Application of CNN in Hypertensive Retinopathy Classification

By

Degel Sharon Agoreyo

In Partial Fulfillment of the Requirements for the Degree of Master of Science in Biomedical Engineering

Center of Biomedical Engineering

Addis Ababa Institute of Technology

Addis Ababa University

Advisor: Dawit Assefa Haile (PhD)

Addis Ababa, Ethiopia

January, 2022

Declaration

I, the undersigned, declare that this MSc thesis is my original work, has not been presented for fulfillment of a degree in this or any other University, and all sources and materials used for the thesis have been acknowledged.

Name: Degel Sharon Agoreyo

Signature: _____

Date: _____

This MSc thesis has been submitted for examination with my approval as an advisor.

_____

Dawit Assefa Haile (PhD)

Certificate of Examination

Addis Ababa University

School of Graduate Studies

This is to certify that the thesis prepared by Degel Sharon Agoreyo entitled: "*Application of CNN in Hypertensive Retinopathy Classification"* submitted in partial fulfillment of the requirements for the Degree of Master of Science in Biomedical Engineering (Bioinstrumentation and Imaging) complies with the regulations of the University and meets the accepted standards with respect to originality and quality.

Signed by the examining committee

Examiner _____Signature _____Date _____

Examiner _____Signature _____Date _____

Advisor _____ Signature _____Date _____

_____

Chief of Department or Graduate Program Coordinator

ABSTRACT

Hypertension (HTN) as defined by the world health organization (WHO) is when the blood pressure is elevated with a systolic blood pressure equal to or greater than 140mmHg and a diastolic blood pressure equal to or greater than 90mmHg when measured on 2 separate days. HTN, according to Center for Disease Control and Prevention (CDC), raises the risk for stroke and other heart diseases. In a previous study, it was estimated that there were 972 million adults with HTN in 2000; 333 million in developed countries and 639 million people in developing countries. In a review of current trends by WHO, there has been an increase in adults with HTN from 594 million in 1975 to 1.13 billion in year 2015 with a prevalence of 27% in Africa and 18% in the America. Hypertensive Retinopathy (HPR) is damage to the retinal blood vessels caused by HTN. The use of artificial intelligence has recently been applied in Medical Imaging. One of such areas is in classification of eye diseases. In HPR, previous studies only focused on classification according to two classes; normal or HPR. In the current study, a classification method based on four classes according to the Wong and Mitchell classification which gives a classification based on the severity of HTN severity is presented; Normal, Mild, Moderate, and Severe or Malignant HPR. The study explored the idea of applying Convolutional Neural Network (CNN) in classifying HPR. The work focused on images collected solely from patients with HTN and will play an important role in assisting physicians to speed up the process of diagnosis. The study used the HPR features of the retina for classification. Transfer learning using existing pretrained models was employed in classifying the dataset. Five state-of-the-art pretrained models were employed for classification into two classes; normal and abnormal and into four classes: normal and three stages of HTR. The five different neural architectures namely: ResNet-101, GoogleNet, AlexNet, VGG-19 and Xception achieved an accuracy of 91.61%, 93.01%, 87.41%, 90.21% & 85.31% for the 4 Classification task and 100%, 100%,100%, 99.30% & 97.89% for the 2 Classification task respectively. The best accuracy was obtained using GoogleNet. The pretrained models show that with proper tuning of training parameters like the learning rate, number of epochs, batch size, type of optimizer, activation and loss function, they can be used in hypertensive retinopathy classification.

Keywords: Hypertension, Hypertensive Retinopathy Classification, Convolutional Neural Networks, Pretrained models, Transfer learning.

ACKNOWLEDGMENTS

I would like to express my gratitude to the God Almighty who has been always been with me. He has given me the platform, strength and wisdom to finish strong. I would like to thank my biological parents Mr and Mrs Sunny EL-Stella Agoreyo who have stood by my side all the way. I would like to thank my biological siblings Ese, Mareh and Pleion and my extended sibling Temi, Oreofe and UK for cheering me on. Special thanks to the Omorogbe's, my Uncle Kingsley and Aunty Vivian and my Uncle Nosa for their encouraging words and support. I would also like to thank my surrogate parents, The Owvahs, The Edemas, The Akwukwumas, The MfonAbasis, The Gagars, The Osarhenkhoes, The Olomukoros, The Orighorherhas and The Olunubis for their countless prayers and support. Specials thanks to Jesam Ujong for the help rendered during my thesis.

I would like to thank my friends at the Addis Ababa University, Wafa, Stephen, Lucky, Annitah, Beauty, Pius and Dr Hart who have been my support system here in Addis and to other friends at the Addis Ababa University I say a big thank you. Thank you for the encouragements. I would like to thank also the Ophthalmology Department Staff at the Saint Paul Hospital, Addis Ababa. Special thanks to Dr Adugna, Mr Dereje, Mrs Tesgy, Mr Zewde and Miss Kalkhidan for their unwavering help during my data collection and classification. I would like to thank my Supervisor Dr. Dawit Assefa Haile who has made my education in Addis to be very interesting and who gave me guidance during my thesis. I would like to thank my sponsors ABEM and the European Union for the opportunity and financial support.

TABLE OF CONTENT

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATION

HTN	Hypertension
HPR	Hypertensive Retinopathy
ANN	Artificial Neural Network
CNN	Convolutional Neural Network
DL	Deep Learning
ML	Machine Learning
AI	Artificial Intelligence
AFIO	Armed Forces Institute of Ophthalmology
FOV	Field of View
ReLU	Rectified Linear Units
GPU	Graphic Processing Unit
DSC	Depthwise Separable Convolutions
TP	True Positives
TN	True Negatives
FP	False Positives
FN	False Negative
SGDM	Stochastic Gradient Descent
ADAM	Adaptive Moment Estimation

CHAPTER 1
INTRODUCTION

1.1 Background

Hypertension (HTN), as defined by the world health organization (WHO), is when the blood pressure of a person is elevated. HTN according to CDC [1] raises the risk for stroke and heart diseases which are the two leading causes of death in the US, for example. It is diagnosed when the diastolic blood pressure is above or equal to 90 mmHg and the systolic pressure is greater than or equal to 140 mmHg when measured on 2 separate days [2]. In a past study [3], it was estimated that there were 972 million adults with HTN in 2000; 333 million in developed countries and 639 million people in developing countries. They predicted that by 2025, the number would increase by 60% to about 1.56 billion. In a review of current trends by WHO [2], there has been an increase in adults with HTN from 594 million in 1975 to 1.13 billion in year 2015 with a prevalence of 27% in African regions and 18% in the Americas.

Hypertensive Retinopathy (HPR) is damage to the retinal blood vessels caused by HTN. It occurs when there are changes in the blood vessels of the eyes as a result of high blood pressure. Elevated blood pressure causes the walls of the blood vessels in the eye to increase and this reduces the amount of space in the blood vessels which makes it hard for blood to access the retina. There are 4 stages of HPR based on the Wong and Mitchell Classification: No retinopathy, Mild retinopathy, Moderate, and Severe or Malignant Retinopathy. The Wong and Mitchell (WM) classification simplifies the Keith-Wagener-Barker (KWB) method by combining the first and second stage of the KWB classification into one stage. Also, retinopathy characterization into four grades in retinal studies does not appear to be supported with the use of fluorescein angiography [4]. The Wong and Mitchell classification gives a classification based on the severity of hypertensive retinopathy. According to several articles, stroke risk is higher in people with HPR and imaging of the retina can be used in detection of persons at risk, this is still being researched to determine the exact risk stratification with the different stages of HTR. In one of the previous studies [5], it was discovered that assessment of the signs of HPR can be used to predict the long-term risks of stroke even in patients with well controlled HTN.

1.2 Problem Statement

The use of Convolutional Neural Networks (CNN) has been applied in different medical imaging applications. One of such areas is in classification of eye diseases. This has been met with good success and showed high potentials. A lot of research work has been done extensively on the classification of eye diseases like diabetic retinopathy, age-related eye diseases, Cataract and some other eye diseases. In HPR, previous studies only focused on classification according to two classes; normal or HPR and not based on the HPR stages apart from a recent study by [6], where they employed a semantic instance segmentation in DenseNet architecture for the classification Hypertensive Retinopathy into 5 stages according to the KWB Classification. They made use of pretrained HPR related lesions in their HYPER-RETINO system and used images from the PRV-HR, DRIVE, DR-HAGIS, DiaRETDB&1, DR1&DR2, Kaggle-DR, and APTOS-DR datasets. In this study, a classification method based on four classes according to the Wong and Mitchell Classification based on HPR severity is presented; Normal, Mild, Moderate, and Severe or Malignant HPR. This classification is employed because in the previous classifications of HPR, Grade 1 and Grade 2 signs are not easily distinguishable and the grades of retinopathy were not closely associated with the severity of hypertension [4]. While KWB is a grading system widely cited by researchers, it has a subjective scaling amongst clinicians when grading the first and second stages of HPR [7]. According to [8], HPR classification is simplified for clinical practice using the WM classification. This study aims to improve HPR stage classification for easy diagnosis by eye specialists and makes use of the WM Classification. The current study explored the idea of applying CNN in classifying HPR using images obtained from patients diagnosed with hypertension and graded by ophthalmologists to improve HPR classification and it plays an important role in assisting physicians by speeding up the process of HPR diagnosis. This study used HPR features of the retina and exploited them in the classification of HPR into stages. The number of research carried in literatures with an attempt to classify HPR into multiple classes based on the severity is low. The classification scheme for HPR stages proposed in the current study is just only one of the very few studies already reported in the literature.

1.3 Research Objective

1.3.1 General Objective

The main objective of this thesis study was to use CNN to classify HPR in order to make it easier for medical specialists to diagnose. It involved using HPR features of the retina for classification.

1.3.2 Specific Objectives

- To obtain classified Fundus Camera retinal images of subjects with HPR;
- To use CNN models that allows accurate classification for hypertensive retinopathy;
- To quantify the performance of the developed AI based system based on Performance metrics;

1.4 Research Question/Hypothesis

- What distinguishes each class of HPR from the other?
- Can the pretrained models classify the images into the different classes correctly?
- Which pretrained model gives the best performance?

1.5 Scope and Thesis Organization

The scope of this study included the collection of retinal images from patients confirmed to have HTN and these images are then classified using deep learning CNN pretrained models. While, Matlab was used in programming. Images were collected locally from the Saint Paul Millennium Hospital and from the VICARV and AFIO dataset. As the study focused mainly on patients diagnosed with HTN, the images collected online are based on research work/datasets which stated that they had received images from hypertensive patients. The rest of the thesis has been arranged into four chapters. Chapter 2 gives in-depth details on HPR. Chapter 3 includes details on Deep Learning and literature review of previous works done on classification of HPR and deep learning models applied in retinopathy detection and classification. Chapter 4 includes the methods used and Chapter 5 gives the results which are portrayed through graphs and charts and Chapter 6 ends with conclusion and recommendations of the study.

