as a decision support system in the diagnosis of diseases from retinal images, and this will have a great impact by helping ophthalmologists, especially in those low resource settings where the expertise is scarce.

5.2 Recommendation

It can be seen from the literature that most works are done using specific types of datasets. Using a variety of data sources diversified in gender, ethnicity, race, color, and others is still important in order to develop a universal diagnostic system that is not biased due to the mentioned conditions. Even in this study, the local data included was from a single medical center due to a lack of facilities and costs at different sites.

This study is limited to classifying the top four causes of blindness, but it is also important to identify the classes of other retinal disease types. Moreover, it is limited to classifying retinal diseases from one-time fundus images, but such research can be extended to chronological time images to predict the exacerbation of incurable diseases like diabetics, which have a greater impact on their management.

Bibliography

[1] S. Resnikoff et al., “Global data on visual impairment in the year 2002,” Bull. World Health Organ., vol. 82, no. 11, pp. 844–851, 2004, doi: /S0042-96862004001100009.

[2] J. M. Roodhooft, “Leading causes of blindness worldwide.,” Bull. Soc. Belge Ophtalmol., no. 283, pp. 19–25, 2002.

[3] “9828-Article text-51489-1-10-20040202.pdf.”.

[4] N. Zerihun and D. Mabey, “8215 (90.4%),” Ophthalmic Epidemiol., vol. 4, no. 1, pp. 19–26, 1997.

[5] R.O.Fite, E. A. Lake, and L. K. Hanfore, “SC,” Diabetes Metab. Syndr. Clin. Res. Rev., 2019, doi: 10.1016/j.dsx.2019.04.016.

[6] T.K. Azeze, M. Sisay, and E.G. Zeleke, "Incidence of diabetes retinopathy and determinants of time to diabetes retinopathy among diabetes patients at Tikur Anbessa Hospital, Ethiopia: a retrospective follow up study," BMC Research Notes, vol. 11, no. 1, p. 542, Aug. 2018. Available: https://doi.org/10.1186/s13104-018-3660-7.

[7] “retinal structure-Bing images.” https://www.bing.com/images/search? (accessed Apr. 28, 2024).

[8] M. W. Report, “CDC Health Disparities and Inequalities Report —United States , 2011,” vol. 60, 2011.

[9] “6 Diseases Digital Retinal Imaging Can Help Detect Sooner | Fundus.”

https://www.coburntechnologies.com/2016/05/11/retinal-imaging-diseases/ (accessed Mar. 03, 2021).

[10] "hypertensive retinopathy imaging - Bing images." https://www.bing.com/images/search? (accessed Mar. 03, 2021).

[11] Cleveland Clinic, "Retinal vein occlusion (RVO); Causes, symptoms, treatment,"ClevelandClinic. https://my.clevelandclinic.org/health/diseases/14206-retinal-vein-occlusion-rvo (accessed Mar. 03, 2021).

[12] A. E. Gabbey and S. Falck, "Retinal Vascular Occlusion: Causes, Symptoms, and Diagnosis," 2017, 2017. https://www.healthline.com/health/retinal-artery-occlusion (accessed Mar. 03, 2021).

[13] "Retinal Artery Occlusion Fundoscopy - Bing images." https://www.bing.com/images/search? (accessed Mar. 03, 2021).

[14] "glaucomicimage,"unsplash. https://www.wolfeeyeclinic.com/medical-services/glaucoma (accessed Jul. 02, 2023).

[15] K. Shintani, D. L. Shechtman, and A. S. Gurwood, "Review and update: Current treatment trends for patients with retinitis pigmentosa," Optometry, vol. 80, no. 7, pp. 384–401, Jul. 2009, doi: 10.1016/j.optm.2008.01.026.

[16] Mayo Clinic, "Retinoblastoma - Symptoms and causes - Mayo Clinic," Mayo Clinic, 2021. https://www.mayoclinic.org/diseases-conditions/retinoblastoma/symptoms-causes/syc-20351008 (accessed Mar. 03, 2021.

[17] "Eye Health / Macular Pucker / Symptoms & Treatments / Optical Express." Available at: https://www.opticalexpress.co.uk/eye-health/eye-conditions/macular-pucker (accessed Mar. 03, 2024).

[18] "Eye Health / Coats'disease/ Symptoms & Treatments / Optical Express." Available at: https://www.opticalexpress.co.uk/eye-health/eye-conditions/macular-pucker (accessed Mar. 03, 2024).

[19] Abràmoff, Michael D.; Garvin, Mona K.; Sonka, Milan (2010-01-01). "Retinal Imaging and Image Analysis" (https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3131209). IEEE Transactions on Medical Imaging. 3: 169–208. doi:10.1109/RBME.2010.2084567 (https://doi.org/10.1109.

[20] "Fundusphotography imaging - Bing images." https://www.bing.com/images/search? (accessed Mar. 03, 2021).

[21] A. B. Kello and C. Gilbert, "Causes of severe visual impairment and blindness in children in schools for the blind in Ethiopia," Br. J. Ophthalmol., vol. 87, no. 5, pp. 526–530, 2003, doi: 10.1136/bjo.87.5.526.

[22] K. G. Lee, S. J. Song, S. Lee, H. G. Yu, D. Kim, and K. M. Lee, "A deep learning-based framework for retinal fundus image enhancement," PLoS One, vol. 18, no. 3 March, pp. 1–17, 2023, doi: 10.1371/journal.pone.0282416.

[23] "IBM, What is Deep Learning? / IBM," IBM Cloud Learn Hub, 2020. https://www.ibm.com/cloud/learn/deep-learning (accessed Feb. 28, 2021).

[24] M. Varone, D. Mayer, and A. Melegari, "What is Machine Learning? Adefinition," ExpertSystem, 2020. https://expertsystem.com/machine-learning-definition/#:~:text=Adefinition,-ExpertSystemTeam&text=Machinelearningisanapplication,useitlearnforthemselves (accessed Feb. 28, 2021).

[25] A. Mathew, P. Amudha, and S. Sivakumari, "Deep learning techniques: an overview," Adv. Intell. Syst. Comput., vol. 1141, pp. 599–608, 2021, doi: 10.1007/978-981-15-3383-9_54.

