RetinalNet-500: A newly developed CNN Model for Eye Disease Detection

by

Sadikul Alim Toki
18101467
Sohanoor Rahman
21141072
SM Mohtasim Billah Fahim
18101147
Abdullah Al Mostakim
19301268

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree B.Sc. in Computer Science

Department of Computer Science and Engineering
Brac University
May 2022

Declaration

It is hereby declared that

1. The thesis submitted is our own original work while completing degree at Brac University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. We have acknowledged all main sources of help.

Student's Full Name & Signature:

	Sohanoor Rahman
Sadikul Alim Toki	Sohanoor Rahman
18101467	21141072
Fahim	Abdullah Al Mostakim
SM Mohtasim Billah Fahim	Abdullah Al Mostakim
18101147	19301268

Approval

The thesis titled "**RetinalNet-500: A newly developed CNN Model for Eye Disease Detection**" submitted by

1. Sadikul Alim Toki (ID: 18101467)
2. Sohanoor Rahman (ID: 21141072)
3. SM Mohtasim Billah Fahim (ID: 18101147)
4. Abdullah Al Mostakim (ID: 19301268)

Of Spring, 2022 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science on May, 2022.

Examining Committee:

Supervisor:
(Member)

Md. Khalilur Rhaman

Md. Khalilur Rhaman, PhD

Associate Professor
Department of Computer Science and Engineering
BRAC University

Co-Supervisor:
(Member)

Faisal Bin Ashraf

Lecturer
Department of Computer Science and Engineering
BRAC University

Program Coordinator:
(Member)

Md. Golam Rabiul Alam, PhD

Associate Professor
Department of Computer Science and Engineering
BRAC University

Head of Department:
(Chair)

Sadia Hamid Kazi, PhD

Chairperson and Associate Professor
Department of Computer Science and Engineering
BRAC University

Ethics Statement

In order to successfully complete an undergraduate thesis, it is imperative that the thesis adheres strictly to the rules and regulations of the university, as well as ethical principles for conducting research. We have incorporated original data into our thesis. We have carefully checked our references and citations. The four co-authors of the paper all acknowledge responsibility for any violation of the thesis rule. In order to solve problems we used several conference paper, journal paper, and Coursera/EdX videos. As a result, we'd like to express our appreciation to everyone who has helped us throughout the journey. None of the unethical practices have been used in the completion of our thesis. This has been conducted in accordance with the BRAC university's ethics code.

Abstract

Fundus images are commonly used by medical experts like ophthalmologists, which are very helpful in detecting various retinal disorders. They used this to diagnose the different types of eye diseases like Cataracts, Diabetic Retinopathy, Glaucoma etc. These fundus images can be also used for the prediction of the severity of the diseases and can provide early signs or warnings. Recently, different machine learning algorithms are playing a vital role in the field of medical science, and it is no different in Ophthalmology either. In this research, we aim to automatically classify healthy and diseased retinal fundus images using deep neural networks. Because deep learning is an excellent machine learning algorithm, which has proven to be very accurate in computer vision problems. In our research, we used convolutional neural networks(CNN) to classify the retinal images whether they are healthy or not.

Keywords: Retinal Diagnosis, Fundus Images, CNN, Deep Learning, ML

Dedication

In dedication to our families and team members, we have written this paper. In making this project a success, the support of the family and the tenacity of the team members played a critical role. Throughout the writing of our thesis, our esteemed supervisor and co-supervisor—who has been a steady source of guidance and advice throughout—were an indispensable part of the process. Their contributions are also gratefully acknowledged in this document.

Acknowledgement

Firstly, all praise to the Great Allah for whom our thesis have been completed without any major interruption.

Secondly, to our supervisor Dr. Md. Khalilur Rhaman sir and our co-supervisor Faisal Bin Ashraf sir for their kind support and advice in our work. They helped us whenever we needed them.

Thirdly, MIUCC Conference and the whole judging panel of that conference. Where our paper was accepted, all the reviews they gave helped us a lot in our later works.

And finally to our parents without their throughout support it may not be possible. With their kind support and prayer we are now on the verge of our graduation.

Table of Contents

List of Figures

List of Tables

Nomenclature

The next list describes several symbols & abbreviation that will be later used within the body of the document

β	Beta
∞	Infinity
max	Max Function
$\sum$	Summation
tanh	Hyperbolic Tangent Function
θ	Theta
CNN	Convolutional Neural Network
OS	Operating System

Chapter 1

Introduction

The eye is an important organ for a human being which plays an important role compared to other organs of the human body. If our eyes face any diseases, it may cause a great loss of vision. Without vision, it almost becomes difficult to survive for a human being. There are a lot of eye diseases such as diabetic retinopathy, age-related macular degeneration, Cataract, Glaucoma, Amblyopia, etc. To prevent such eye diseases, early detection and timely treatment can help to reduce the risk of great loss of vision.

In many under developed countries, most eye care institutions are not very rich. There is also a lack of proper treatment and ophthalmologists in rural areas. So, it becomes quite tough for the people of rural areas to carry the expenses or cost for better treatment. As the population is increasing, the number of patients with eye diseases is also rapidly increasing. So, it's a community's or government's obligation to improve eye care facilities for its citizens[1].

Using digital image processing and machine learning, it is possible to develop a system that can detect eye diseases from the retinal fundus image. The system will take the retinal fundus images as input. Then from the fundus images, the system can extract and simplify the features of specific eye diseases.

In this research paper, our main focus is to develop a system by using deep learning through CNN which will use the retinal fundus image to identify, extract and evaluate disease-specific characteristics. The system will help to early detect the diseases which allows patients to keep a good quality of vision while avoiding serious vision loss and blindness.

In this paper, we did construct a new model which then was compared with the accuracy of other pre-built models to see how our newly built model perform.

1.0.1 Overview of the retinal diseases we work on

Cataract

A cataract is a cloudy area in the lens of your eye. Cataracts are more prone to develop as you become older. In fact, more than half of persons in their eighties and nineties have cataracts or have had cataract surgery. At first, you may be unaware that you have a cataract. Cataracts, from the other hand, may cause your vision to become hazy, clouded, or less colored as time passes. It's possible that you'll have trouble reading or doing other normal chores. The good news is that cataracts can be removed surgically. Cataract surgery is a safe technique that improves vision in patients who have cataracts. The most of cataracts are age-related and develop as a result of changes that occur in your eyes as you age. Cataracts can form as a result of a variety of factors, such as an eye surgery or injury to treat another eye condition (like glaucoma).

Glaucoma

Glaucoma is a series of illnesses affecting the visual cortex, a nerve in the back of the eye that causes vision loss and blindness. Because the symptoms appear gradually, it's likely that you won't even notice them. The only way to tell if you do have glaucoma is to have a full dilated eye exam. Although there is no cure for glaucoma, early treatment can often prevent future visual loss and protect your vision. Other varieties of glaucoma exist, but open-angle glaucoma is the most prevalent and is what most people hear the term the word glaucoma. Angle-closure glaucoma and congenital glaucoma are fewer common kinds of glaucoma.

Diabetic retinopathy

Diabetic retinopathy is an infection of the eye that can cause diabetics to lose their vision and eventually go blind. The blood vessels in the eyes are damaged (the light-sensitive area of tissue located behind of your eye). If you have diabetes, you should have a fully dilated eye test at least once a year. Diabetic retinopathy might manifest itself without signs at first, but catching it early can help you keep your eyesight. Physical activity, a healthy diet, and medication compliance can all prevent you from getting or delay loss of vision.

1.0.2 CNN in Medical Imaging

A retinal eye condition is often diagnosed through a Fundus image taken with specialized equipment. The abnormality may be detected in the eye fundus image as some abnormal impact on the retina. With the development of computer vision and computer-assisted image identification, breakthroughs in medical science have occurred. Through the use of machine learning and neural networks in image recognition, improvements in image categorization and identification have been possible. CNN will ensure nearly as much precision in image recognition as any existing identification technology. In cases where a patient cannot obtain professional guidance for identification of disease or the hospital needs automated assistance, those images can be utilized to identify specific eye diseases. Using CNN-based classification methods, the need for manually segmenting retinal disease zones is eliminated, resulting in a completely automatic classifier. CNN's extensive integration with medical image identification brings up a whole new set of possibilities for progress.