CHAPTER 2

HYPERTENSIVE RETINOPATHY

2.1 Structure of the Eye: Overview

The eye is made up of the Anterior chamber which allows free flow of aqueous humor, Aqueous humor that clears liquid in the eye, Choroid that houses the blood vessels, Ciliary body that is in charge of producing aqueous humor and helps in focusing the lens of the eye, Cornea that helps to focus light and aids visual sharpness and clarity, Fovea which is the center of the macula, Iris that regulates how much light gets into the eye, Lens that is the clear structure inside the eye and focuses light, Macula which contains special light-sensitive cells that allow for details to be seen, Optic nerve that has over 1 million nerve fibers and links the eye to the brain's visual cortex, Pupil which is the dark opening in the middle of the iris and helps in adjusting based on the amount of light in an area, Retina that senses light and creates an electrical impulse that is then sent through the optic nerve, Suspensory ligament of lens which is composed of series of fibers that connect the ciliary body of the eye with the lens and helps to keep the eye in place, Sclera which is white visible portion of the eye that allows movement of the eye and Vitreous body that is the clear, gelatinous substance at the back part of the eye [9]. Figure 1 depicts basic parts of the eye.

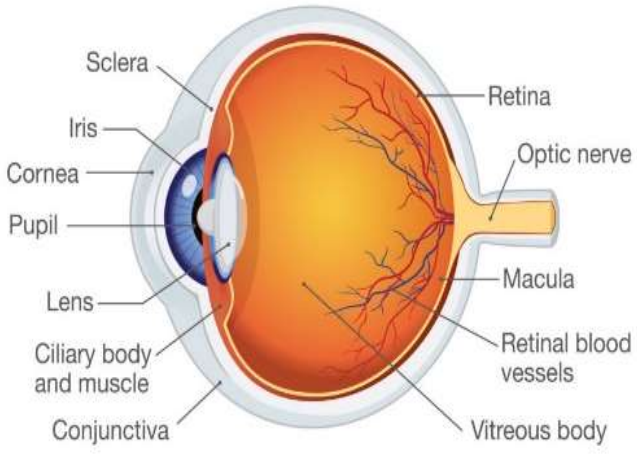


Figure 1: Eye Structure. ***[9]***

2.2 Structure of the Retina

The retina is the internal layer of the eye which is in charge of visual processing that converts light energy photons into 3D images. It is said to be the only adjunct of the brain that can be examined physically and gives eye specialists a unique view into concurrent pathology that affects the retina [10]. The retina has changing thickness in various sections:

- 0.4 – 0.5 mm at the edge of the optic nerve head;
- 0.2 – 0.25 mm at the central fovea;
- 0.07 – 0.08 mm at the foveal pit, and
- 0.1 mm at the ora serrata.

The macula is a very important section of the eye that ensures vision. It contains many photoreceptors and allows for good daylight vision [11]. The retina has ten definite layers of neurons linked by synapses [10]. The major layers of the retina are the photoreceptor layer and the pigment epithelium. Other layers are the external limiting membrane, outer nuclear layer, outer plexiform (synaptic) layer, inner nuclear layer, inner plaxiform layer, ganglion cell layer, nerve fiber layer, and inner limiting membrane [11] (see Fig. 2).

The pigment epitheliums are shaped like a hexahedral prism and are arranged in a line. They are part of the hematoretinal barrier which provides particular input of Choroid blood capillaries substances into the retina. The light-sensitive cells are also called the con-shaped and rod-shaped cells due to the shape of the outer segment. Rods aid eyesight in poor illumination and cones aid central form vision and color perception [11].

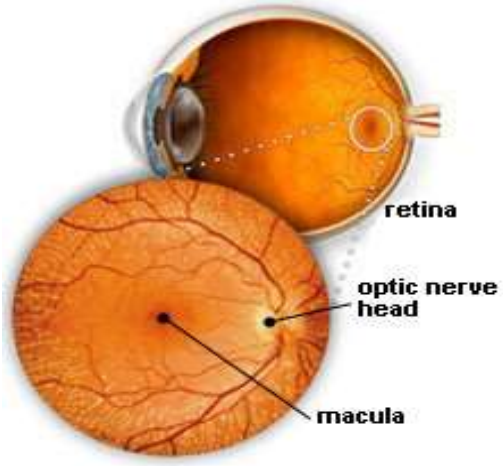


Figure 2: Structure of the Retina [11]

2.3 Hypertensive Retinopathy

There are two distinct differences between the blood vessels in the retina and other parts of the body [12]:

- Blood retinal barriers are present;
- There is lack of sympathetic innervations and blood flow is controlled by auto regulation mechanisms.

This means that an initial increase in the pressure of blood will be sent directly to blood vessels which will cause constriction and more increase of blood pressure results in muscle layers and endothelium damage [13]. HPR is characterized by the following phases:

Vasoconstrictive Phase – In this phase, autoregulatory mechanism causes vasospasm and retinal narrowing. The arteriole to venue ratio is decreased as compared to the normal 2:3 [13].

Sclerotic Phase – Constant blood pressure increase leads to thickening of the intima layer, hyperplasia in the media layer and hyaline degeneration in the arteriolar wall. This result in tightening of arteriolar narrowing, silver and copper wiring and arteriovenous (AV) cross changes. AV cross changes cause the veins where it occurred to look dilated and painful [13].

Exudative Phase – This occurs when the blood brain barrier is disrupted and there is blood and plasma leakage into the vessel wall stopping autoregulatory mechanism. As a result, retinal signs such as retinal haemorrhage, formation of hard exudate, retinal ischemia and smooth muscle cells necrosis occur [13].

Malignant Hypertension – This phase results in papilledema, nerve ischemia and choroidal arterioles necrosis. The latter causes Elschinig's spot, Siegrist's streak and Neurosensory RPE detachment.

2.4 Classification of Hypertensive Retinopathy

2.4.1 Keith-Wagener- Barker Classification [13]

- Group 1 - Slight constriction of retinal arterioles.
- Group 2 - Group 1 plus focal narrowing of retinal arterioles and AV nicking.
- Group 3 - Group 2 characterized by flame-shaped hemorrhages, cotton-wool spots and hard exudates.
- Group 4 - Group 3 plus optic disc swelling.

2.4.2 Scheie Classification [13]

HPR:

- Stage 0 - No visible abnormalities.
- Stage 1 - Diffuse arteriolar narrowing.
- Stage 2 - Stage 1 plus focal arteriolar constriction.
- Stage 3 - Stage 2 plus retinal hemorrhage.
- Stage 4 - Stage 3 characterized by hard exudates, retinal edema and optic disc swelling.

Arteriosclerosis:

- Stage 0 - Normal.
- Stage 1 - Broadening of arteriolar light reflex.
- Stage 2 - Stage 1 + AV crossing changes.
- Stage 3 - Copper wiring of arterioles.
- Stage 4 - Silver wiring of arterioles.

2.4.3 Wong and Mitchell Classification [14]

- No retinopathy – No visible retinal signs.
- Mild retinopathy – Characterized by narrowing of the arteriole, arteriovenous nicking and silver wiring. It has a modest association with clinical stroke risk, subclinical stroke, coronary disease and mortality.
- Moderate retinopathy – Characterized by hemorrhage, microaneurysm, cottonwool spot and hard exudates, It has a strong association with clinical stroke, subclinical stroke, cognitive decline and cardiovascular mortality.
- Malignant retinopathy – Characterized by moderate retinopathy added with optical swelling. It has a strong association with mortality.

Figure 3 presents a normal retinal image while Fig. 4 presents manifestations of the four different grades of HPR according to Keith-Wagner- Barker classification.

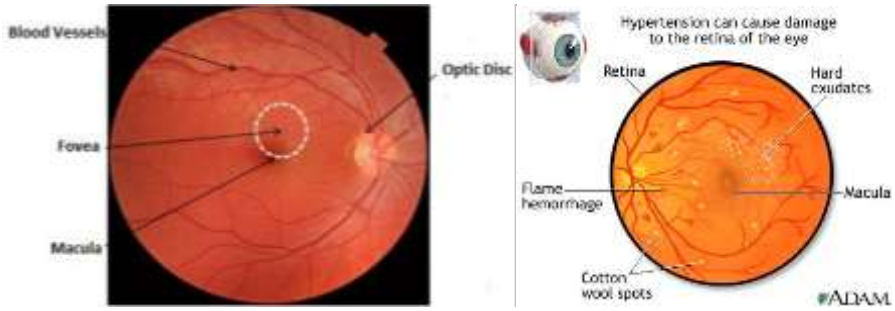
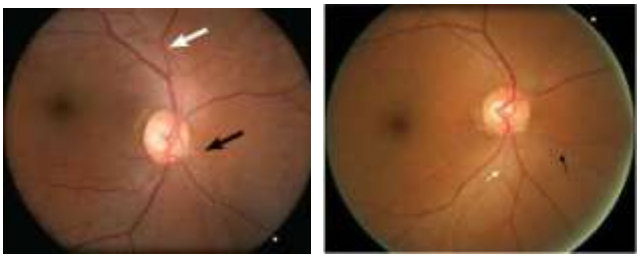
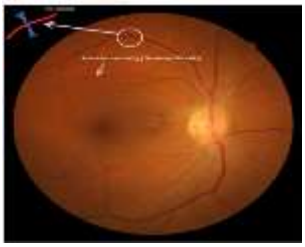


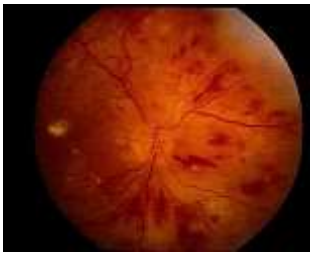
Figure 3: Normal Retina Image from a Fundus Camera (left) & Hypertensive Retinopathy labeled (right): Courtesy of *[15]* & *[16]*.



Grade 1: Generalized and focal arteriolar (white arrow) narrowing with moderate opacification (silver wiring) of arterioles (black arrow).



Grade 2: Grade 1 + Arteriole-venous nipping.



Grade 3: Grade 2 + flame-shaped hemorrhage + cotton wool spots + hard exudates.

Grade 4: Grade 3 + Papilledema.

Figure 4: Classification according to Keith-Wagner-Baker (Stage 1 to Stage 4):

2.5 Retinal Imaging Modalities

There are 2 imaging modalities commonly utilized for capturing pictures of the retina: fundus camera and Optical Coherence Tomography (OCT).

2.5.1 Fundus Camera

This is a special low power microscope to which a camera is connected and used for imaging of the retina [17]. The design of the fundus camera is established on the indirect ophthalmoscope. It creates a colour image of the retina. Images are acquired rapidly with high resolution and available immediately and can be enhanced [18]. Their angle of view is used to describe them. Two types of fundus camera exist: the standard view and the ultrawide field. The standard view fundus camera gives a 30° – 50° image which has the macula and optic nerve included in it. The ultrawide field provides a 200° retinal view which allows for 80% of the retina to be viewed beyond the macula to peripheral retina. The modern fundus photography has colour fundus photography, red free photography, stereo photography, hyper-spectral imaging, scanning laser ophthalmoscopy, adaptive optics and angiography [19].

2.5.2 Optical Coherence Tomography (OCT)

OCT is a non-invasive imaging modality used to perform a real-time, in-vivo optic biopsy of the human tissues [20]. It can be used to measure the retinal nerve fiber layer thickness. It gives a 3D view of the retina by making use of the principle of interferometry and confocal microscope [19]. OCT advancement has made it suitable for producing angiograms for the assessment of retinal vasculature.