[26] L. Alzubaidi et al., Review of deep learning: concepts, CNN architectures, challenges, applications, future directions, vol. 8, no. 1. Springer International Publishing, 2021.

[27] G. Litjens et al., "A survey on deep learning in medical image analysis," Med. Image Anal., vol. 42, pp. 60–88, 2017, doi: 10.1016/j.media.2017.07.005.

[28] K. Weiss, T. M. Khoshgoftaar, and D. D. Wang, A survey of transfer learning, vol. 3, no. 1. Springer International Publishing, 2016.

[29] S. Pouyanfar et al., "A survey on deep learning: Algorithms, techniques, and applications," ACM Comput. Surv., vol. 51, no. 5, 2018, doi: 10.1145/3234150.

[30] R. Arunkumar and P. Karthigaikumar, "Multi-retinal disease classification by reduced deep learning features," Neural Comput. Appl., vol. 28, no. 2, pp. 329–334, 2017, doi: 10.1007/s00521-015-2059-9.

[31] H. Pratt, F. Coenen, D. M. Broadbent, S. P. Harding, and Y. Zheng, "Convolutional neural networks for diabetic retinopathy," Procedia Comput. Sci., vol. 90, pp. 200–205, Jan. 2016.

[32] W. Chen, B. Yang, J. Li, and J. Wang, "An approach to detecting diabetic retinopathy based on integrated shallow convolutional neural networks," IEEE Access, vol. 8, pp. 178552–178562, 2020, doi: 10.1109/ACCESS.2020.3027794.

[33] J. Y. Choi, T. K. Yoo, J. G. Seo, J. Kwak, T. T. Um, and T. H. Rim, "Multi-categorical deep learning neural network to classify retinal images: A pilot study employing small database," PLoS One, vol. 12, no. 11, pp. 1–16, 2017, doi: 10.1371/journal.pone.0187336.

[34] S. S. Chaturvedi, K. Gupta, V. Ninawe, and P. S. Prasad, "Automated Diabetic Retinopathy Grading using Deep Convolutional Neural Network," 2020, [Online]. Available: http://arxiv.org/abs/2004.06334.

[35] T. Sajana, K. Sai Krishna, G. Dinakar, and H. Rajdeep, "Classifying diabetic retinopathy using deep learning architecture," Int. J. Innov. Technol. Explor. Eng., vol. 8, no. 6 Special Issue 4, pp. 1273–1277, 2019, doi: 10.35940/ijitee.F1261.0486S419.

[36] B. K. Triwijoyo, W. Budiharto, and E. Abdurachman, "The Classification of Hypertensive Retinopathy using Convolutional Neural Network," Procedia Comput. Sci., vol. 116, pp. 166–173, 2017, doi: 10.1016/j.procs.2017.10.066.

[37] M. Shaban et al., "A convolutional neural network for the screening and staging of diabetic retinopathy," PLoS One, vol. 15, no. 6 June, pp. 1–13, 2020, doi: 10.1371/journal.pone.0233514.

[38] B. Tymchenko, P. Marchenko, and D. Spodarets, "Deep learning approach to diabetic retinopathy detection," ICPRAM 2020 - Proc. 9th Int. Conf. Pattern Recognit. Appl. Methods, pp. 501–509, 2020, doi: 10.5220/0008970805010509.

[39] M. D. Abràmoff et al., "Improved automated detection of diabetic retinopathy on a publicly available dataset through integration of deep learning," Investig. Ophthalmol. Vis. Sci., vol. 57, no. 13, pp. 5200–5206, 2016, doi: 10.1167/iovs.16-19964.

[40] R. Gargeya and T. Leng, "Automated Identification of Diabetic Retinopathy Using Deep Learning," Ophthalmology, vol. 124, no. 7, pp. 962–969, 2017, doi: 10.1016/j.ophtha.2017.02.008.

[41] L. Cen et al., "Automatic detection of 39 fundus diseases and conditions in retinal photographs using deep neural networks," Nat. Commun., no. 2021, pp. 1–13, doi: 10.1038/s41467-021-25138-w.

[42] K. M. Kim et al., "Development of a fundus image-based deep learning diagnostic tool for various retinal diseases," J. Pers. Med., vol. 11, no. 5, 2021, doi: 10.3390/jpm11050321.

[43] S. S. M. Sheet, T. S. Tan, M. A. As'ari, W. H. W. Hitam, and J. S. Y. Sia, "Retinal disease identification using upgraded CLAHE filter and transfer convolution neural network," ICT Express, vol. 8, no. 1, pp. 142–150, 2022, doi: 10.1016/j.icte.2021.05.002.

[44] S. Qummar et al., "A Deep Learning Ensemble Approach for Diabetic Retinopathy Detection," IEEE Access, vol. 7, pp. 150530–150539, 2019, doi: 10.1109/ACCESS.2019.2947484.

[45] L. Quaranta, F. Calefato, and F. Lanubile, "KGTorrent: A Dataset of Python Jupyter Notebooks from Kaggle," pp. 550–554, 2021, doi: 10.1109/MSR52588.2021.00072.

[46] J. Terstriep, “Keras Spatial,” 2019.

[47] J. Qin, W. Pan, X. Xiang, Y. Tan, and G. Hou, “Journal of,” Ecol. Inform., p. 101093, 2020, doi: 10.1016/j.ecoinf.2020.101093.

[48] S. Serte and A. Serener, “Deep learning in medical imaging: A brief review,” no. June, pp. 1–14, 2020, doi: 10.1002/ett.4080.

[49] Cen, Ling-ping and Ji, Jie and Lin, Jian-wei and Ju, Si-tong and Lin, Hong-jie and Li, Tai-ping and Wang, Yun and Yang, Jian-feng and Liu, Yu-fen and Tan, Shaoying and Tan, Li and Li, Dongjie and Wang, Yifan and Zheng, Dezhi and Xiong, Yongqun and Wu, Hanfu and Jiang, Jingjing and Wu, Zhenggen and Huang, Dingguo and Shi, Tingkun and Chen, Binyao and Yang, Jianling and Zhang, Xiaoling and Luo, Li and Huang, Chukai and Zhang, Guihua and Huang, Yuqiang and Ng, Tsz Kin and Chen, Haoyu and Chen, Weiqi and Pang, Chi Pui and Zhang, Mingzhi, "Automatic detection of 39 fundus diseases and conditions in retinal photographs using deep neural networks",journal = Nature Communications, pages = 1–13, 2021, doi = 10.1038/s41467-021-25138,

[50] P. D. Moral, S. Nowaczyk and S. Pashami, "Why Is Multiclass Classification Hard?," in IEEE Access, vol. 10, pp. 80448-80462, 2022, doi: 10.1109/ACCESS.2022.3192514.