1.0.3 Computer Vision and Implementation of CNN

The ever-expanding breadth of technical progress has resulted in an incalculable number of computational process-based tasks. One of the most prominent examples is computer vision. This method teaches the machine to recognize photos and videos. The emergence of machine learning, sometimes known as "deep learning," has revolutionized computer vision applications by allowing for large-scale computer vision and pattern recognition. The rise of public picture repositories (ImageNet) and improved computing performance in graphics processing units can be credited for this boom in popularity (GPUs). Convolutional neural networks (CNN), a machine learning-based computer vision technology, are swiftly gaining traction among other classic machine learning techniques for image recognition. The fundamental reason for this is that CNN is faster and more accurate than other approaches like Recurrent Neural Networks (RNN), Long Short-Term Memory (LSTM), and Artificial Neural Networks (ANN) (ANN). To emphasize this point, CNN has a high degree of accuracy in detecting altered or manipulated photos, which is the most important advantage in real-world circumstances. The accuracy of multiple convolutional neural network models has recently increased to the point that, in extreme circumstances, CNN can statistically surpass even humans in terms of identifying a specific breed or species. Computational power and data collection will continue to expand at an exponential rate. As a result, the seemingly endless options for improving CNN and deep learning have been significantly explored.

1.0.4 Research Problem

The classification of retinal eye images is a fascinating computer vision topic with many applications in the medical field. Understanding the retinal blood vessels, for example, is critical for eye doctors to assess eye illnesses like glaucoma and hypertension, which can cause vision problems and blindness if left untreated. Much prior research has concluded that comprehending vascular anomalies from retinal images aids medical physicians in diagnosing and treating stroke, brain injury, carotid atherosclerosis, artery disease, and cerebral amyloid angiopathy early. Some data from these trials suggests that having a specialist examine the retinal eye on a regular basis can improve a patient's quality of life.

The human visual retina is a light-sensitive layer that is critical to human vision. The retina of the eye shares certain anatomical similarities with the central nervous system. The fact that the brain's capillary condition affects the retinal blood vessels produces damage to the retinal eye, indicating systemic microvascular damage linked with disorders like hypertension or diabetes.

Retinal image categorization has gotten a lot of attention in the last decade, which has resulted in a lot of papers. Although various approaches have been proposed, recognizing retinal ocular pictures remains a difficult calculation challenge. The fundus diversity of each patient is one of the challenges.

With technological advancements and image analysis, the procedure of disease detection can be automated, and the patient can be referred to a doctor for additional evaluation. Using developments in electronic computer vision and machine learning, a number of clinical decision support systems are developed specifically to identify diabetic retinopathy and age-related macular disease. Although most of these algorithms are capable of performing as well as human experts, many are focused on diagnosing a specific retinal condition. Most of these models use the retinal picture to identify, extract, and analyze disease-specific characteristics. This necessitates a thorough understanding of the condition as well as time spent developing characteristics for the classifiers. We propose to construct a universal classification algorithm that can identify good retinal images from sick retinal images in this research. The proposed system is based on deep learning and can automatically discover characteristics at various levels from a retinal image training dataset. A broad model like this could be beneficial as a first-level screening tool, particularly in rural areas. This would allow for early diagnosis of retinal illnesses, as well as the avoidance of costly travel and testing for those who do not require further consultation. This is beneficial to both the medical world and rural residents. The diagnostic instrument may be handled by semi-skilled technicians, thanks to an improved user interface, solving the problem of a shortage of skilled medical personnel in rural areas. A fundus camera captures the retinal pictures. Despite the fact that fundus cameras are indeed costly, low-cost cameras are being created that are now inexpensive. This is yet another setup fee that will benefit society as a whole.

Machine Learning

Machine Learning is a data analysis method used by Insights. Machine Learning techniques learn from data to uncover hidden patterns without having to be instructed to look for them. The use of machine learning in disease diagnosis has exploded in recent years. These software algorithms perform by identifying patterns in data in photos at various levels and matching them to diseases that are known to exist. As evidenced by the academic literature, supervised learning is being employed for the early recognition and characterization of eye illnesses such as cataracts, conjunctivitis, and diabetic retinopathy. We describe various research findings in the part on related work where machine prognosis is equivalent to that of human experts for particular eye conditions.

Neural Network

A CNN (Convolutional Neural Network) is a form of neural network which can recognize structural characteristics in a picture. By allowing filtration to slide through the image sequence and perform pattern matching, the CNN is able to capture the sequence at any place across the retina. The stride determines how far the filtration must move across the image as it matches the image pattern. CNN models are made up of self-learning weight matrices that are built into the processing units. Every neuron receives some inputs, does a dot product with weights and biases, and optionally performs an activation function. From raw picture frames on one end to category scores on the other, the entire network employs a single algorithm as follows. Because the inputs are images, the CNN design can encode specific attributes. This minimizes the number of variables in the network and makes the forward algorithm more efficient to implement.
The research is aiming to address the following question:
What will be the research goal of our developed deep learning model (CNN)?
The goal of our research is to detect retinal eye disorders using fundus images through creating a deep learning network (CNN) model.

1.0.5 Research Objective

The goal of this study is to create a proper ensemble learning model which can determine between healthy and diseased retinal pictures. The proposed system is based on deep learning and can automatically discover characteristics at various levels from a retinal image training dataset. The objectives of this research are:

1. To evaluate the deep learning model using the ensemble learning technique.
2. To offer recommendations on improving the deep learning model.
3. To detect multiple eye diseases using a single model.

Chapter 2

Related Work

There has been a good number of research papers in our targeted field. The most common eye diseases studied with fundus images are Cataracts, Diabetic Retinopathy and Glaucoma.

Amongst the papers we studied, only [2] suggests a multiple eye disease detection model using CNN. Their suggested model can predict Glaucoma and Diabetic Retinopathy. Two separate CNN models were trained for two separate diseases. They obtained an accuracy of around 80%. They further added that the model could be improved through parameter tuning.

For Cataract detection, we have studied [3-5] in depth. In [3], they have developed an automated cataract grading system through balancing lumination problems in the dataset and building an eight-layered Deep Convolutional Neural Network (DCNN) model. In [4], they followed the same method as in [4], but they added a Deconvolutional Network in order to extract the features more accurately from each convolutional layer. Lastly in [5], the authors trained a DCNN model similar to Res-Net50 architecture and they have achieved a good accuracy without any preprocessing of data.

For Diabetic Retinopathy, the researchers in [6] used transfer learning in order to train their model using ResNet and added an extra layer of Artificial Neural Network (ANN) which is used as a “meta-classifier” for optimizing the results of the previous layers. They did use CLAHE for image enhancement and achieved an accuracy of around 80%. In [9], the authors did develop their own model consisting of five convolutional layers. They also used Min-Max normalization in order to reduce the background noise of the image. They trained their model only with the Green channel image with equalized histogram and avoided padding to preserve the spatial size of the input and output volumes.

In the case of ensemble learning, Huang, Jonathan, et al. [10] the authors applied four different models. They used Vgg12, ResNet50, AclNet, and AclSincNet to enhance the model’s accuracy. Pre-training with audio set data supported the development of all of the models. With all these models, ensemble learning has been achieved, and the validation set of experiments has been used to obtain the results. 83.01% was the highest result when ensemble averaging all models together.

Then, Kumar, Ashnil, et al. [11] Medical Image Classification can be performed by applying an ensemble of fine-tuned CNN algorithms. This method can be used for diagnosis, teaching, and biomedical research. 4166 images were used for the tests, while 6776 images served as training images. For images classification, two different architectures of CNN were used, AlexNet and Google-Net. Individual models and ensemble models have been used to conduct the experiments. Overall, the method generated an accuracy of 96.59%.

Lastly, Mahmoud Smaida, Serhii Yaroshchak.[12] the authors used here three different models. They used the CNN model, VGG-16 model, Inception-v3 model to improve the accuracy. Their models were applied to eye disease datasets ensemble with bagging, boosting, and stacking to improve their performance. 86.43% of accuracy is obtained after combining all the different structures into a bagging ensemble.