CHAPTER 3

DEEP LEARNING

3.1 Deep Learning,

Deep Learning is subdivision of machine learning where a computer in order to learn ways of enhancing processes and creating new one, analyses algorithms and their results [21]. Deep learning can solve more complex problems than other types of machine learning.

3.2 Convolutional Neural Network

Convolutional Neural Networks (CNN) are types of neural network systems that are suitable for image recognition problems [22]. CNN takes in an input image, assigns learnable weights and biases to different objects in the image and is able to separate one from another [23]. The neurons in the convolutional layer of CNN are not all connected to the pixels in the input image as seen in the dense multilayer perceptron. Hence, a CNN is also called a sparse neural network. CNNs consist of three major layers: Convolutional Layer, Pooling Layer and Fully-connected (FC) layer as also depicted in Fig. 9.

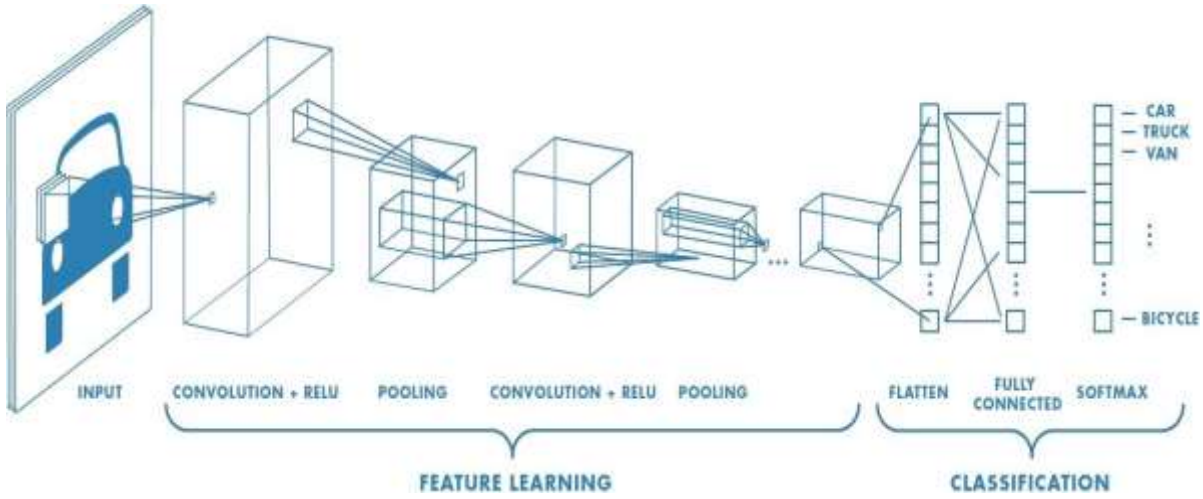


Figure 5: Convolutional Neural Network [23].

Convolutional Layer: This is the first layer of a CNN and the main structure of a CNN. It requires input data, a filter and a feature map. CNN has a feature detector/kernel which travels across the receptive areas of the image examining if the feature is present. For the case of an input which is a color image consisting of 3D matrix of pixels, the input would have three dimensions (height, width and depth) which fits the RGB in the image (See Fig. 10).

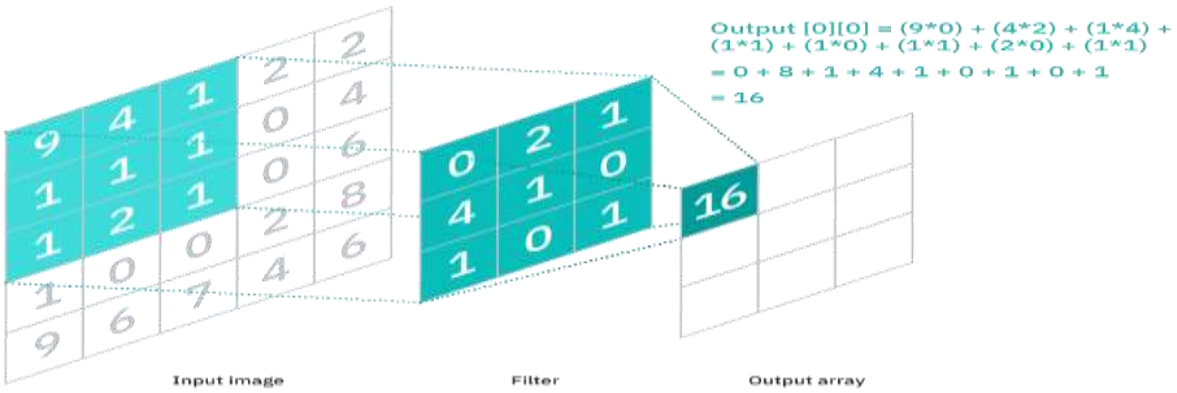


Figure 6: Convolutional Layer.

Pooling Layer: The pooling layer carries out reduction of dimension by reducing the number of parameters in the input. The pooling operation takes a filter across the whole input but in comparison with the convolutional layer, the pooling operation does not have any weights. Instead, the filter introduces an aggregation function to the values in the receptive field, increasing the array [24]. With the pooling layer, complexity is reduced, efficiency is increased and the risk of overfitting is limited. Pooling has two main types (see also Fig. 11):

- Max pooling: The maximum value of the pixel is selected as the kernel moves across the input and sent to the output array.
- Average pooling: The average value of the pixel is calculated as the kernels moves across the input and sent to the output array.

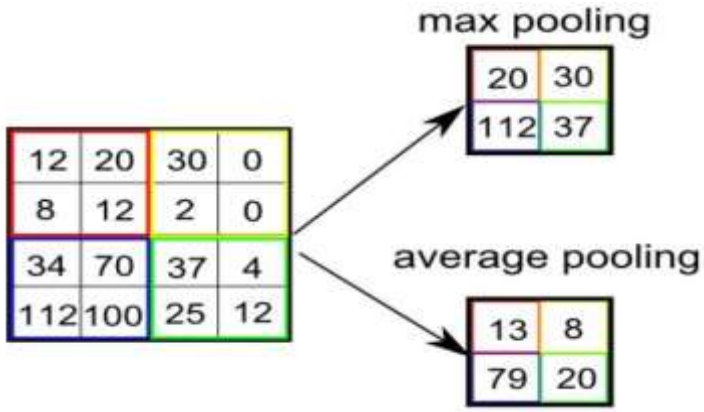


Figure 7: Types of pooling.

Fully Connected Layer: The fully connected layer has each node in the output layer connected directly to a node on the previous layer. Classification is carried out on this layer based on features

obtained from the previous layer and their different kernels. The layer usually uses a SoftMax activation function in order to classify inputs correctly as also depicted in Fig. 12.

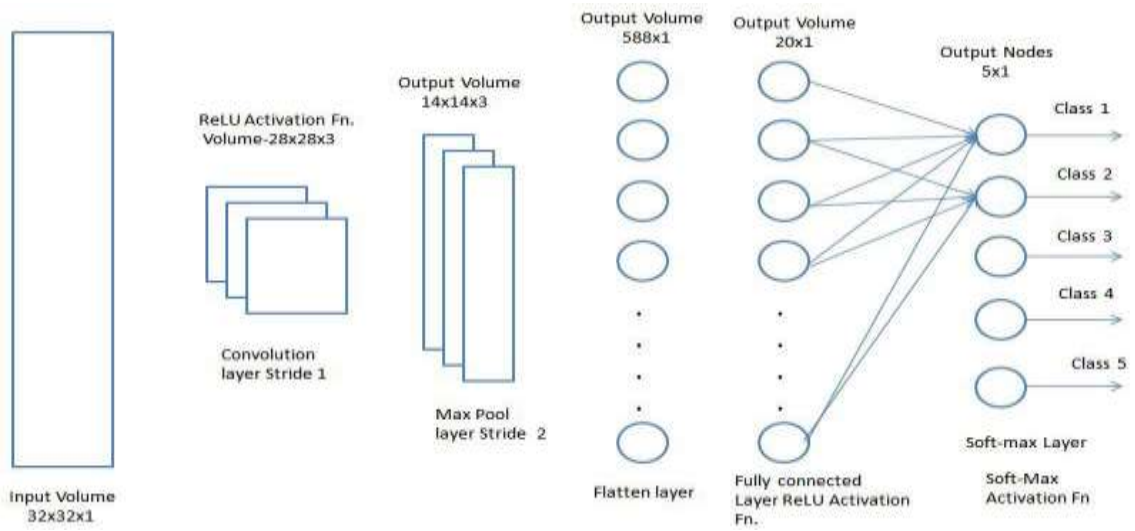


Figure 8: Fully Connected Layer.

3.3 Transfer Learning

This is a deep learning approach in which a model developed for a particular task is reused as the point of beginning for a new task. In transfer learning, a model is trained on a task and the knowledge acquired is applied on a second task that is related [25]. It increases the speed and performance for the second modeling task. Weights learned are transferred from one task into another to improve the performance of another related task. For example, if one wants to train a classifier to determine whether an image contains cats or dogs, the knowledge gained from training to classify rats and mice can be used. It is related mainly to challenges such as multi-task learning and concept drift. The first and middle layers are used in transfer learning while the end layers are retrained. This helps to support the labeled data of the task it was originally trained [26].

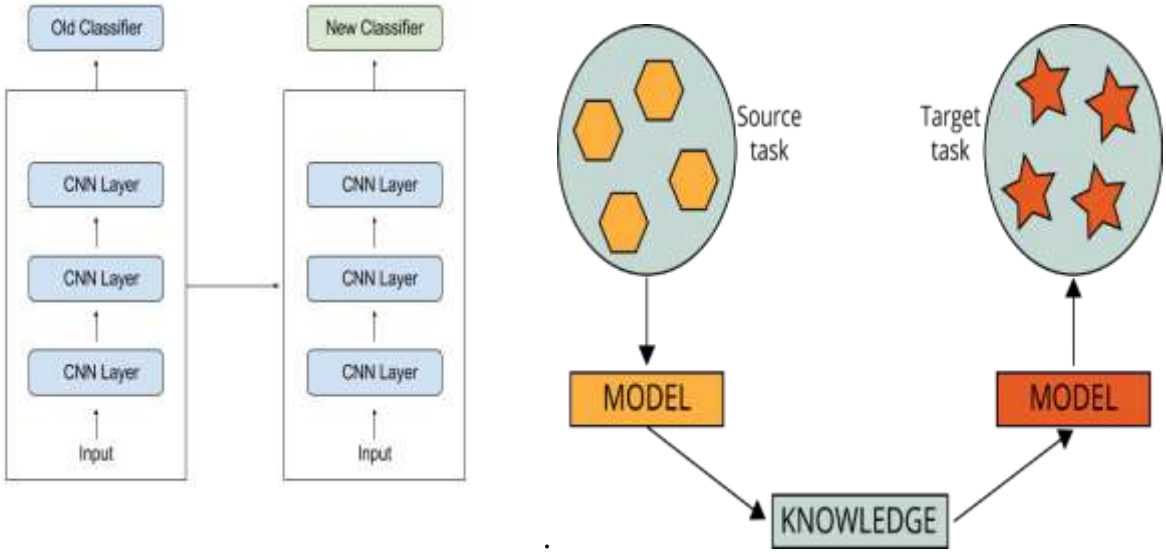


Figure 9: Transfer Learning [26] [27].