[51] D. P. Kingma and J. L. Ba, “Adam: A method for stochastic optimization,” 3rd Int. Conf. Learn. Represent. ICLR 2015 - Conf. Track Proc., pp. 1–9, 2015.

[52] Kim, K.M.; Heo, T.-Y.; Kim, A.; Kim, J.; Han, K.J.; Yun, J.; Min, J.K.Development of a Fundus Image-Based Deep Learning Diagnostic Tool for Various Retinal Diseases. J. Pers. Med. 2021, 11, 321. https://doi.org/10.3390/jpm11050321

[53] Hossin, M. 1 and Sulaiman, M.N. 2 1. "Technology and I. Technology." Vol. 5, no. 2, 2015, pp. 1–11.

[54] A. B. Kello and C. Gilbert, "Causes of severe visual impairment and blindness in children in schools for the blind in Ethiopia," *Br. J. Ophthalmol.*, vol. 87, no. 5, pp. 526–530, 2003, doi: 10.1136/bjo.87.5.526.

Appendix A

GUI Source Code

imported neccesssary libraries

```
[ ]: import tkinter as tk
     import tkinter.ttk as ttk
     from tkinter import *
     from functools import partial
     from keras.models import load_model,Model
     from tensorflow.keras.optimizers import Adam
     import cv2
     import numpy as np
     from keras import backend as k
     import time
     from tkinter.filedialog import askopenfile,askopenfilename
     from PIL import Image,ImageTk
     from PyQt5 import QtCore, QtGui, QtWidgets
```

configuring the root/the main GUI frame

```
[2]: root=tk.Tk()
     root.geometry("1280x720")
     #root.title('multi-disease classifier system  from retinal')
     bgimg= tk.PhotoImage(file = r"C:/Users/Admnistor/back.png")
```

```
limg= Label(root, i=bgimg)
limg.pack()


#root.configure(bg='#007700')
#bg=PhotoImage(file=r"C:/Users/Admnistor/Desktop/ju.png")
filename = r'logo.ico'
#bg=PhotoImage(file="C:/Users/Admnistor/Desktop/ked.jpg")
img = Image.open(filename)
img.save('logo.ico')
root.iconbitmap('logo.ico')
root.resizable(False, False)
tit = tk.Label(root, text="Portable multi-disease classifier system␣
 ↪from retinal images",
               padx=16, pady=1, font=("courier", 20)).pack()


top_frame=Frame(root,bd=10)
top_frame.pack()
middle_frame=Frame(root,bd=10)
middle_frame.pack()
button_frame=Frame(root,bd=10)
button_frame.pack()
bottom_frame=Frame(root,bd=10)
bottom_frame.pack()
notification_frame=Frame(root,bd=10)
notification_frame.pack()
```

user defiened functions to acess saved model weights

```
#user defiened functions
#openfile from file disk
def open_file(initialdir='/'):
    file_path=askopenfilename(initialdir=initialdir,
```

```
                                filetypes=[("model weights",'*.hdf5')])
    dialog_var.set(' weight browsed but not yet loaded')
    h5_var.set(file_path)
    return file_path
```

defiened function to load the model weights

[4]:
```
def load_weights():
    dialog_var.set('loading weights.....')
    weight_path=h5_entry.get()
    global model,height,width,channel
    model=load_model(weight_path)
    model.summary()
    load_input=model.input
    input_shape=list(load_input.shape)
    height=int(input_shape[1])
    width=int(input_shape[2])
    channel=int(input_shape[3])
    print(height,width,channel)
    dialog_var.set('weight loaded')
    return
```

defiened function to browse image to be classified

[5]:
```
#openimage from file disk
def open_image(initialdir='/'):
    file_path=askopenfilename(initialdir=initialdir,
                              filetypes=[("Image File", '*.*')])
    dialog_var.set(' image browsed but not yet loaded ')
    img_var.set(file_path)

    image=Image.open(file_path)
    image=image.resize((224,224))
```

```
    photo= ImageTk.PhotoImage(image)


    img_label=Label(middle_frame,image=photo,padx=5,pady=5)
    img_label.image=photo
    img_label.grid(row=3,column=1)
    return file_path
```

defiened function to load browsed image

[6]:
```
def load_image():
    dialog_var.set('image loading ...')
    path=img_entry.get()
    global imgs
    #int channel,
    #if channel == 1:
        #imgs=cv2.imread(path,cv2.IMREAD_GRAYSCALE)
    #else:
    imgs=cv2.imread(path)
    imgs=cv2.resize(imgs,(224,224))
    imgs=imgs.reshape(1,224,224,3).astype("float32")
    imgs=np.array(imgs)/255
    print(imgs.shape)
    dialog_var.set('image loaded!,now test it')
    return
```

appliying test on loaded image function

[7]:
```
def test_image():
    model.compile(loss="categorical_crossentropy",
                  metrics=["accuracy"],optimizer="adam")
    k.set_value(model.optimizer.lr,1e-5)
    old=time.time()
    #predictions=model.predict_classes(imgs)
```

```
    predictions=np.argmax(model.predict(imgs),axis=-1)
    new=time.time()
    print(predictions)
    print(new-old)
    result_text = "output class: "+str(predictions)
    test_result_var.set(result_text)
    dialog_var.set('you done it')
```