Chapter 3

Work Plan

3.1 Proposed Model

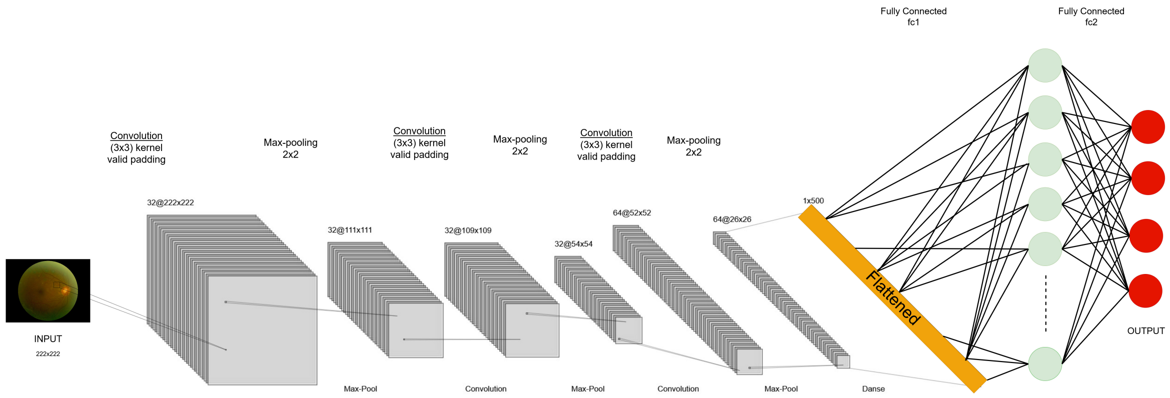


Figure 3.1: Architecture of RetinalNet-500

3.1.1 The CNN Architecture

This figure depicts the various deep network designs. The network has an input layer that accepts pictures with a resolution of 222x222 pixels as input. The design is made up of three sets of convolution, activation, and max pooling layers that are stacked one on top of the other. Then there are two sets of fully linked hidden layers. The final output layer is then added. Valid-padding was used in the convolutional layers.

3.1.2 Convolutional Layer

CNN is more than just a deep neural network with abunch of layers disguised underneath the surface. It's a large network that resembles the visual cortex of the brain in terms of how it analyzes and discerns images. This technique demonstrates the importance of profound layer improvements for information; basically, images processing [17]. The images comprise a 2D matrix with pixels on which the CNN algorithm is executed [18]. CNN uses the brain as a motivation for detecting and classifying images. As we know, a human brain is split into two cell types which are simple cells that work as feature detection and complex cells that combine local features from narrow spatial neighbourhoods. Information that has a location-based relationship with other information is referred to as spatial information. The human brain identifies images by combining all of the local features they can with their eyes, which is how humans can see images. The equation of the convolution operation is:

$$s[t] = (x * w)[t] = \sum_{a=-\infty}^{\infty} x[a]w[a+t]$$

Here, $s=$ feature map, $x=$ input image, $w=$ feature detector or kernel. A convolution is a mathematical term for a function that is derived by integrating two other functions. It describes how another function can change the structure of a function. For example: say am image that is denoted by "x" here. The image is a two-dimensional array of pixels with distinct colour channels. Here we use kernel "w", basically our feature detector by which we obtain the output after applying the feature map. A feature map is a technique that determines how similar two signals are, and this is a result of the convolution layer. A feature detector or filter is used to identify the edges of a picture. The whole convolution operation is responsible for calculating the image's edges. In our model, the first and second convolutional layers are made up of 32 3x3 filters. With no striding, each filter convolves across the input picture. The third convolutional layer is made up of 64 3x3 filters. Each filter convolves over the preceding layer's output with no striding as the previous ones.

Activation Function

The ReLU activation function is inserted between the three sets of repeated layers to assist identify edges more thoroughly. Furthermore, ReLU was applied after the first dense layer, while softmax was employed after the second and final dense layers. Rectified linear units or ReLU is the most advanced and commonly used functions among the ones discussed. ReLU is a non-linear function that acts as a linear function. This dramatically helps the computational process time [19]. Generally, ReLU can be classified as a piecewise linear function. This is different from the other two functions as those are continuously differential. One of the critical features for the ReLU function is that it mitigates one of the main issues, which is dealing with the negative weighted numbers [20]. The function that denotes ReLU is:

$$f(x) = \theta(x) = max(x, 0)$$

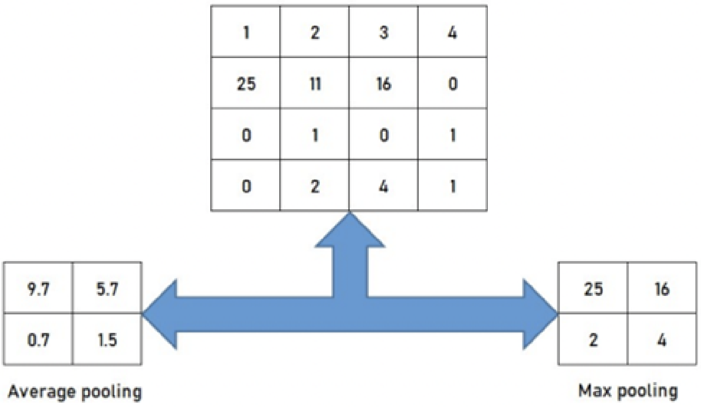
Function	Equation	Range	Derivative Equation
ReLu	$f(x) = \begin{cases} 0;x<0 \\ x;x\geq0 \end{cases}$	0, $+\infty$	$f'(x) = \begin{cases} 0;x<0 \\ 1;x\geq0 \end{cases}$

Table 3.1: ReLu Function

From the figure and equation above, it is evident that the output will be zero(0), if the input is any negative number (x < 0). Nevertheless, for the positive inputs, the output will produce a linear output proportional to the input. Which is given as “x” in this equation. Effectively making the slope/gradient = 1. For half of the input domain, this function is linear; for the other half, it is non-linear. This results in a far more efficient computation. The negative values are neglected in the process, which makes the process a bit easier, and at the same time, it stays a non-linear function for the positive values. If the system has enough positive values, then this function is the most preferable. Furthermore, ReLU is the only function that can output a true zero (0) value out of all three functions. The hidden layer benefits from this function because of its simplistic approach.

Pooling Layers

CNNs use two different types of layers: convolutional (which resembles primary cells) and pooling (which models complicated cell behavior). Each convolutional layer applies a non-linear transfer function to the source picture and executes a discrete 2D convolution operation with an altered kernel. The pooling layers reduce the amount of the input by aggregating neurons from a narrow spatial region. The activity done by pooling layers can be simply replaced without affecting the architecture, which is one of the key advantages for employing CNN [21]. Due to its ability to reduce the dimensionality of feature maps, pooling is an essential step in convolutional systems. A group of values is combined into a smaller number of values, decreasing the dimensionality of the feature. By retaining relevant information, It turns the joint feature representation into useful information. Pooling operators enable spatial transformation invariance while minimizing the computational cost for top layers by eliminating specific connections between convolutional layers. The pooling layer is mostly used for two advantages. The first is to keep the set of variables or weights as low as possible to reduce processing costs, and the second is to avoid over-fitting. A pooling method should only extract valuable information, while extraneous details should be eliminated. Though there are also other forms of pooling operation, it is mainly divided into 1) Max Pooling and 2) Average Pooling. There also exists another form of Pooling which is known as subsampling.



1	2	3	4
25	11	16	0
0	1	0	1
0	2	4	1

9.7	5.7
0.7	1.5

25	16
2	4

Figure 3.2: Finding Avg and Max Pooling From given image pixel

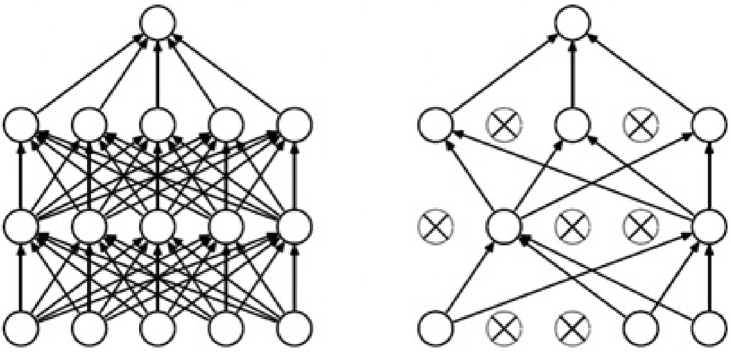


Figure 3.3: Dropout Neural Net Model

Average Pooling: Down-sampling is performed via an intermediate pooling layer, with which rectangular pooling regions are divided and the average values of each zone are computed.

$$a_j = \tanh(\beta \sum_{N*N} a_i^{n*n} + b)$$

Max Pooling: The maximum value inside a group of R activations is passed forward by the max-pooling operator.

$$a_j = \max_{N*N}(a_i^{n*n} u(n, n))$$

In our model, each pooling layer used max-pooling with a 2x2 filter. The max-pooling layer will conduct a MAX operation on the depth slice of the input, taking the maximum over the 2x2 region.