Some benefits of transfer learning are: training time is saved and neural networks perform better in most cases and the amount of data needed is less (See also Fig. 13). Below we will see typical approaches utilized to transfer learning.

a) Training a Model to Reutilize Again: In a situation where, one does not have enough data for training of a deep neural network, a related task having enough data can be used to train the neural network and the model achieved can be used as a starting point for the task that was intended to train. The type of problem being solved determines whether the whole model from previous training or only some layers will be used.

b) Applying a Pretrained Model: In this approach, an already existing pretrained model is used in training a task at hand. The number of layers and how many needs to be retrained relies on the type of problem being solved. Keras provides 9 pretrained models which can be used for prediction, feature extraction, transfer learning and fine tuning [26]. This approach is commonly used in deep learning.

c) Extraction of Features: The third approach is the use to deep learning to find out the best resemblance of the problem at hand. This involves finding the most important features in the task. This is also known as representation learning. Neural networks have the ability to learn the important features. The representation learned can be used for other problem solving. The topmost layers are used to get the right feature representation but the output of the network is discarded because it is too

task specific. Feature extraction is widely used in computer vision because it helps in dataset size reduction. It also reduces the time for computation [26].

3.4 Types of Learning

Three types of learning are common in the AI literature: supervised, unsupervised and reinforced learning as also depicted in Fig. 14.

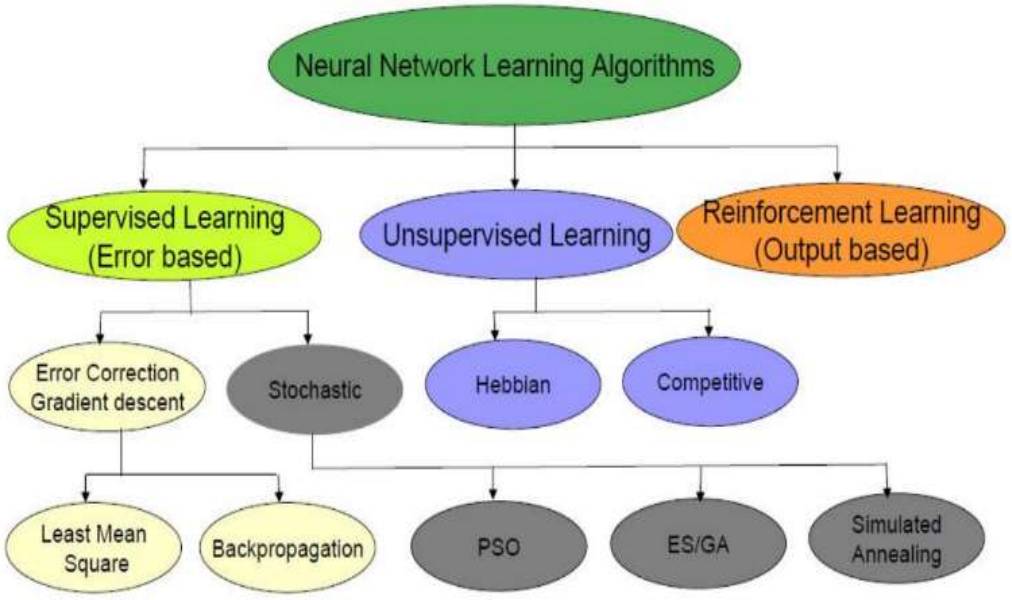


Figure 10: Classification of learning Algorithms [28]

Supervised Learning: This simply refers to guided learning by a teacher. Supervised learning systems include feedforward, functional link, product unit, recurrent and time delay. They require a training set that is made up of input vectors and a target vector associated [28].

Unsupervised Learning: In unsupervised learning, there is no help from a teacher in finding of patterns or features in the input data. The system does the learning by itself from the data without having a prior knowledge [28].

Reinforced Learning: In reinforced learning, even though a teacher is present the teacher does not give expected answers but will only indicate the answer/output given is wrong or correct and gives a penalty or reward respectively [28].

3.5 Training Parameters in CNN Architecture

3.5.1 Optimizers

By optimizers we are referring to algorithms that are used to change the attributes of a neural architecture such as learning rates, weights and the like in order to minimize losses. They help to achieve results faster by reducing training time [29]. There are different types of optimizers used in CNN implementation and some are described below:

Gradient Descent: Gradient descent is a first order optimizer that is dependent on the first order derivative of a loss function [29]. It computes how weights can be altered in order for the function to get to a minima. Gradient descent is used heavily in classification algorithms and linear regression. Backpropagation makes use of gradient descent [29]. In backpropagation, loss is passed on from one layer to another and the weights of the model are changed based on the losses in order for the loss to be reduced. Gradient descent is easier to compute, implement and understand [29]. Some disadvantages are that Gradient descent may trap at local minima. If the data is very large, it may take a long time to converge at the minima because the model parameters are changed after gradient is computed on the whole data. It also requires a large memory space to compute gradient descent on the whole data [29].

$$\theta = \theta - \alpha \cdot \nabla J(\theta) \qquad \textit{Eqn 1}$$

Stochastic Gradient Descent (SGDM): SGDM is a form of gradient decent that tries to update the weights of the model more frequently. Weights are changed after loss computation on each training sample [29]. SGDM converges much faster because of the frequent updates of the model's weights. It does not require much memory because there is no need for the loss function values to be stored [29]. Some disadvantages are that there is high variance in the weight of the models, in order to achieve the same convergence as gradient descent, the learning rate has to be reduced slowly and it can shoot even after obtaining the global minima [29].

$$\theta = \theta - \alpha \cdot \nabla J\big(\theta; x(i); y(i)\big) \qquad \textit{Eqn 2}$$

where **{x(i), y(i)}** are the training samples.

Adam: Adam also known as the Adaptive moment estimation and uses momentum of the first and second order. Adam stores the exponentially decaying average of past squared gradients and also keeps an exponentially decaying average of past gradients **M(t)**. It calculates the squared gradient and the

gradient's exponentially moving average. The mean **M(t)** is the value of the first moment and the uncentered variance of the gradients **V(t)** is the value of the second moment.

$$\widehat{\boldsymbol{m}}_{\boldsymbol{t}} = \frac{\boldsymbol{m}_{\boldsymbol{t}}}{\mathbf{1}- \boldsymbol{\beta}_{\mathbf{1}}^{\boldsymbol{t}}} \qquad \textit{Eqn 1}$$

$$\widehat{\boldsymbol{v}}_{\boldsymbol{t}} = \frac{\boldsymbol{v}_{\boldsymbol{t}}}{\mathbf{1}- \boldsymbol{\beta}_{\mathbf{2}}^{\boldsymbol{t}}} \qquad \textit{Eqn 2}$$

The mean of **M(t)** and **V(t)** are taken so that $E[m(t)]$ can be equal to $E[g(t)]$ where $E[f(x)]$ is an expected value of a given function **f(x)** (Doshi, 2019).

The weights are updated based on:

$$\boldsymbol{\theta}_{\boldsymbol{t+1}} = \boldsymbol{\theta}_{\boldsymbol{t}} - \frac{\eta}{\sqrt{\widehat{\boldsymbol{v}}_{\boldsymbol{t}} + \in}} . \widehat{\boldsymbol{m}}_{\boldsymbol{t}} \qquad \textit{Eqn 3}$$

Adam is very fast and converges quickly and it also rectifies vanishing learning rate and high variance (Doshi, 2019). One disadvantage is that it is expensive computionally.

3.5.2 Activation Function

Activation function is a very important part of a neural network architecture. The activation function controls how the input weighted sum is changed into an output from a node(s) in a layer of the model. It is used within or next to the internal processing of each node in the model. The activation function chosen in a network affects the network's performance and capability [30]. There are different types fo activation functions. The major ones are Rectified Linear Activation (ReLU), Logistic (Sigmoid), Hyperbolic tangent, Linear and Softmax.

Rectified Linear Activation (ReLU): ReLU is one of the most common type of activation functions used in hidden layers. It is easy to use and also effective at suppressing the limitations of other activation functions. It is less vulnerable to vanishing gradients which prevents the training of deep models. It however is susceptible to saturated or dead units' problem. It is calculated as follows [30]:

$$\boldsymbol{max}(\mathbf{0.0}, \boldsymbol{x}) \qquad \textit{Eqn 4}$$

If the input (x) is negative then 0.0 is returned else the value is returned [30] (see also Fig. 15).

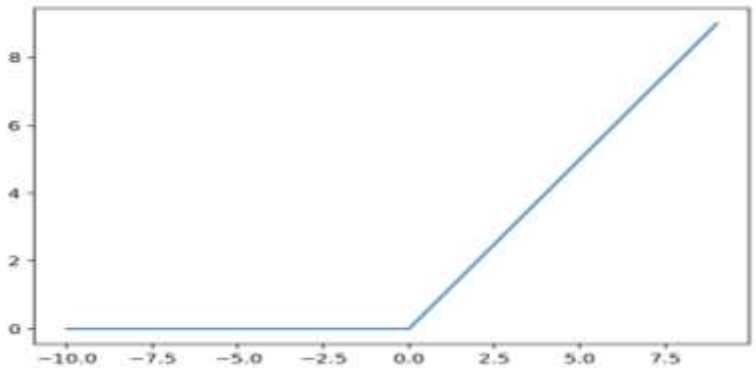


Figure 11: ReLU Activation function plot [30].

Sigmoid Activation Function: Sigmoid activation function or logistic function takes in any real value as an input and outputs a value within the range of 0 to 1 [30]. The bigger the input, the closer the value of the output will be close to 1 and the smaller the input, the closer the value of the output will be to 0 (see Fig. 16). It is calculated as [30]:

$$\mathbf{1.0} \,/\, (\mathbf{1.0} + \boldsymbol{e^{-x}}) \qquad \textit{Eqn 5}$$

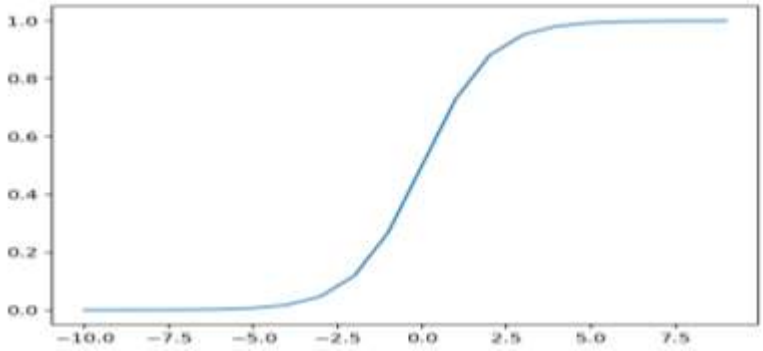


Figure 12: Sigmoid Activation Function [30].

Linear Activation Function: Linear activation function is also referred to as identity or no activation function because it does not change the input's weighted sum in anyway but rather returns the value directly.