labels, buttons definition on top of root frame

[8]:
```
#top frame
btn_h5_fopen=Button(top_frame,text='browse weights',font=("courier",
 ↪14),
                    command= lambda: open_file(h5_entry.get()),
                    bg='#88324f',fg='white')
btn_h5_fopen.grid(row=2,column=1)
h5_var=StringVar()
h5_var.set("/")
h5_entry=Entry(top_frame,textvariable=h5_var, width=40)
h5_entry.grid(row=2,column=2)

btn_h5_confirm=Button(top_frame,text='load weights',
                      font=("courier", 14),
                      command=load_weights,bg='#88324f',
                      fg='white')
btn_h5_confirm.grid(row=2, column=4)

btn_img_fopen=Button(top_frame,text='browse image',
                     font=("courier", 14),
                     command=lambda: open_image(img_entry.get()),
                     bg='#88324f',fg='white')
btn_img_fopen.grid(row=7, column=1)
```

```
img_var=StringVar()
img_var.set('/')
img_entry=Entry(top_frame,textvariable=img_var,width=40)
img_entry.grid(row=7,column=2)

btn_img_confirm=Button(top_frame,text='Load image',
                       font=("courier", 14),
                       command=load_image,bg='#88324f',
                       fg='white')
btn_img_confirm.grid(row=7,column=4)

ml=LabelFrame(middle_frame,font=("courier",10),bg='gray',fg='white',
              text='browse images shown below').grid(row=1,column=1)

btn_test=Button(bottom_frame,text='test_image',
                font=("courier", 14),
                command=test_image, bg='#ffff00',fg='red')
btn_test.pack()



test_result_var=StringVar()
test_result_var.set("your result shown here")
test_result_label=Label(bottom_frame,font=('courier',20),height=3,
                        textvariable=test_result_var,bg='white',
                        fg='purple').pack()
```

notification frame, wellcoming and finished message defination

```
dialog_var=StringVar()
dialog_var.set('wellcome to your eye care')
label_frame1=LabelFrame(notification_frame,text=" Notification Box",
```

```
                        font=("courier",22), bg='white')
label_frame1.pack()

top_label=Label(label_frame1,font=("courier",18),height=2,
                textvariable=dialog_var,fg='red',bg='lightcyan')
top_label.pack()

tite = tk.Label(root, text="by kedir Jundi", padx=16,
                pady=1, font=("courier", 14)).pack()

top_frame.mainloop()
print('finished')
```

Appendix B

Transfer Learning Source code

```
[ ]: ! nvidia-smi
```

```
[ ]: from google.colab import drive
     drive.mount('/content/drive')
```

```
[ ]: import os
     from tensorflow import keras
     from keras.models import Model
     import tensorflow as tf
     from tensorflow.keras.optimizers import Adam,RMSprop
     #from keras.applications.mobilenet_v2 import MobileNetV2,␣
      ↪preprocess_input
     from keras.applications.resnet import ResNet50, preprocess_input
     from keras.preprocessing.image import ImageDataGenerator
     from keras.callbacks import ModelCheckpoint, EarlyStopping
     from keras.layers import Dense, Dropout, Flatten
     from pathlib import Path
     import numpy as np
     import h5py
     import tensorflow as tf
     import cv2
```

```
from PIL import Image
```

```
[ ]: BATCH_SIZE = 32
train_generator = ImageDataGenerator(#rescale=1./255,
                                     rotation_range=90,
                                     #brightness_range=[0.1, 0.7],
                                     #width_shift_range=0.1,
                                     #height_shift_range=0.1,
                                     #horizontal_flip=True,
                                     #vertical_flip=True,
                                     validation_split=0.20,
                                    ␣
 ↪preprocessing_function=preprocess_input)
test_generator = ImageDataGenerator(#rescale=1./255,
                                    ␣
 ↪preprocessing_function=preprocess_input)
```

```
[ ]: download_dir = Path('/content/drive/MyDrive/dr_5class')
```

```
[ ]: train_data_dir = download_dir/'/content/drive/MyDrive/dr_5class/
 ↪train-data115'
#valid_data_dir = download_dir/'/content/drive/MyDrive/overallclass/
 ↪validation_data'
test_data_dir = download_dir/'/content/drive/MyDrive/dr_5class/
 ↪test_data115'
class_subset = sorted(os.listdir(download_dir/'/content/drive/MyDrive/
 ↪dr_5class/train-data115'))[:5] # Using only the first 5 classes
traingen = train_generator.flow_from_directory(train_data_dir,
                                               target_size=(224, 224),
                                               class_mode='categorical',
                                               classes=class_subset,
                                               subset='training',
                                               batch_size=BATCH_SIZE,
```

```
                                                shuffle=True,
                                                seed=42)
validgen = train_generator.flow_from_directory(train_data_dir,
                                                target_size=(224, 224),
                                                class_mode='categorical',
                                                classes=class_subset,
                                                subset='validation',
                                                batch_size=BATCH_SIZE,
                                                shuffle=True,
                                                seed=42)
testgen = test_generator.flow_from_directory(test_data_dir,
                                              target_size=(150, 150),
                                              class_mode=None,
                                              classes=class_subset,
                                              batch_size=1,
                                              shuffle=False,
                                              seed=42)
```

```
: def create_model(input_shape, n_classes,optimizer='Adam', fine_tune=0):
    # optimizer='rmsprop'
    conv_base = ResNet50(include_top=False,
                         weights='imagenet',
                         input_shape=input_shape)
    if fine_tune > 0:
        for layer in conv_base.layers[:-fine_tune]:
            layer.trainable = False
    else:
        for layer in conv_base.layers:
            layer.trainable = False
    top_model = conv_base.output
    top_model = Flatten(name="flatten")(top_model)
    #top_model = Dense(256, activation='relu')(top_model)
```

```
    top_model = Dense(1, activation='relu')(top_model)
    top_model = Dropout(0.2)(top_model)
    #x=Flatten()(conv_base.output)
    #prediction=Dense(n_classes,activation='softmax')(x)
    output_layer = Dense(n_classes, activation='softmax')(top_model)
    #output_layer=Flatten(name="flatten")(output_layer)
    # Group the convolutional base and new fully-connected layers into␣
↪a Model object.
    model = Model(inputs=conv_base.input, outputs=output_layer)
    # Compiles the model for training.
    model.compile(optimizer=optimizer,
                  loss='categorical_crossentropy',
                  metrics=['accuracy'])
    return model
```

```
input_shape = (224, 224,3)
optim_1 = Adam()#learning_rate=0.00001
n_classes=5
n_steps = traingen.samples // BATCH_SIZE
n_val_steps = validgen.samples // BATCH_SIZE
n_epochs =50
# First we'll train the model without Fine-tuning
resnet_model = create_model(input_shape, n_classes, optim_1,␣
↪fine_tune=0)
resnet_model.summary()
```

```
input_shape = (224, 224,3)
optim_1 = Adam()#learning_rate=0.00001
n_classes=5
n_steps = traingen.samples // BATCH_SIZE
n_val_steps = validgen.samples // BATCH_SIZE
n_epochs =50
```

```
# First we'll train the model without Fine-tuning
resnet_model = create_model(input_shape, n_classes, optim_1,␣
↪fine_tune=0)
resnet_model.summary()
```