Dropout Layer

In machine learning, the term “Dropout” [22] refers to the process of randomly omitting some nodes from a layer during training. Deep neural networks with many parameters are robust machine learning systems. The difficulty is that these networks are overcrowded. Large networks are highly sluggish to utilize, making it challenging to combine numerous forecasts—massive neural networks for testing. Dropout is a solution to this issue.

A standard neural network with two hidden layers is shown on the left side of the diagram, while a reduced network constructed by removing the network on the left is shown on the right. Crossed units have been removed from the network. On the upper-left, a typical neural network with all units active is displayed. Fewer weights and biases are considered during training for the crossed teams led on the right side. Dropout prevents overfitting and enables the efficient combination of an exponentially large number of distinct neural network designs. When a unit is dropped, all of its incoming and outgoing links are ended, and it is no longer linked to the network. Randomly, the order in which neurons are dropped is determined. In the simplest example, each unit is given a fixed probability ’P’ that is independent of other units, where ’P’ can be selected from a validation set or set to 0:5, which appears to be near to optimal for a variety of networks and tasks. On the other hand, the best probability for the input units is typically closer to 1 than to 0.5. In our model, dropout was only employed after the first dense layer in this model to address overfitting concerns, at a ratio of 0.5.

Flattening

Another crucial layer of the Convolutional Neural Network is this one (CNN). It is an extremely important layer for data feeding. The last layer of some artificial neural networks is dense, which anticipates data in a one-dimensional system. The final level of a CNN model is a classifier, which is a dense layer. Before the ANN can use the pooling layer's output, it must be transformed into a one-dimensional feature vector. Flattening is the term for this procedure. This is all that is required to flatten the output of the convolutional or pooling layer and create a single lengthy feature vector that the dense layer can utilize to make the final classification. The long vector that we obtain after flattening will be the input layer for an artificial neural network.

Hidden Layer

A completely linked unit of the hidden layers was set at 500 to aid in the discovery of hidden elements within the photos.

Optimizer, Loss Function and Output Layer

The output layer is the final layer of a CNN. This layer generates an estimate of each class based on the input image provided. There is a neuron in the output layer for each conceivable class, i.e. one neuron for unmodified pictures and another for each potential modification. This layer makes use of the softmax activation function, which maps the last dense layer and provides vector output that is summed together in a single output. It will indicate whether or not each element belongs to a specific class. The softmax function normalizes a vector of K absolute values into a probability distribution of K probabilities.Each element will have a value between 0 and 1, and they will add up to 1, making them a probability value. Additionally, a greater number of input components indicates a higher chance. With the softmax layer, we may convert a network's non-normalized output to a probability distribution across anticipated output classes. The softmax function is:

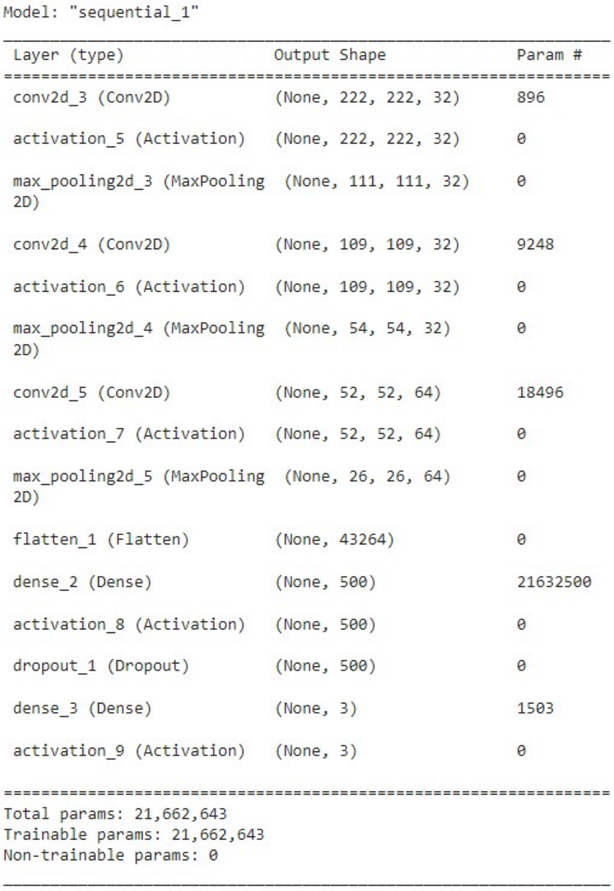
$$P(c_r|x,\theta) = g(a(x,\theta))_r \frac{e_r^a(x,\theta)}{\sum_{j=1}^{k} e_j^a(x,\theta)} = \frac{P(x,\theta|c_r)P(c_r)}{\sum_{j=1} KP(x,\theta|c_j)P(c_j)}$$

The softmax layer computes the standard exponential function for each component zj in the input vector and normalizes the outcome by dividing it by the sum of all these exponentials. This normalization guarantees that the output vector z component sums to one. The output layer employs a three-way softmax activation function to generate a probability distribution over the three classes. As an optimizer, the model employed RMSProp (Root Mean Squared Propagation), which is a superior form of Gradient Descent and helps to focus on recently observed partial derivatives[13]. Categorical cross entropy was employed as the loss function, as it is in multi class classification models.

Including the dropout layer and hidden layer which we already mentioned above we have used other hyperparameter as well in the training, which are batch size of 32 as it is good for low learning rate[14] and the learning rate itself is 0.01 because we believe its a good starting point for our problem in the training.

Another hyperparameter we used in our paper is epoch. Here we used 10 Numbers of epochs to prevent overfitting and maximize our models generalization performence. Model validation, which entails evaluating the model's performance after each training session, is done with a portion of the training data. On both the training and testing sets, loss and accuracy are collected to identify the epoch number at which the model begins to overfit.

3.1.3 Architecture of feature Extractor:



```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_3 (Conv2D)           (None, 222, 222, 32)      896

 activation_5 (Activation)   (None, 222, 222, 32)      0

 max_pooling2d_3 (MaxPooling  (None, 111, 111, 32)     0
 2D)

 conv2d_4 (Conv2D)           (None, 109, 109, 32)      9248

 activation_6 (Activation)   (None, 109, 109, 32)      0

 max_pooling2d_4 (MaxPooling  (None, 54, 54, 32)       0
 2D)

 conv2d_5 (Conv2D)           (None, 52, 52, 64)        18496

 activation_7 (Activation)   (None, 52, 52, 64)        0

 max_pooling2d_5 (MaxPooling  (None, 26, 26, 64)       0
 2D)

 flatten_1 (Flatten)         (None, 43264)             0

 dense_2 (Dense)             (None, 500)               21632500

 activation_8 (Activation)   (None, 500)               0

 dropout_1 (Dropout)         (None, 500)               0

 dense_3 (Dense)             (None, 3)                 1503

 activation_9 (Activation)   (None, 3)                 0

=================================================================
Total params: 21,662,643
Trainable params: 21,662,643
Non-trainable params: 0
_________________________________________________________________
```

Figure 3.4: Architecture of feature Extractor

Our model was implemented on a Tensorflow sequential model to extract features. In general, we know CNN has three different layers: the Conv Layer, the pooling layer, and the fully connected layer. So we had our three Conv2D layer. While each Conv layer was stacked with a Max-pooling layer and a activision layer. Next a flatten layer is used to turn the data into 1D to feed it to the next layer. Then a dropout layer is used to prevent overfitting issues. At the end, a fully connected layer was added. We also used the softmax Activision function at the output to generate a probability distribution of our result. This is the overall flow in the model:

ConV2D → ActivisionFunction → MaxPooling → (Previous three layers continues for two cycles) → Flatten → FC

This is our overall architecture of our CNN model that used here.

3.2 Pre-trained Model

We used three pre-trained models in our thesis to compare the accuracy to our built model. This three models are: MobileNetV2, InceptionV3 and Xception. In this part we will talk about those pre-trained model.