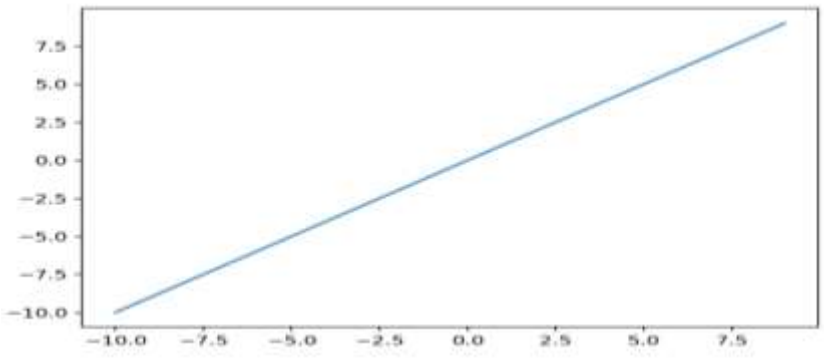


Figure 13: Linear Activation Function.

Softmax Activation Function: The softmax function returns an output of vector values which sum up to 1 and can be interpreted as probabilities of membership of a class. It is a softer version of argmax function which allows a probability like output of a winner take all function. Softmax takes in a vector of real values as the input and gives an output of a vector of real values which sum up to 1 like probabilities. It is calculated as follows:

$$\boldsymbol{e^x} \;/\; \boldsymbol{sum}(\boldsymbol{e^x}) \qquad \textit{Eqn 6}$$

where **x** is vector of inputs.

3.6 Batch Normalization

Normalization is a type of tool used in pre-processing to guide the numerical data to a scale which is common without changing its shape. In normalization, the data is transformed to have a mean of 0 and standard deviation of 1 [31]. This process takes place in batches instead of as a single input [31]. Batch Normalization is a process that uses the addition of extra layers to make neural architectures more stable and faster in deep neural networks [31]. Normalization and standardization operations are performed by the new layer on the output of the previous layer coming in as an input [31].
In the first step the mean of the hidden activation is computed as:

$$\boldsymbol{\mu} = \frac{\mathbf{1}}{\boldsymbol{m}} \sum \boldsymbol{h_i} \qquad \textit{Eqn 7}$$

where **m** is the number of neurons and **h** is a layer.

Next, the standard deviation of the hidden activations is calculated:

$$\boldsymbol{\sigma} = \left[\frac{\mathbf{1}}{\boldsymbol{m}} \sum (\boldsymbol{h_i} - \boldsymbol{\mu})^{\mathbf{2}}\right]^{\mathbf{1/2}} \qquad \textit{Eqn 8}$$

Next, the hidden activations are normalized:

$$h_{i(norm)} = \frac{(h_i - \mu)}{\sigma + \varepsilon} \qquad \text{Eqn 9}$$

Here, the mean is subtracted from each input and the whole value is then divided with the sum of the standard deviation σ and the smoothening term ε.

3.7 Data Augmentation

Data Augmentation is used to solve the problem of low dataset size. It involves making minor changes to an existing dataset. These changes include: Flips, rotation, translation and the like. A CNN network can classify images even when they have been placed in a different orientation. There are two ways to augment a dataset. One option is by first performing the necessary transformation on the data just before inserting it into the model (offline augmentation) and another option is to perform these transformations first on a mini batch just before imputing it into the model (online augmentation) [32].

Below are few data augmentation techniques.

- **Flip:** With flip, images can be flipped vertically and horizontally. Flipping vertically rotates an image by 180 degrees and then flips it horizontally.
- **Rotate:** An important thing to note about rotation is that the image dimensions change after rotation. If the images are square, the images should be rotated at right angles to in order retain the size of the images. If the images are rectangular then rotating by 180 degrees retains the initial size. Using finer angles to rotate images will change the size of the final images.
- **Scale:** An image can be scaled either outward or inward. In the outward scaling, the size of the final image is bigger than the size of the original images. Normally the image framework will cut out a section of the new images with a size equal to the original images.
- **Crop:** In cropping, a section of the original image is randomly sampled and then resized to the size of the original image. It can be called random cropping.
- **Translation:** In translation, the image is moved along the X or Y direction or in both directions. It is quite useful as most image objects can be located in any place on the image. The CNN is forced to check everywhere on the image.

- **Gaussian Noise:** The right amount of Gaussian noise can enhance the learning capability of the model and prevent overfitting. Gaussian noise (with a mean of 0) has data point in all frequencies which are really good for distorting high frequency features.

3.8 Review of Previous Works

In a recent work done by a group of researchers [33] on evolving CNN for glaucoma diagnosis, they made use of a generic algorithm to optimize the architecture of CNN through an evolution that helps in diagnosis of glaucoma using eye's fundus images acquired from RIMONEr2 dataset. The technique used here was based on neuroevolution. This technique executes optimization of components, topologies and hyperparameters. They made use of 227 images for training, 92 images for validation and136 images for test data. All processes were carried out using Python with Keras API and they finally achieved an overall accuracy of 91%. In his work on detection and grading of trachoma using deep CNN [34] a researcher proposed a novel segmentation algorithm to extract the ROI of the eyes and used predetermined coordinates from the dataset and used Gabor filter for feature extraction. He made use of CNN for classification and a 4-way SoftMax for grading into distinct classes. The system proposed had an accuracy of 98% for training, 97.9% for testing in trachoma grading and detection. In another recent work [35], researchers proposed a CNN based method on the use of thermal images to detect diabetics retinopathy. The images are first converted into grey scale images and features required are extracted. Five stages of classification was done here. They recorded more accuracy than previous works and the method used is helpful in detecting diabetic retinopathy in large proportion. [36] In their research work deployed three state of the art CNN architectures (AlexNet, VGG16 and InceptionNet V3) to classify the different stages of diabetic retinopathy. They applied the stochastic gradient descent with momentum in order to determine the global minimum of the cost function for the 3 architectures used. A total of 166 images were chosen from the kaggles database screened and relabeled by eye specialists. A 5-fold cross validation process was used to train and test the model. [37] In their research work designed a three state of the art CNNs and sequenced their features in order to create a system that was able to classify the different Diabetes stages from color fundoscopic images. They made use of the Kaggle dataset which contained 500 images of the retina. Their study shows an accuracy of 80.1% after the concatenating of VGG16, Alexnet and InceptionNet V3. In a study by [38]**,** the authors gave a summary of various deep learning methods applied in diabetic retinopathy, glaucoma, age-related macular degeneration and retinopathy of prematurity using photographs generated on a fundus camera. This is shown in Table 2 below:

Table 1: Review Summary of Deep Learning Systems by [38]

DL systems	Year	Test data sets	Test images (n)	CNN	AUC	Sensitivity (%)	Specificity (%)
Referable diabetic retinopathy							
Abràmoff *et al*[14]	2016	Messidor-2	1748	AlexNet/VGG	0.98	96.80	87.00
Gulshan *et al*[12]	2016	Messidor-2	1748	Inception-V3	0.99	87	98.50
						96.10	93.90
		EyePACS-1	9963		0.991	90.30	98.10
						97.50	93.40
Gargeya and Leng[15]	2017	Kaggle images	75 137	Customised CNN	0.97	NA	NA
		E-Ophtha	463		0.96	NA	NA
		Messidor-2	1748		0.94	NA	NA
Ting *et al*[11]	2017	SiDRP 14–15	71 896	VGG-19	0.936	90.50	91.60
		Guangdong	15 798		0.949	98.70	81.60
		SIMES	3052		0.889	97.10	82.00
		SINDI	4512		0.917	99.3	73.3
		SCES	1936		0.919	100	76.30
		BES	1052		0.929	94.40	88.50
		AFEDS	1968		0.98	98.80	86.50
		RVEEH	2302		0.983	98.90	92.20
		Mexican	1172		0.95	91.80	84.80
		CUHK	1254		0.948	99.3	83.10
		HKU	7706		0.964	100	81.30
Abràmoff *et al*[28]	2018	10 primary care practice sites from the USA	892 patients	Alex/VGG	NA	87.2	90.7
Glaucoma suspect*							
Ting *et al*[11]	2017	SiDRP 14–15	71 896	VGG-19	0.942	96.40	93.20
Li *et al*[16]	2018	Guangdong	48 116		0.986	95.60	92.00
Age-related macular degeneration							
Ting *et al*[11]	2017	SiDRP 14–15	35 948	VGG-19	0.932	93.20	88.70
Burlina *et al*[17]	2017	AREDS	120 656	AlexNet, OverFeat	0.940–0.96	NA	NA
Grassmann *et al*[18]	2018	AREDS	120 656	AlexNet, GoogleNet, VGG, Inception-V3, ResNet, Inception-ResNet-V2	NA	84.20	94.30
Retinopathy of prematurity							
Brown *et al*[19]	2018	i-ROP	100	Inception-V1 and U-Net	NA	100	94

Studies of interest directly related to HPR are discussed below. In a study presented in [39], the authors proposed a system for early detection of HPR using CNN and achieved an accuracy of 98.6% using the DRIVE image dataset. They made use of grayscale fundus images which were resized to 32 by 32 pixels and then converted to the CSV file format before inputted to the CNN. RELU activation, 2 convolutional layers, 2 pooling layers which used max pooling and a softmax output were used for testing. All convolutional layers used a 5 by 5 kernel with filters that had certain coefficient values. The CNN model was implemented using R. In another study [40], researchers developed a retinal vein segmentation method making use of the randon transform and Hough transformation for optical disk detection to get the artery to vein ratio for HPR detection and they achieved an accuracy of 92% using the DRIVE data sets. They made use of a couple of dimensional data reduction and backpropagation neural network. They used principal component analysis for data dimensional reduction. Their method showed a reduction of image data up to 99.9%, and allows for a reduced amount of time and computational resource consumption when compared with other methods. In another study [41],

efforts were employed to use a support system for the detection of HPR using segmentation of blood vessels using randon transform. The artery to vein ratio (AVR) was found by detecting the optical disc and marking out the region of interest (ROI) as artery and vein which are used to calculate the AVR. The DRIVE database was used and an accuracy of 92% was achieved in segmentation of blood vessels. To validate their model, 30 images were taken from the KMC hospital, Manipal, India and were classified into normal and HPR using the AVR by an eye specialist and the model could classify 25 images as HPR and 5 as normal images. The authors in [42] proposed a novel system to detect HPR based on trained features (TF-L) and dense feature transform layer (DFT-L) to the deep residual learning (DRL) methods. They obtained a sensitivity of 93%, specificity of 95%, and accuracy of 95%. Another study [43] utilized backpropagation neural network for retinal fundus identification. Prior to identification, they performed pre-processing (picked the green channel, applied contrast limited adaptive histogram equalization (CLAHE), morphological close, background exclusion, thresholding and connected component analysis) and feature extraction using zoning. They obtained and accuracy of 95% with maximum epoch of 1500. Dai and colleagues [44] used deep learning techniques to explore HTN effect on morphological features of the retinal microvasculature on retinal images in East Asian population. Two classes were used here: normal and HPR. A total of 2012 retinal images were acquired from 1007 hypertensive patients and 1005 normotensive control patients. The images acquired were segmented to obtain vessels morphological information and this information was used for training the CNN. They used a gradient weighted class activation mapping to generate heat maps for the HTN class. The accuracy of their model was 60.96% with a specificity of 51.54%, precision of 59.27% and a recall of 70.48%. The study suggests that HTN effect on retinal microvascular morphology occurs mainly at vessels branching and that the changes of the branching pattern of retinal vessels is probably the most important in response to blood pressure elevation.