[ ]:
```
MODEL_DIR = "/content/drive/My Drive/temp"
if not os.path.exists(MODEL_DIR):  #If the directory does not exist,␣
↪create it.
    os.makedirs(MODEL_DIR)
```

[ ]:
```
from livelossplot.inputs.keras import PlotLossesCallback
plot_loss_1 = PlotLossesCallback()
filepath=os.path.join(MODEL_DIR, "model-{epoch:02d}.h5")
#filepath='t6_model_v1_weights_best.hdf5'
# ModelCheckpoint callback - save best weights
t7_checkpoint_1 = ModelCheckpoint(filepath,
                                  save_best_only=True,
                                  verbose=1)
# EarlyStopping
early_stop = EarlyStopping(monitor='val_loss',
                           patience=10,
                           restore_best_weights=True,
                           mode='min')
```

[ ]:
```
%%time
resnet_history =resnet_model.fit(traingen,
                                 batch_size=BATCH_SIZE,
                                 epochs=n_epochs,
                                 validation_data=validgen,
                                 steps_per_epoch=n_steps,
                                 validation_steps=n_val_steps,
                                 callbacks=[t7_checkpoint_1,␣
↪early_stop, plot_loss_1],
```

```
                                    verbose=1)
```

[ ]:
```
model_json=resnet_model.to_json()
filepath=os.path.join(MODEL_DIR, "model.json")
with open('model.json',"w") as json_file:
  json_file.write(model_json)
resnet_model.save_weights(MODEL_DIR,"model-{epoch:02d}.h5")
```

[ ]:
```
#plot the result
import matplotlib.pyplot as plt
from matplotlib import *
print(resnet_history.history.keys())
plt.figure(1,figsize=(15,15))
plt.subplot(2,2,1)
plt.plot(resnet_history.history['accuracy'])
plt.plot(resnet_history.history['val_accuracy'])
plt.title('model_accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['train','valid'])
plt.subplot(2,2,2)
plt.plot(resnet_history.history['loss'])
plt.plot(resnet_history.history['val_loss'])
plt.title('val_loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train','valid'])
plt.show()
```

[ ]:
```
# Generate predictions
resnet_model.load_weights(os.path.join(MODEL_DIR,"model-01.h5")) ␣
 ↪#Specify the model of
```

```
#resnet_model.load_weights('t5_model_v1_weights_best.hdf5') #␣
↪initialize the best trained weights
true_classes = testgen.classes
class_indices = traingen.class_indices
class_indices = dict((v,k) for k,v in class_indices.items())
resnet_model_preds_ft = resnet_model.predict(testgen)
resnet_model_pred_classes_ft = np.argmax(resnet_model_preds_ft, axis=1)
```

[ ]:
```
from sklearn.metrics import accuracy_score
resnet_model_acc_ft = accuracy_score(true_classes,␣
↪resnet_model_pred_classes_ft)
print("resnet_model Model Accuracy with Fine-Tuning: {:.2f}%".␣
↪format(resnet_model_acc_ft * 100))
```

[ ]:
```
import seaborn as sns
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
import sklearn.metrics as metrics
from sklearn.metrics import plot_confusion_matrix
from sklearn.metrics import accuracy_score
import matplotlib.pyplot as plt
import pylab as pl
import pandas as pd
# Get the names of the five classes
class_names = testgen.class_indices.keys()
y_true=true_classes
y_pred=resnet_model_pred_classes_ft
cm=confusion_matrix(y_true, y_pred)
print(cm)
def print_confusion_matrix(cm, class_names, figsize = (10,7),␣
↪fontsize=14):
    df_cm = pd.DataFrame(
```

```
        cm, index=class_names, columns=class_names,
    )
    fig = plt.figure(figsize=figsize)
    try:
        heatmap = sns.heatmap(df_cm, annot=True, fmt="d")
    except ValueError:
        raise ValueError("Confusion matrix values must be integers.")
    heatmap.yaxis.set_ticklabels(heatmap.yaxis.get_ticklabels(),␣
↪rotation=0, ha='right', fontsize=fontsize)
    heatmap.xaxis.set_ticklabels(heatmap.xaxis.get_ticklabels(),␣
↪rotation=45, ha='right', fontsize=fontsize)
    plt.ylabel('True label')
    plt.xlabel('Predicted label')
    return fig
cm = np.array()
fig = print_confusion_matrix(cm, ['dr ','glaucoma',␣
↪'hr','normal','occlusion'])

#plot_heatmap(true_classes, resnet_pred_classes_ft, class_names,ax,␣
↪title="Transfer Learning (ResNet) with Fine-Tuning")
from sklearn.metrics import classification_report
print(classification_report(y_true, y_pred))
```

Appendix C

Sample Code for 5-fold Model Evaluation

```
#Trianing the model with 5 fold validation
from PIL import Image
import os
import numpy as np
from sklearn.model_selection import KFold
from keras.applications.resnet50 import preprocess_input
from sklearn.preprocessing import LabelEncoder
from keras.utils import to_categorical
import matplotlib.pyplot as plt
from keras.callbacks import ModelCheckpoint,

# Set the number of folds for cross-validation
k = 5

# Set the path to your dataset
data_path = "/content/drive/MyDrive/occsub/train"

# Load the dataset and labels
dataset = []
labels = []

# Assuming you have a folder for each class in your dataset
classes = os.listdir(data_path)

# Initialize the label encoder
label_encoder = LabelEncoder()

for class_name in classes:
    class_path = os.path.join(data_path, class_name)
    for img_name in os.listdir(class_path):
        img_path = os.path.join(class_path, img_name)
        try:
            img = Image.open(img_path)
            img = img.convert("RGB")  # Convert the image to RGB format
            img = img.resize((224, 224))  # Resize the image to the desired input shape
            img = np.array(img)  # Convert PIL Image to NumPy array
            dataset.append(img)
            labels.append(class_name)
        except Exception as e:
            print(f"Error processing image: {img_path}")
            print(e)

# Convert the dataset and labels to NumPy arrays
dataset = np.array(dataset)
labels = np.array(labels)

# Encode the class labels as integers
labels = label_encoder.fit_transform(labels)
```