3.2.1 InceptionV3

One of the most popular models among researchers is the Inceptionv3 model. We can now go back and retrain the last layers of existing products, saving a significant amount of time. InceptionV3 was trained on over a million pictures from the ImageNet database, suggesting that the model had learned from its original training and could be used to a smaller dataset with excellent classification accuracy without having to retrain the entire model. The Inception Layers are a series of layers (1*1 convolutional layer, 3*3 convolutional layers and 5*5 convolutional layers) that merge the result filters into a single output vector[16], which generates the parameters for the next step. Modifying an Inception network for multiple use cases becomes a difficulty because the new network's performance is so unclear. Inception v3 has already introduced a variety of approaches to strengthen the network and remove constraints, allowing for faster model adoption. Only a few of the techniques used in parametric convolution include batch normalizing, down sampling, and parallel processing. To prevent losing any operational advantages, changes to an Inception Network must be done with caution.

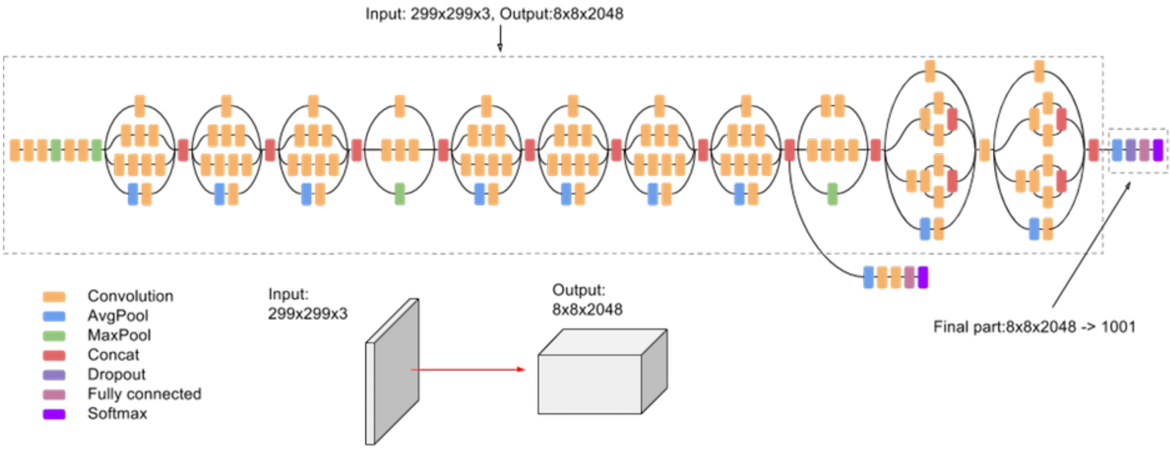


Figure 3.5: Architecture of InveptionV3

3.2.2 Xception

Xception is after inception, takes the principle of incpetion. It stands for extreme inception. To compress the original input in Inception, 1x1 convolutions were employed, and different types of filters were used to each of the depth spaces dependent on the input spaces. This is exactly what Xception does. Instead, the filters are applied to each depth map separately before the input space is compressed using 1*1 convolution across the depth. This method is similar to depthwise separable convolution, a neural network development method that has been used since 2014. Another distinction that can be found in Inception and Xception is after the original operation, the existence or absence of a non-linearity. Where in inception that was followed by ReLu non-linearity activision function while in Xception, it does not add any non-linearity.

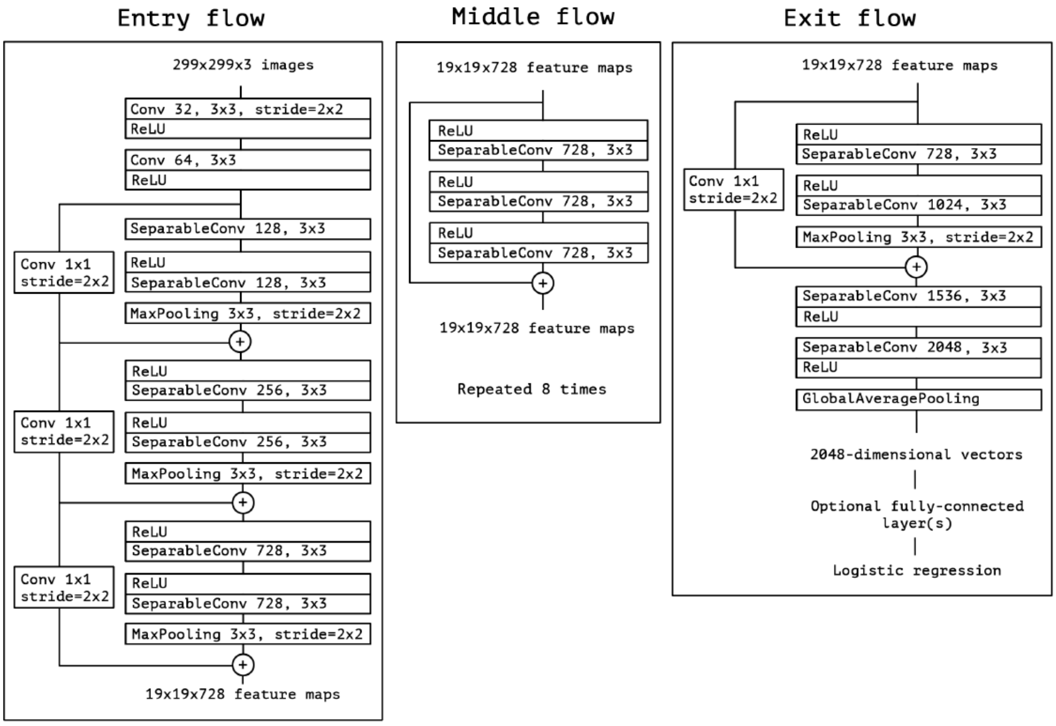


Figure 3.6: Architecture of XceptionV3

3.2.3 MobileNetV2

MobileNets are a depthwise separable convolution design that reduces the number of connections in order to reduce the size of the model and the complexity of the system. The technique is useful for embedded and mobile applications. In this type of system, the author has incorporated two global hyperparameters, which are as follows: This method achieves a reasonable balance between model latency and accuracy. Furthermore, the hyperparameters allow for the selection of a suitable scaled model in line with the problem constraints if needed.

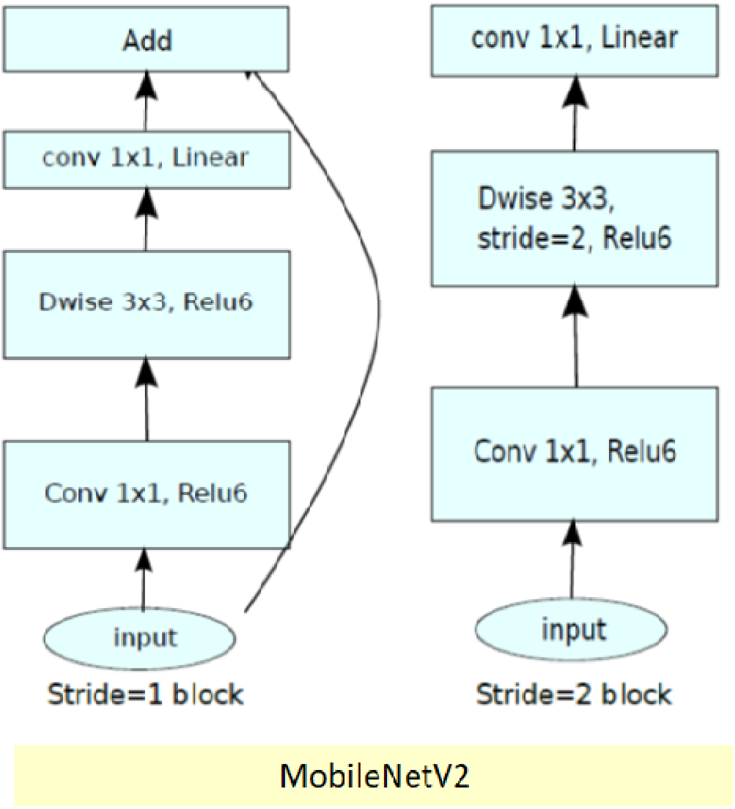


Figure 3.7: Architecture of MobileNetV2

Chapter 4

Input Data

4.1 Dataset Collection

In order to construct the dataset for detecting multiple eye diseases, we collected images from a local eye hospital of approximately 6000 different images. For our algorithm to be trained, we had to ensure that we had the images for all of the retinal diseases. Our datasets included both Training and Validation data sets, which were both carefully checked. For experimentation, we have splitted the dataset into 8:2 ratio of training and testing sets. In our dataset, we used 4 classes:

1. Normal
2. Glaucoma
3. Cataract
4. Diabetic Retinopathy

4.2 Dataset Sample

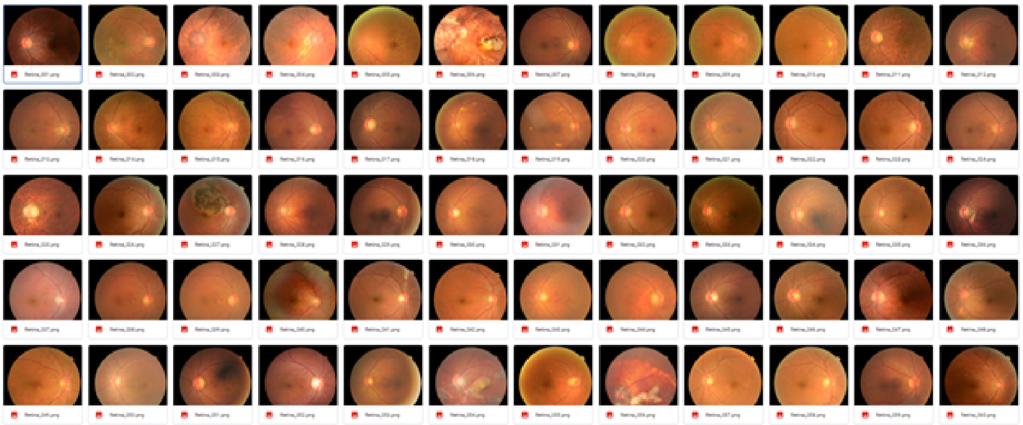


Figure 4.1: Sample Images from the Dataset

Our datasets were created by combining four different illnesses, and each class included 1500 photos. Thus, it is quite difficult to display an appropriate sample from the database. A total of 60 images were able to be combined into Figure 2 from our dataset, we attempted to display at least 50 to 60 images. It is evident from this sample image that we collected photographs from multiple disease classifications to generate our dataset. Approximately three to four photos are presented in each disease category.

4.3 Data Labels

We can divide our dataset into four separate labels. These photos show human eyes with normal retinas and three retinal diseases. As a result, a binary classifier may be employed to describe our dataset. The ”Class” attribute is used to distinguish between four binary datasets that are all identical.

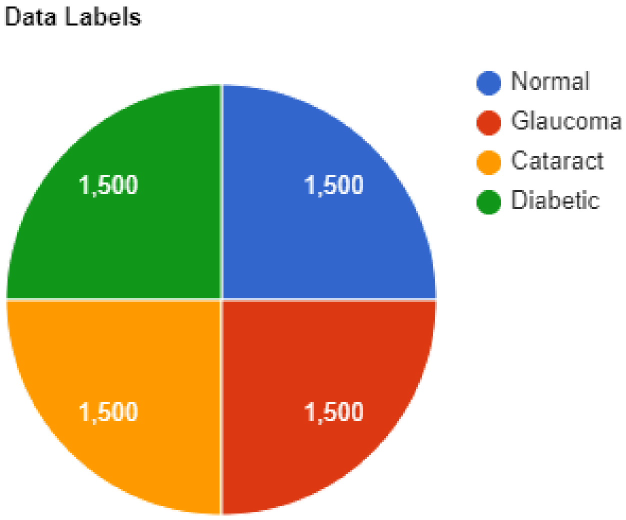


Figure 4.2: Pie-chart illustration of label balancing

4.4 Data Pre-processing

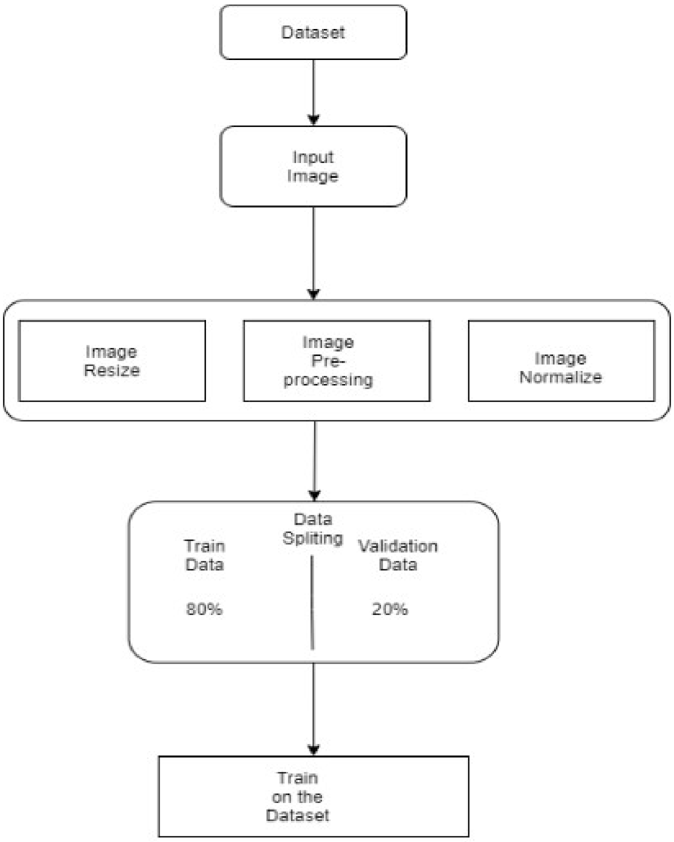


Figure 4.3: Data Preprocessing

In order to extract features for the dataset we used to recognize a variety of objects from a single image, we needed to load some necessary libraries, resize images, and initialize the directory. Before data preprocessing can begin, we imported OS, TensorFlow, NumPy as np, glob, matplotlib.pyplot as plt, and other libraries. Once the libraries had been imported, we set up the dataset's directory so that the data can be obtained by the model, and details can be extracted before images can be resized. A folder called Multiple eye disease was made from our entire dataset and uploaded to Google Drive. The folder consisted of two subfolders: Train and Validation. To round things out, our images have been resized as follows: image height = 222, image width = 222, class count = 4. After resizing the photos, we used our own model to extract features.
We also used the Prewitt operator to preprocess our dataset.

4.4.1 Image Resizing

To get optimal performance, each type of CNN architecture necessitates a different image size to resize or distort our image from the one-pixel grid to another. Image resizing is applied based on the model architecture. The input shape is set as (222,222) in our proposed RetinalNet-500 model.

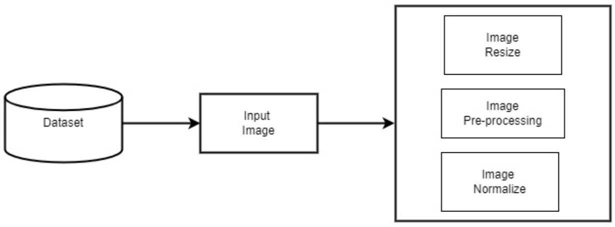


Figure 4.4: Image resizing

4.4.2 Splitting Train and Validation Set

The data set is divided into two parts, a training set and a validation set. In order to determine the accuracy of our CNN models, we will train them using batch size and epochs on the training set, and then evaluate them on the validation set. A comparison of models will take into account precision, recall, f1-score, and accuracy. Our multiple eye diseases dataset included 6,000 fundus images that were split into two subsets: 80% for training and 20% for validation. We tested the accuracy of the model on a subset of these images. Before fitting the model, we shuffled the images so that the model does not memorize anything if the images belonging to same category are fed in a consequent manner.

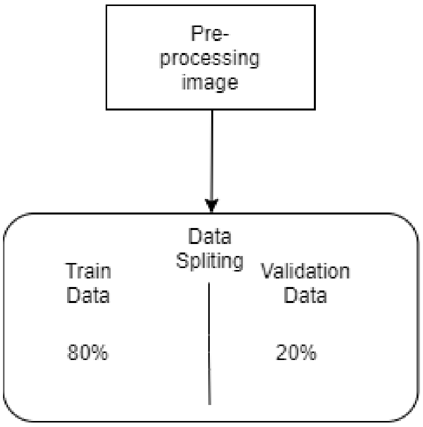


Figure 4.5: Splitting Train and Validation Set

Prewitt Operator

In order to find edges in an image, the Prewitt operator is utilized. Horizontal and vertical edges are the two types of edges it detects. Edges are calculated using the difference between equivalent image pixel brightness. Derivative masks refer to all of the masks that are utilized for edge detection. Because an image is also a signal, the only way to calculate changes in a signal is to use differentiation. As a result, derivative masks or derivative operators are occasionally used to refer to these operators.