In a recent study by [6], they employed a semantic instance segmentation in DenseNet architecture for the classification of HPR into 5 stages. They made use of pretrained HPR related lesions in their HYPER-RETINO system and used images from the PRV-HR, DRIVE, DR-HAGIS, DRIVE, DiaRETDB&1, DR1&DR2, Kaggle-DR, and APTOS-DR datasets. They obtained a sensitivity (SE) of 90.5%, specificity (SP) of 91.5%, accuracy (ACC) of 92.6%, precision (PR) of 91.7%, Matthews correlation coefficient (MCC) of 61%, F1-score of 92% and area-under-the-curve (AUC) of 0.915 on 1400 HR images. One thing noted is these datasets used in [6] were not exclusive to retina images gotten from patients with hypertensive retinopathy but contain images gotten from other eye

retinopathies with HPR related lesions. Most research papers focused on vessel segmentation and optic disk detection to diagnose HPR and most of the data sets used are not focused on HP patients. The current thesis work focuses on retinal features peculiar to HPR in order to detect and classify the different stages of HPR using CNN. This research work uses only dataset retrieved from patients with HPR.

CHAPTER 4
METHODOLOGY

4.1 Introduction

This chapter gives an overview of the methods used for classification of HPR using CNN. It involves five main stages: image acquisition, image preprocessing & data augmentation, pretrained model importation, Training parameter setting, model training and model validation.

4.2 Study Design

This research study was both a retrospective and prospective study. Data was collected from 2 online databases and also from the hospital locally. It was a study based on images collected from patients diagnosed with HTN.

4.3 Dataset and Computer model

The first step was to acquire images of the HPR retina. This study includes images collected from 2 online databases (VICARV and AFIO) and Saint Paul's Millenium Hospital, Ethiopia. A total of 273 pictures of the retina was acquired locally. 58 images were acquired from the VICARV database and 99 images were acquired from the AFIO [45]. A total of 430 images were collected and divided into training data and validation data. All images were in RGB color and JPEG format (see Table 3 and Table 4).

The AFIO dataset [45] contains 99 fundus images captured of the retina taken from patients at the Armed Forces Institute of Ophthalmology (AFIO) in Rawalpindi, Pakistan. Patients captured are between the ages of 25 and 80. The images are in JPEG format and were centered around the optical disc with 30% FOV and are 1504 by 1000 in size. Vessel segmentation was done by 4 experts at the hospital. The images were taken using a TOPCON TRC-NW8 system.

The VICARV dataset [46] contains 58 images which were used for A/V ratio computation. Images are of size 768 by 584. Vessel segmentation was done by 3 experts. Image vessels are measured at different radii from the optic disc. The images were captured using a TopCon non-mydriatic camera NW-1PP model.

The hospital data includes 273 images captured from 116 patients with HTN. The age range was between 21 and 85. The mean systole and diastole were 140/80 with the highest being 180/90 and the lowest 120/70. The patients included were diagnosed with HTN only. Images were then classified independently by ophthalmologists from two different hospitals. The images were classified into 4 stages: normal, mild, moderate and severe. The data classified was divided into training set and testing set. The training set was used to train the neural network and the validation set was used to test the system. 70% of the data was allocated to training and 30% allocated to validation. Five preexisting models were used to train and validate the images through transfer learning. The hardware and software systems used were: a processor with Intel (R) Core (TM) i7-6500U CPU @ 2.50GHz, 2592 MHz, 2 Core(s), 4 Logical Processor(s), 8GB RAM, HP Spectre x360 convertible for importing, training and modeling. Figure 19 depicts a histogram showing the composition of the training and validation data while Fig. 20 presents typical images acquired from AFIO, VICARV and SPH databases.

Table 2: Data Sources

Data Source	**Number of images**
Hospital Data	273
VICARV Data	58
AFIO Data	99

Table 3: Dataset Composition

	Label	**Count**
1	Mild HP	306
2	Moderate HP	66
3	No Visible HP	5
4	Severe HP	53

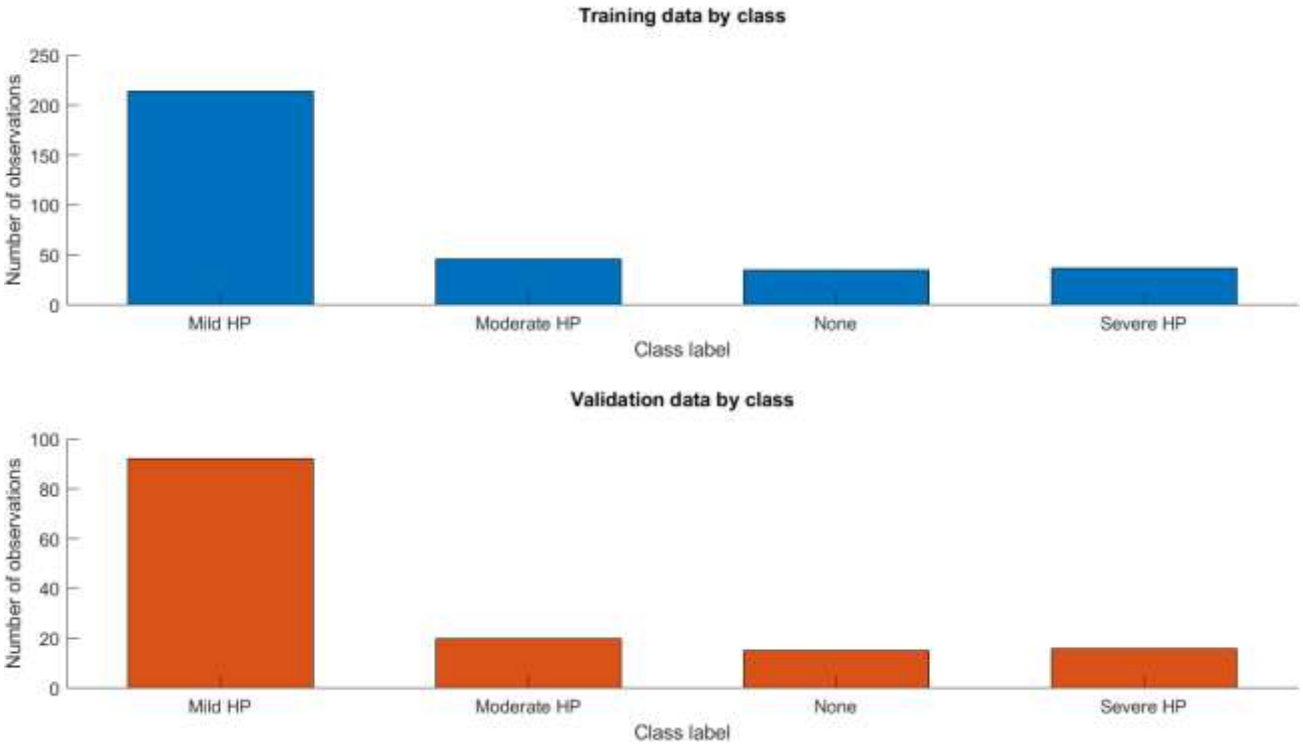


Figure 14: Histogram Representation of Training and Validation Datasets.

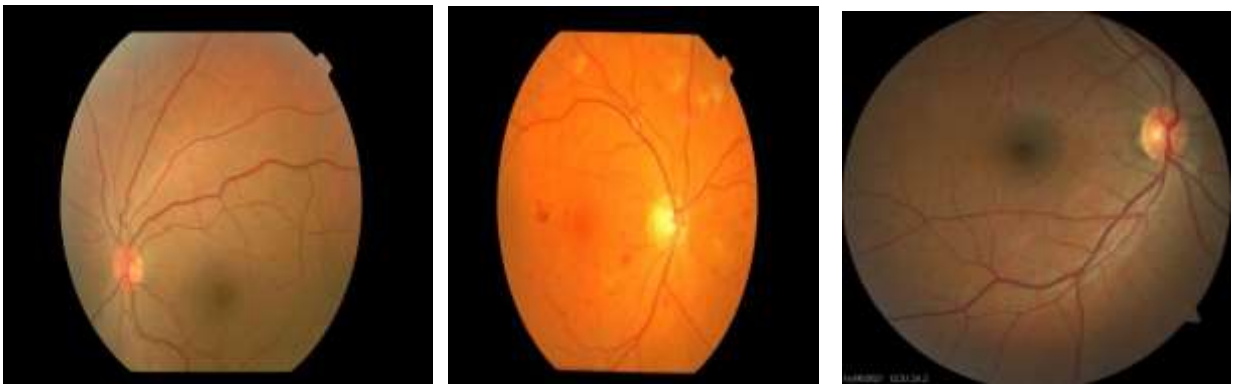


Figure 15: Images captured from AFIO, VICARV and SPH.

Figure 21 presents the general flow chart of the proposed methodology including image acquisition, data augmentation and image preprocessing, pretrained model importation, training parameter setting, model training as well as model validation. The major steps involved in the proposed methodology are explained in the subsequent sub sections.

4.4 Data Augmentation and Image Pre-processing

Data augmentation was carried out for the class labelled 'None', i.e., those with no signs of HP, because of the number of images in that class. The images were rotated at angles 10, 20, 30, 45, 60, 75 and 90 degrees. The images were also flipped horizontally and vertically. Due to the difference in size of images from the images collected from the three databases, the images were resized to 227-by-227. Hence, all input colour images under consideration are of size $227 \; x \; 227 \; x \; 3$. All images are in .jpeg format and the colour channel was set to RGB as in a typical hospital setting, images captured by the fundus camera are in RGB and .jpeg format. That replicates that scenario on the images acquired in order to achieve good performance when incorporated into the hospital setting.