5 Fold cross validation

```
# Encode the class labels as integers
labels = label_encoder.fit_transform(labels)

# Perform k-fold cross-validation
kf = KFold(n_splits=k, shuffle=True)
fold = 1

# Initialize lists to store scores and histories
train_scores = []
val_scores = []
train_histories = []
val_histories = []
n_epochs = 50

for train_index, val_index in kf.split(dataset, labels):
    print(f"Fold {fold}")
    fold += 1

    # Use format() to insert the fold number into the filepath
    checkpoint = ModelCheckpoint(filepath=f"model_fold_{fold}.h5", monitor='val_loss', verbose=1, save_best_only=True)
    early_stop = EarlyStopping(monitor='val_loss', patience=5, verbose=1)

    # Split the data into training and validation sets for the current fold
    train_data, val_data = dataset[train_index], dataset[val_index]
    train_labels, val_labels = labels[train_index], labels[val_index]

    # Preprocess the input data
    train_data = preprocess_input(train_data)
    val_data = preprocess_input(val_data)

    # Create and compile the model for each fold
    resnet_model = create_model((224, 224, 3), len(classes))
    resnet_model.compile(optimizer="adam", loss="categorical_crossentropy", metrics=["accuracy"])

    # Train the model
    resnet_history = resnet_model.fit(train_data, to_categorical(train_labels, num_classes=len(classes)),
                                      batch_size=BATCH_SIZE,
                                      epochs=n_epochs,
                                      validation_data=(val_data, to_categorical(val_labels, num_classes=len(classes))),
                                      callbacks=[checkpoint, early_stop],
                                      verbose=1)

    # Evaluate the model on training and validation data
    train_score = resnet_model.evaluate(train_data, to_categorical(train_labels, num_classes=len(classes)))
    val_score = resnet_model.evaluate(val_data, to_categorical(val_labels, num_classes=len(classes)))

    # Append scores to lists
    train_scores.append(train_score[1])
    val_scores.append(val_score[1])

    # Append histories to lists
    train_histories.append(resnet_history.history['accuracy'])
    val_histories.append(resnet_history.history['val_accuracy'])
```