The following properties must be included in all derivative masks: the mask's opposite sign must be utilized, the mask's sum must be zero, and higher weight equals better edge detection. We get two masks from the Prewitt operator: one for detecting horizontal edges and the other for identifying vertical edges.

Vertical

-1	0	1
-1	0	1
-1	0	1

Horizontal

-1	-1	-1
0	0	0
1	1	1

This mask will highlight the horizontal and vertical boundary lines of a picture. In the same way as the above mask, it computes the distance between the pixel brightness of a specific edge. Because the middle row of the mask is all zeros, it neglects the image's initial edge values and calculates the difference among the above and below image intensity of the given edge instead. As a consequence, the abrupt change in intensities is accentuated, highlighting the edge. The derivative mask concept is used in both of the above masks. The opposing signs in both masks are the same, and their aggregate is zero.

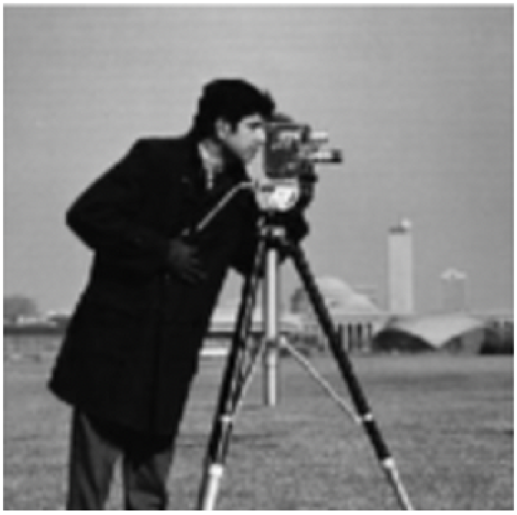


Figure 4.6: Before

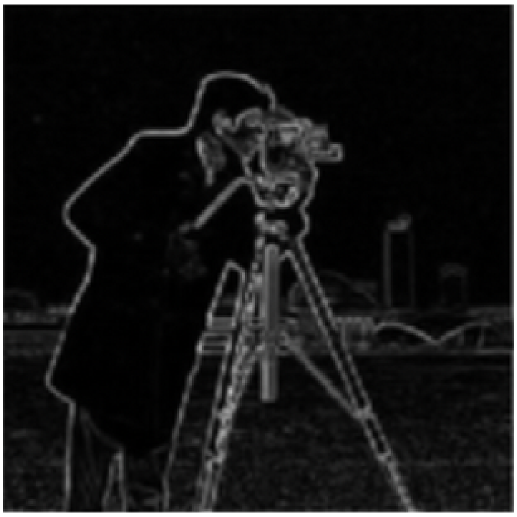


Figure 4.7: After

Chapter 5

Experiment And Result

We did create our own model, which is depicted in the architecture above. We trained several of the pre-trained models with our gathered dataset to assess the dependability of our model, and the results are displayed below.

5.1 Individual results for each model

The accuracy of each pre-trained model i.e InceptionV3, MobileNetV2 and Xception, as well as our freshly trained one, RetinalNet-500, is listed in the table below:

Model	Epochs	Accuracy
InceptionV3	10	97.30%
MobileNetV2	10	97.30%
Xception	10	96.45%
RetinalNet-500	10	95.15%

Table 5.1: This shows the accuracy result of each model

Table 5.1 summarizes the results of the pre-trained models along with the result of our freshly trained RetinalNet-500. For 10 epochs, the accuracy for each of these models are listed here.

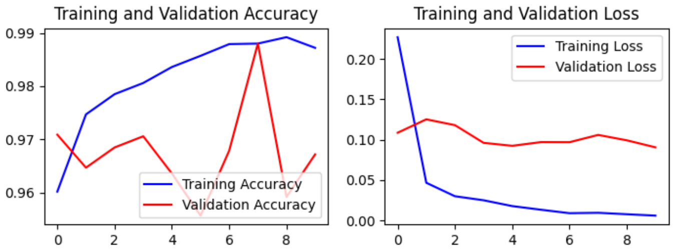


Figure 5.1: Accuracy in Inception-V3 architecture

Figure 5.1 demonstrates the training and validation accuracy and loss of InceptionV3 architecture where training curves are depicted using the blue line and validation curves are depicted using the red line. In case of training accuracy, starting from a value of 0.96, the accuracy reaches up to 0.99 then started drooping in the last epoch. Again, in case of validation accuracy, starting from a value of 0.971, the accuracy reaches up to 0.987 then drops to 0.97 in the last epoch.

On the other hand, in case of training loss, starting from a value of 0.25, the loss decreases almost to 0.00 in the last epoch. In case of validation loss, starting from a value of 0.11, the loss decreases to 0.1 in the last epoch.

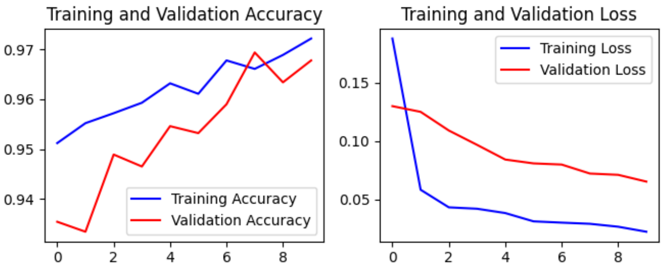


Figure 5.2: Accuracy in MobileNetV2 architecture

Figure 5.2 demonstrates the training and validation accuracy and loss of MobileNetV2 architecture where training curves are depicted using the blue line and validation curves are depicted using the red line. In case of training accuracy, starting from a value of 0.95, the accuracy reaches up to 0.99 in the last epoch. Again, in case of validation accuracy, starting from a value of 0.93, the accuracy reaches up to 0.965 in the last epoch.

On the other hand, in case of training loss, starting from a value of 0.20, the loss decreases to 0.00 in the last epoch. In case of validation loss, starting from a value of 0.155, the loss decreases to 0.055 in the last epoch.

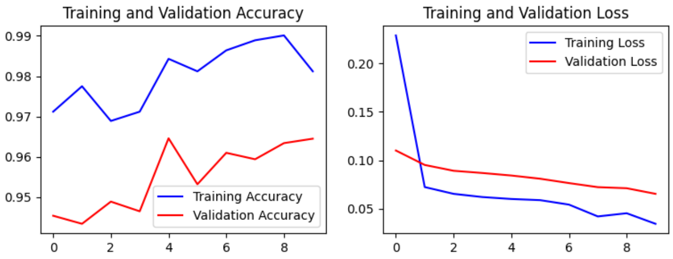


Figure 5.3: Accuracy in Xception architecture

Again, Figure 5.3 demonstrates the training and validation accuracy and loss of Xception architecture where training curves are depicted using the blue line and validation curves are depicted using the red line. In case of training accuracy, starting from a value of 0.97, the accuracy reaches up to 0.98 in the last epoch. Again, in case of validation accuracy, starting from a value of 0.955, the accuracy reaches up to 0.962 in the last epoch.

On the other hand, in case of training loss, starting from a value of 0.25, the loss decreases to 0.00 in the last epoch. In case of validation loss, starting from a value of 0.12, the loss decreases to 0.055 in the last epoch.

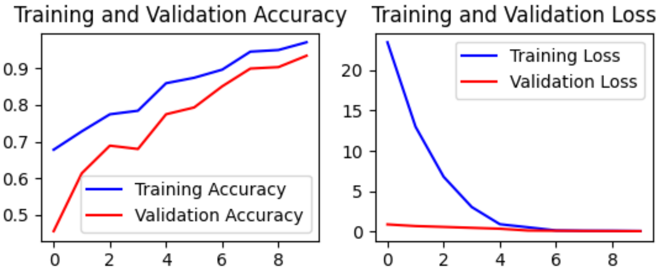


Figure 5.4: Accuracy of the RetinalNet-500

Figure 5.4 demonstrates the training and validation accuracy and loss of RetinalNet-500 architecture where training curves are depicted using the blue line and validation curves are depicted using the red line. In case of training accuracy, starting from a value of 0.69, the accuracy reaches up to 0.99 in the last epoch. Again, in case of validation accuracy, starting from a value of 0.00, the accuracy reaches up to 0.95 in the last epoch.