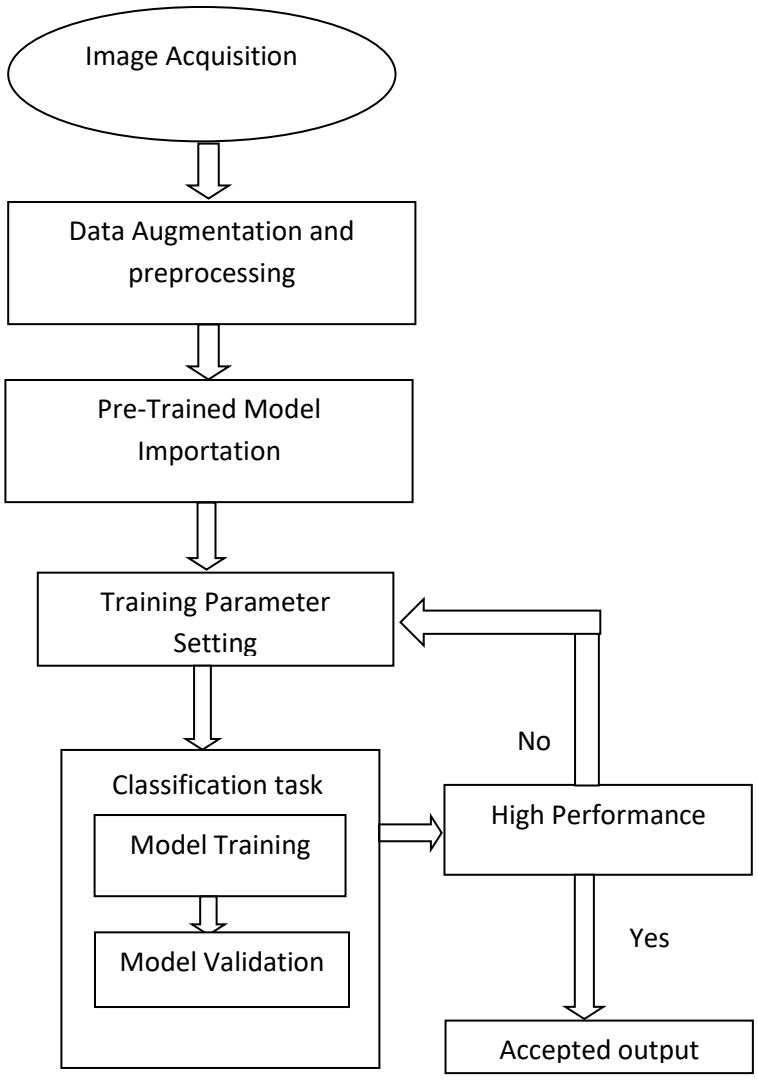


Figure 16: Methodology.

The Matlab implementation of the resizing operation is shown below.

Matlab Implementation

```
imds = imageDatastore("C:\Users\HP\Desktop\Hypertensive Retinopathy\Grading\Grader 2\","IncludeSubfolders",true,"LabelSource","foldernames");
[imdsTrain, imdsValidation] = splitEachLabel(imds,0.5);

% Resize the images to match the network input layer.
augimdsTrain = augmentedImageDatastore([224 224 3],imdsTrain
augimdsValidation = augmentedImageDatastore([224 224 3],imdsValidation);
```

4.5 Model Training and Validation

Training Parameter of the Network: In choosing the set of training parameters used for the training of the model, a series of trial and error was carried out. Two optimizers were tested: the SGDM and Adam optimizer and it was observed that the SGDM performed much better than the Adam optimizer hence the SGDM optimizer was selected. Three learning rates: 0.01, 0.001 & 0.0001 were also tried in the training algorithm. The learning rate of 0.001 was chosen as the network achieved a higher accuracy than 0.01 & 0.0001 in the training and validation sets. The number of epochs chosen was also set at 15. ReLU was used for the activation function and SoftMax used for the classification task. Since the final layer of the networks used is a classification layer, the loss function used is the crossentropy loss. The training parameters used for training the model are listed in Table 5.

Table 4: Parameters used for training.

Parameter	**Unit**
Optimizer	Sgdm
Validation frequency	30
Number of Epoch	15
Mini Batch Size	30
Initial learning rate	0.001
Activation Function	ReLU
Loss function	categorical_crossentropy

Setting Training Options: Below shows the Matlab implemetation of setting operation of the training options while Fig. 22 shows a snapshot taken during the setting.

Matlab Implementation:

```
opts = trainingOptions("sgdm",...
    "ExecutionEnvironment","auto",...
    "InitialLearnRate",0.001,...
    "MaxEpochs",15,...
    "MiniBatchSize",30,...
    "Shuffle","every-epoch",...
    "ValidationFrequency",30,...
    "Plots","training-progress",...
    "ValidationData",augimdsValidation);
```

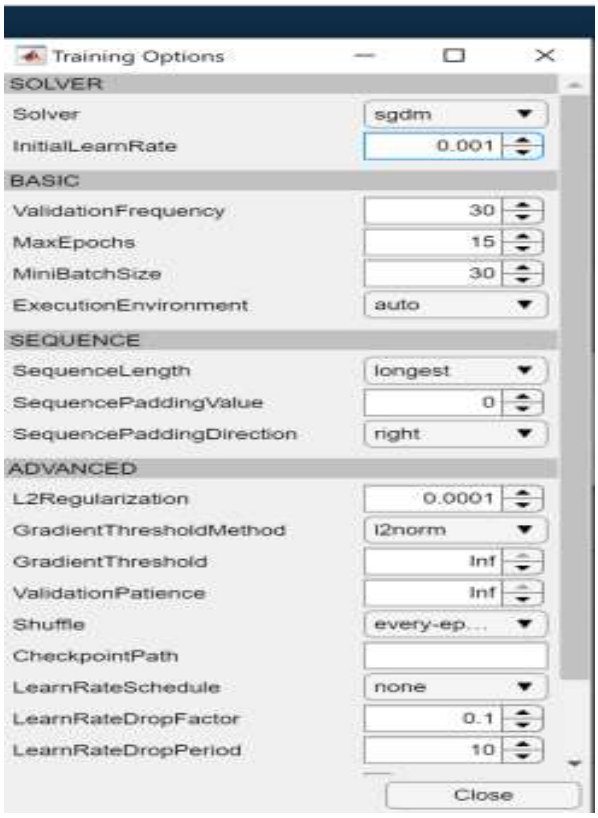


Figure 17: Training Options.

4.6 Network Importation

Five CNN architectures were imported: ResNet-101, AlexNet, GoogleNet, VGG-19, and Xception. The models were imported into Matlab. For each model, the fully connected layer was replaced and the output of the layer was changed to 4 and 2 respectively for the two types of classification done (that corresponds to classification into 4 and 2 classes, respectively). The classification layer was also changed because it still had the output from when the models were trained on the ImageNet dataset.

4.6.1 Convolutional Neural Network Architectures:

There are several neural network architectures often used in the literatures however, five major ones: AlexNet, VGG-19, GoogleNet, ResNet-101 and Xception, are explained below.

- **VGG-19:** VGG-19 was built and trained by two researchers, Karen Simonyan and Andrew Zisserman, in 2014 at the University of Oxford [27] which is reported in their paper entitled "Very Deep Convolutional Networks for Large-Scale Image Recognition" and published in the year 2015. VGG-19 is a type of CNN pretrained model which has a depth of 19 layers. About a million plus images from the ImageNet dataset were trained using this network and the network can classify up to 1000 classes. The image inputs for this model are 224 by 224 colored images. The graphical illustration of the VGG-19 architecture is depicted in Fig. 23.

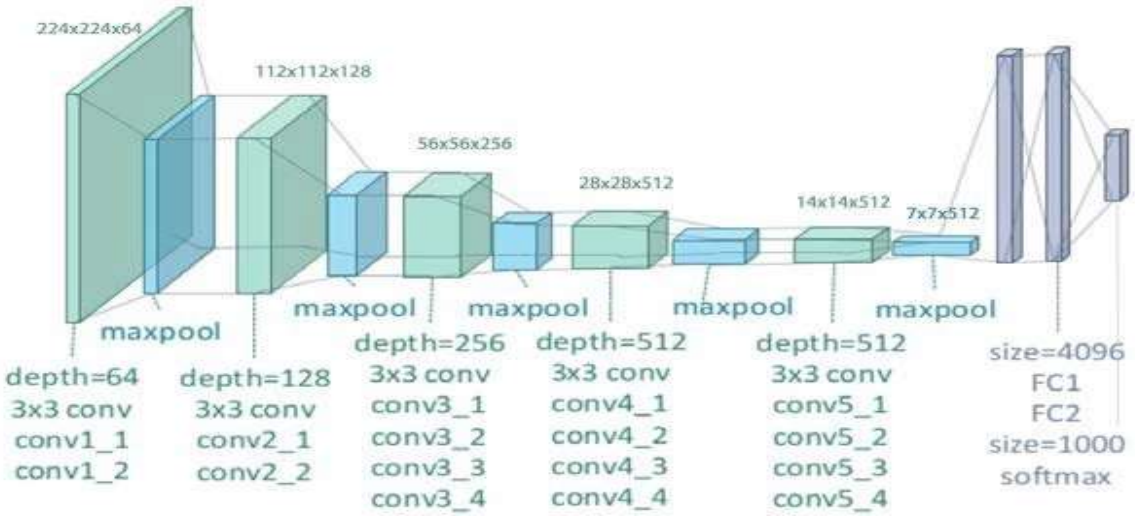


Figure 18: VGG-19 Network [47].

- **GoogleNet:** The GoogleNet was built by Google and has a depth of 50 layers. The network pretrained version with weights from ImageNet can classify up to 1000 classes. The input size of images is 224 by 224. Inception also called GoogleNet was the winner of the ImageNet competition in 2014. Figure 24 graphically illustrates the GoogleNet architecture.

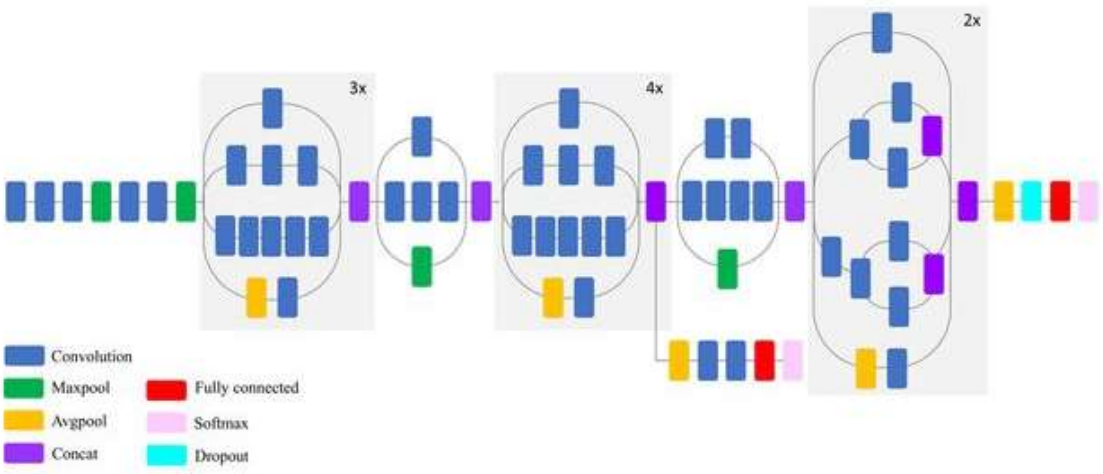


Figure 19: GoogleNet Structure.

- **ResNet-101:** ResNet-101 (Residual Network) is a type of CNN pretrained network that has 101 in depth (see the graphical illustration in Fig.25). It can classify images into 1000 classes. The network has an input size of 224 by 224 for images. This network was also trained on a million plus images on the ImageNet dataset [48]. ResNet-101 won the first place in the ILSVRC 2015 classification competition having a top-5 error rate of 3.57%. Each block of ResNet-101101 is 3 layers deep (see Fig. 26).