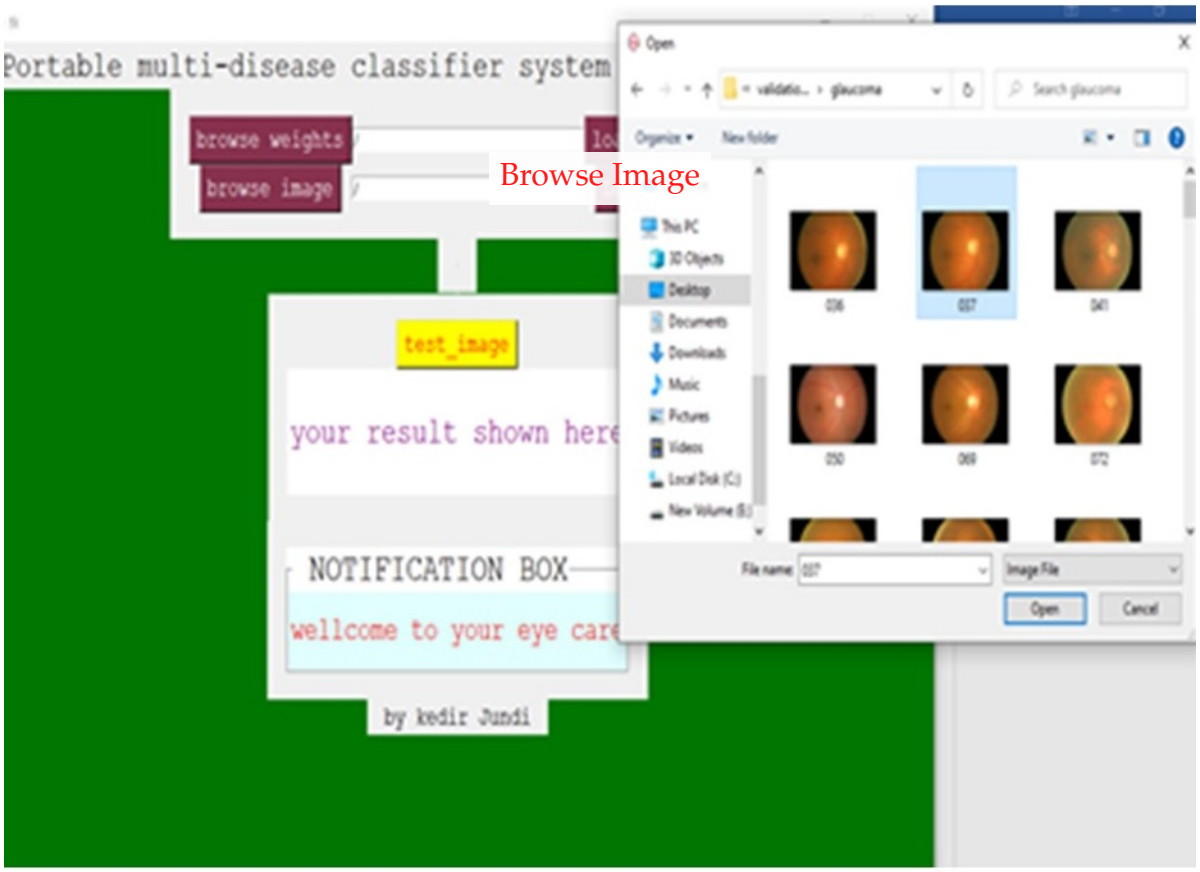
5 Fold cross validation

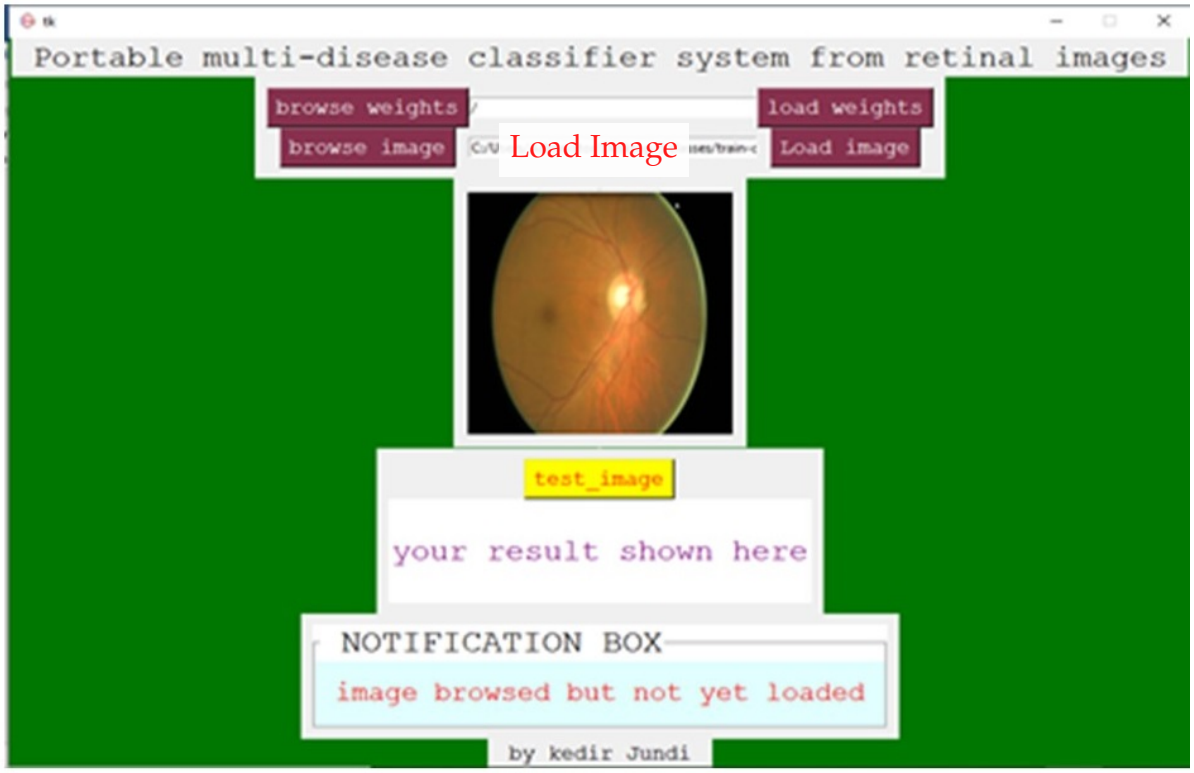
```
# Calculate average scores
avg_train_score = np.mean(train
avg_val_score = np.mean(val_sc
# Create a table to display the fold-wise scores
table = [["Fold", "Training Accuracy", "Validation Accuracy"]]
folds = ['Fold 1', 'Fold 2', 'Fold 3', 'Fold 4', 'Fold 5']
for i in range(len(folds)):
    table.append([folds[i], f"{train_scores[i]:.2f}", f"{val_scores[i]:.2f}"])
# Print the table
print(tabulate(table, headers="firstrow", tablefmt="grid"))
# Print average scores
print(f"\nAverage Training Score: {avg_train_score:.2f}")
print(f"Average Validation Score: {avg_val_score:.2f}")
```