On the other hand, in case of training loss, starting from a value of 0.25, the loss decreases to 0 in the last epoch. In case of validation loss, starting from a value of 0.2, the loss decreases to 0 in the last epoch.

5.2 Experimentation and Analysis

After creating our desired model, we also did compare the results of our model with popularly known pre-trained models such as MobileNet V2, Inception V3 and Xception. We compared the models through different numerical results such as Validation Accuracy, Recall, Precision and F1 Score. After the experimentation we achieved the values which are shown in the table below. Our model did perform very close to the mentioned pre-trained models instead of it being a less layered CNN structure.

Architecture	Precision	Recall	F1 Score	Accuracy
InceptionV3	94.83	97.78	96.28	97.30
MobileNetV2	93.22	98.11	95.60	97.30
Xception	92.16	96.71	94.38	96.45
RetinalNet-500	91.26	96.79	93.94	95.15

Table 5.2: Comparsion between CNN architectures

Chapter 6

Conclusion and Future Works

6.1 Conclusion

In recent years, this field was heavily researched by researchers with different algorithms. Although the techniques and algorithms used for this gave good results, there are still techniques left that might give even better results. In this paper, we used a model that we had recently constructed. We demonstrated how deep CNN can successfully segregate exudates in color fundus images.

6.2 Challenges

6.2.1 Computational Power

The computational power to process this data set with 6000 images was limited due to the lack of power and proximity to the campus computer lab. It was not possible to run specific algorithms such as VGG19, ResNet101, and other models due to inadequate GPU processing power. After the training phase of the VGG16 model was complete, the model crashed. It wasn't able to make predictions. Finally, we used Google's computational server, ”Google Colab”, and finally got the expected results.

6.2.2 Excessive Training Time

Computational power issues are a consequential effect of this. As the computing power we were equipped with was inadequate, each model to train and test took a very long time to run. In some cases, each epoch took more than an hour to run. As a result of the slow training time, the research progress of the team was hampered. For future comparisons and analysis, we plan to run other CNN models and resolve these two issues.

6.2.3 Future Works

To improve the accuracy of our CNN model and shorten the training time, we have implemented convolutional layers, activation functions, max pooling, dense, and various other methods. It is our intention to add more layers with different parameters tweaked in the future to increase the accuracy of the model. Using a more challenging dataset, and then modifying it accordingly, will allow us to test its accuracy in a future implementation of the model. Eventually, we plan to use the ensemble learning approach to solve similar complex problems. This will aid others in solving categorical classification problems in medical imaging datasets. The goal of this research is to develop an application that can be used by anyone in the near future to find out if fundus images indicate retinal diseases or not, allowing them to diagnose such diseases.

Chapter 7

Bibliography

1. Functions, S. D. A. R. (2019, October 8). World report on vision. WHO. https://www.who.int/publications/i/item/9789241516570

2. Prasad, K., Sajith, P. S., Neema, M., Madhu, L., & Priya, P. N. (2019). Multiple eye disease detection using Deep Neural Network. TENCON 2019 - 2019 IEEE Region 10 Conference (TENCON). Published. https://doi.org/10.1109/tencon.2019.8929666

3. Linglin Zhang, Jianqiang Li, i Zhang, He Han, Bo Liu, Yang, J., & Qing Wang. (2017). Automatic cataract detection and grading using Deep Convolutional Neural Network. 2017 IEEE 14th International Conference on Networking, Sensing and Control (ICNSC). Published. https://doi.org/10.1109/icnsc.2017.8000068

4. Xu, X., Zhang, L., Li, J., Guan, Y., & Zhang, L. (2020). A Hybrid Global-Local Representation CNN Model for Automatic Cataract Grading. IEEE Journal of Biomedical and Health Informatics, 24(2), 556–567. https://doi.org/10.1109/jbhi.2019.2914690

5. Hossain, M. R., Afroze, S., Siddique, N., & Hoque, M. M. (2020). Automatic Detection of Eye Cataract using Deep Convolution Neural Networks (DCNNs). 2020 IEEE Region 10 Symposium (TENSYMP). Published. https://doi.org/10.1109/tensymp50017.2020.9231045

6. Doshi, D., Shenoy, A., Sidhpura, D., & Gharpure, P. (2016). Diabetic retinopathy detection using deep convolutional neural networks. 2016 International Conference on Computing, Analytics and Security Trends (CAST). Published. https://doi.org/10.1109/cast.2016.7914977

7. Shankar, B. M., Nagaraj, V., Sivakumar, S. A., Vidhya, B., Ganesh, R. S., & Premalatha, P. (2021). Glaucoma Detection with Fully Convolutional Neural Network using Optic Disc and Segmentation Methods. 2021 7th International Conference on Advanced Computing and Communication Systems (ICACCS). Published. https://doi.org/10.1109/icaccs51430.2021.9441545

8. Serener, A., & Serte, S. (2019). Transfer Learning for Early and Advanced Glaucoma Detection with Convolutional Neural Networks. 2019 Medical Tech-

nologies Congress (TIPTEKNO). Published.
https://doi.org/10.1109/tiptekno.2019.8894965

9. Sridhar, S., & Sanagavarapu, S. (2020). Detection and Prognosis Evaluation of Diabetic Retinopathy using Ensemble Deep Convolutional Neural Networks. 2020 International Electronics Symposium (IES). Published.
https://doi.org/10.1109/ies50839.2020.9231789

10. Huang, J., Lu, H., Lopez Meyer, P., Cordourier, H., & del Hoyo Ontiveros, J. (2019). Acoustic Scene Classification Using Deep Learning-based Ensemble Averaging. Proceedings of the Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2019).
https://doi.org/10.33682/8rd2-g787

11. Kumar, A., Kim, J., Lyndon, D., Fulham, M., & Feng, D. (2017). An Ensemble of Fine-Tuned Convolutional Neural Networks for Medical Image Classification. IEEE Journal of Biomedical and Health Informatics, 21(1), 31–40.
https://doi.org/10.1109/jbhi.2016.2635663

12. Smaida, M., & Yaroshchak, S. (2020). Using Ensemble Learning for Diagnostics of Eye Diseases. International Journal of Scientific and Research Publications (IJSRP), 10(10), 273–279.
https://doi.org/10.29322/ijsrp.10.10.2020.p10639

13. Zaheer, R., Shaziya, H. (2019). A Study of the Optimization Algorithms in Deep Learning. 2019 Third International Conference on Inventive Systems and Control (ICISC).
https://doi.org/10.1109/icisc44355.2019.9036442

14. Kandel, I., & Castelli, M. (2020). The effect of batch size on the generalizability of the convolutional neural networks on a histopathology dataset. ICT Express, 6(4), 312–315.
https://doi.org/10.1016/j.icte.2020.04.010

15. Jain, L., Murthy, H. V. S., Patel, C., & Bansal, D. (2018). Retinal Eye Disease Detection Using Deep Learning. 2018 Fourteenth International Conference on Information Processing (ICINPRO). Published.
https://doi.org/10.1109/icinpro43533.2018.9096838

16. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2015, December 11). Rethinking the inception architecture for computer vision. Retrieved May 27, 2022, from https://arxiv.org/abs/1512.00567v3

17. I. Goodfellow, Y. Bengio, and A. Courville, Deep learning. MIT press, 2016.

18. A. B. Amir, U. H. Nisa, A. A. Shafi, M. Reza, et al., “Traffic sign recognition using deep learning,” Ph.D. dissertation, Brac University, 2019.

19. Y. Liu, J. Zhang, C. Gao, J. Qu, and L. Ji, “Natural-logarithm-rectified activation function in convolutional neural networks,” in 2019 IEEE 5th International Conference on Computer and Communications (ICCC), IEEE, 2019, pp. 2000–2008.

20. V. Nair and G. E. Hinton, “Rectified linear units improve restricted boltzmann machines,” in Icml, 2010

21. D. Scherer, A. M¨uller, and S. Behnke, “Evaluation of pooling operations in convolutional architectures for object recognition,” in International conference on artificial neural networks, Springer, 2010, pp. 92–101.

22. N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: A simple way to prevent neural networks from overfitting,” The journal of machine learning research, vol. 15, no. 1, pp. 1929–1958, 2014