5 Fold cross validation

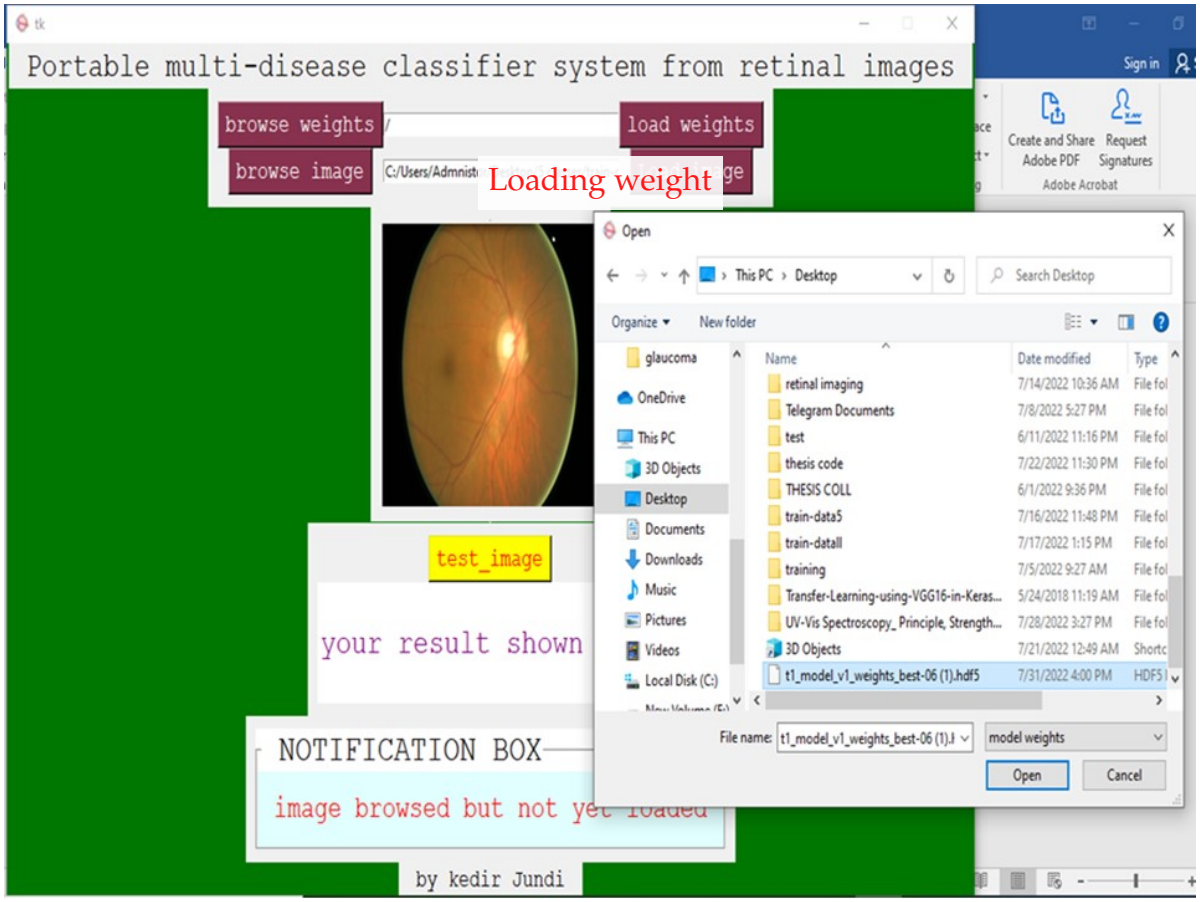
Appendix D

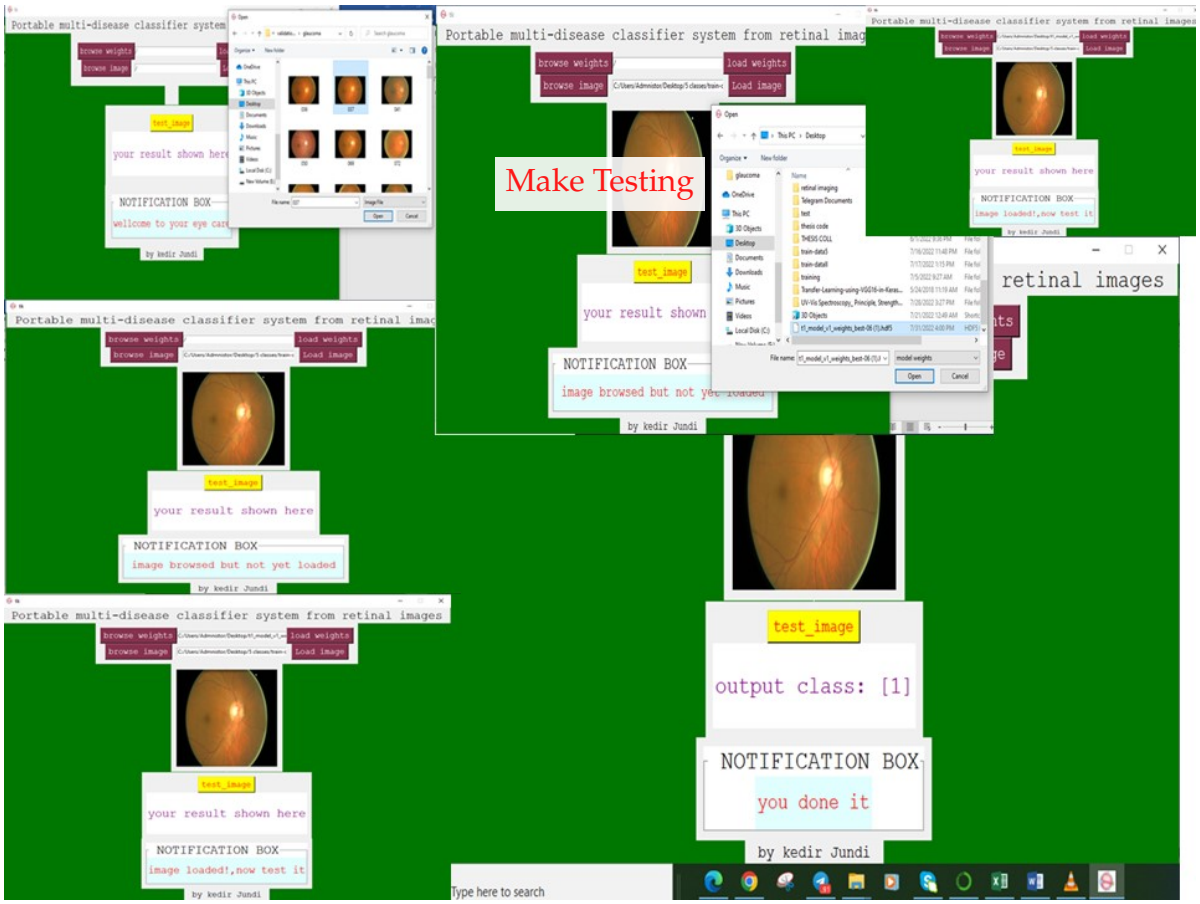
Snapshot Captures for Graphical User Interface Result




Portable multi-disease classifier system from retinal images
browse weights
load weights
browse image
Load image
Load Image
test_image
your result shown here
NOTIFICATION BOX
image browsed but not yet loaded
by kedir Jundi


Portable multi-disease classifier system from retinal images
browse weights
load weights
browse image
Load image
Browse weight
test_image
your result shown here
NOTIFICATION BOX
weight loaded
by kedir Jundi
Portable multi-disease classifier system from retinal ir
browse weights
load weights
browse image
Load image
test_image
your result shown here
NOTIFICATION BOX
image loaded!,now test it
by kedir Jundi


Portable multi-disease classifier system from retinal images
browse weights
load weights
browse image
Loading weight
test_image
your result shown
NOTIFICATION BOX
image browsed but not yet loaded
by kedir Jundi
Open
This PC › Desktop
Search Desktop
Organize
New folder
glaucoma
OneDrive
This PC
3D Objects
Desktop
Documents
Downloads
Music
Pictures
Videos
Local Disk (C:)
Name
Date modified
Type
retinal imaging
Telegram Documents
test
thesis code
THESIS COLL
train-data5
train-datall
training
Transfer-Learning-using-VGG16-in-Keras...
UV-Vis Spectroscopy_ Principle, Strength...
3D Objects
t1_model_v1_weights_best-06 (1).hdf5
File name:
model weights
Open
Cancel
Sign in
Create and Share Adobe PDF
Request Signatures
Adobe Acrobat


Make Testing
Portable multi-disease classifier system from retinal images
test_image
output class: [1]
NOTIFICATION BOX
you done it
by kedir Jundi
Type here to search