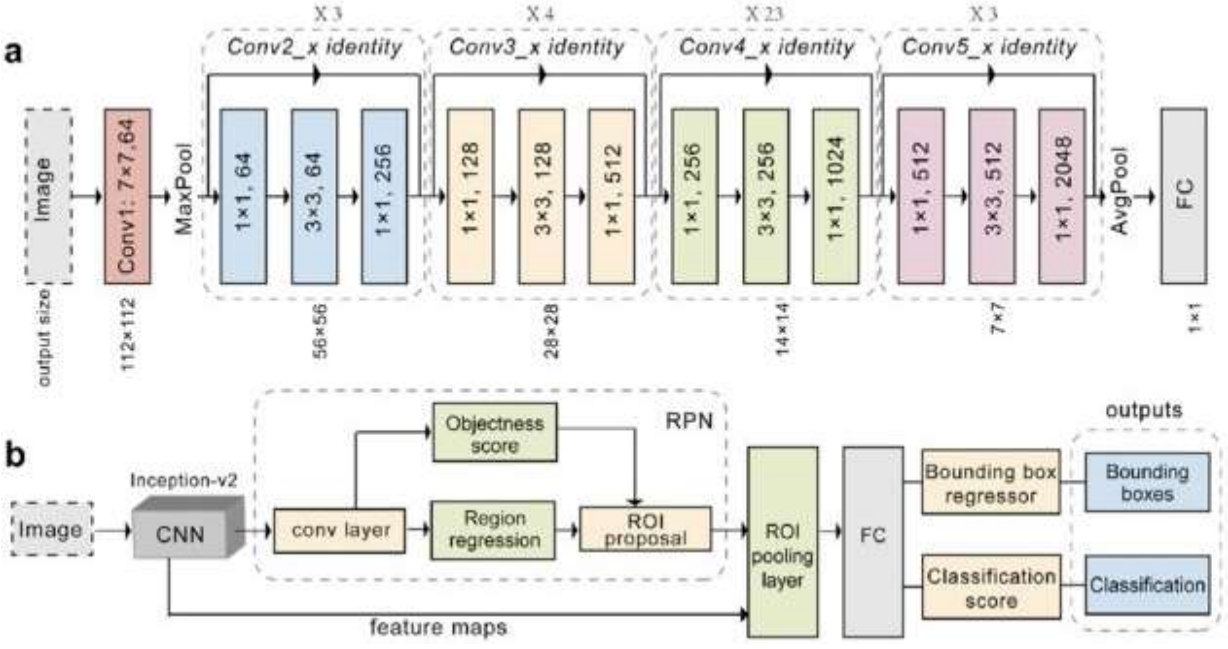


Figure 20: ResNet-101 Structure [49].

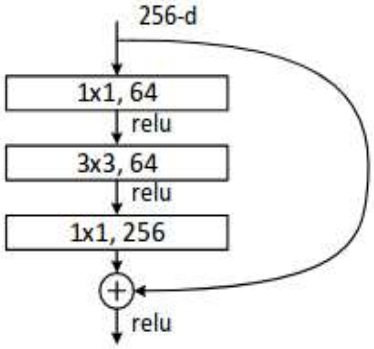


Figure 21: 3-layer Block [50].

- **AlexNet:** The AlexNet comprises of 8 layers which includes 5 convolutional layers and 3 layers that are fully-connected. This network utilizes the ReLU in place of the tanh function which was the standard initially at that time. One main advantage is reduction in training time and error percentage. The network also allows for multi-GPU (graphics processing unit) training which cuts down training time and allows for bigger models to be trained by allocating half of the neurons of the model on one GPU and the others on another GPU [51]. AlexNet employed two methods (Data augmentation and Dropout) to reduce overfitting because of its 60 million parameters. For data augmentation, label preserving transformation was used to make data more varied and principal component analysis was performed on RGB images to replace the RGB channels intensities. This caused more than 1% reduction in top-1 error rate. The dropout technique involved switching neurons off with a probability that was predetermined. This model won the ImageNet 2021 competition having a 15.35 top-5 error rate [51]. Figure 27 graphically illustrates the AlexNet architecture.

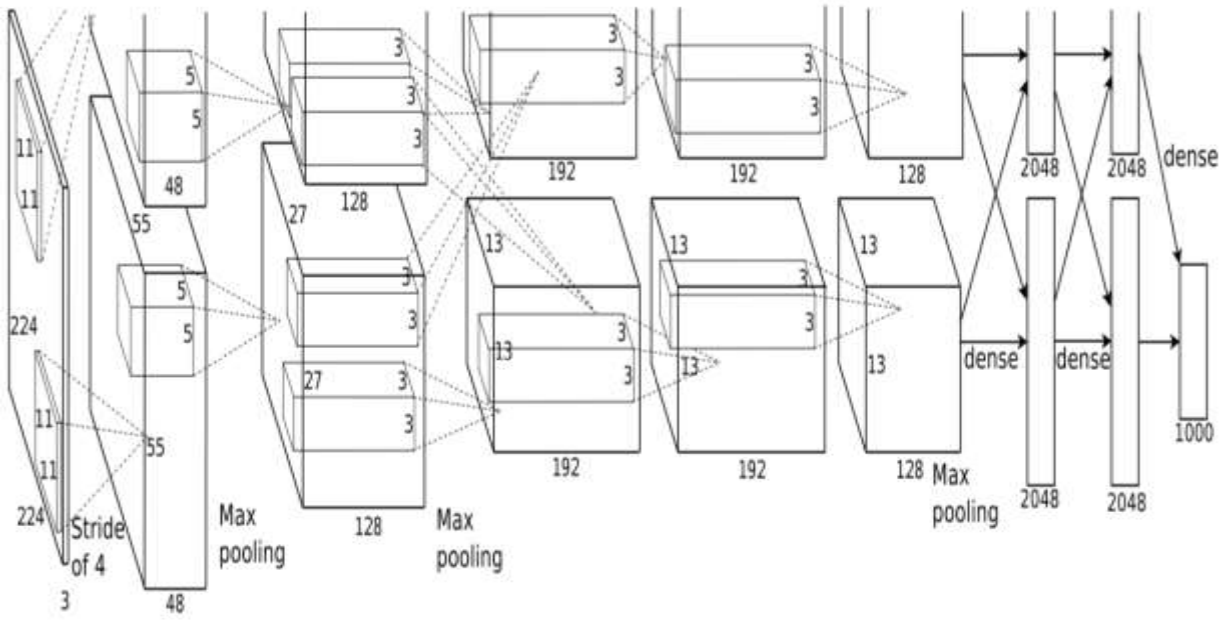


Figure 22: AlexNet Structure [52].

- **Xception:** The Xception network is a type of CNN architecture that relies on two major points: Depth wise separable convolutions (DSC) and shortcuts between convolution blocks. DSCs are substitutes to classical convolutions which are assumed to be better with regards to computation time. Convolution is an expensive operation and DSC is used to overcome this [53]. The name Xception stands for Extreme Inception. The network has 36 CNN layers that form the feature extraction base and these layers are arranged into 14 modules which all have linear residual connections surrounding them apart from the beginning and end modules [54]. The Xception network is a linear stack that has residual connections which makes it easy to define and modify the network. Figure 28 graphically illustrates the Xception architecture.

Table 6 summarizes and compares the different architectures explained above. In the table, Top-1-Accuracy and Top-5-Accuracy refer to the performance of the models on the ImageNet validation dataset while depth refers to the topological depth of the layer which includes the batch normalization, activation layers and other layers.

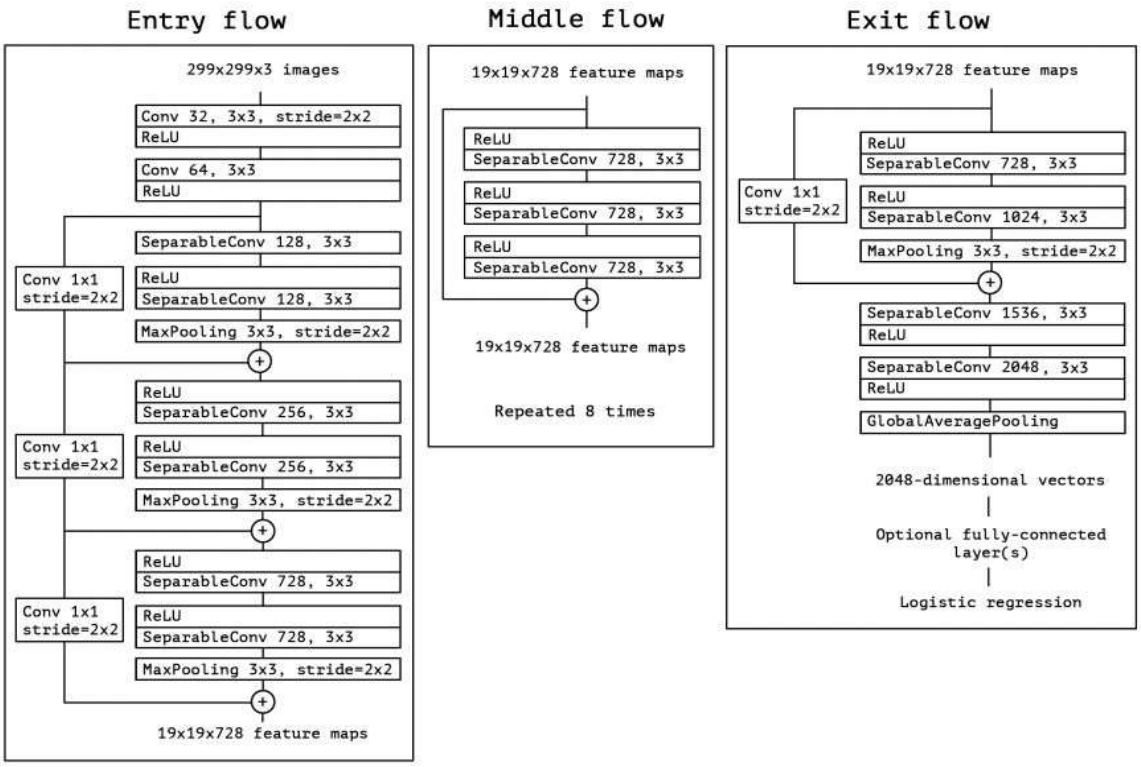


Figure 23: Xception Architecture [54].

Table 5: Summary of some types of CNN Architectures (Das, 2017).

Model	**Size**	**Top-1-Accuracy**	**Top-5-Accuracy**	**Parameters**	**Depth**
VGG-19	549 MB	0.713	0.901	143,667,270	26
GoogleNet	92 MB	0.779	0.937	23,851,784	159
ResNet-101	171 MB	0.764	0.928	44,707,176	-
Xception	88 MB	0.7909	0.945	22,910,480	126
AlexNet	215 MB	0.803	0.953	60,000,000	572

4.6.2 Image Training and Validation

70% of the input images were allocated for training while the remaining 30% of the input data was allocated for validation. The softmax activation function was used for classification. Classification was done in two phases: 4 Classes (Normal, Mild, Moderate & Severe) and 2 Classes: Normal & Abnormal (mild, moderate & severe)

4.7 Evaluation Metrics

Evaluation metrics is used to quantify a predictive model's performance *[55]*. In classification, metrics compares the label for the predicted class with the label for the expected class.

Contingency Table: The contingency table is the heart of all scores used in binary classification performance. The table showcases the number of instances that satisfy certain conditions [56] as shown in Table 7.

Table 6: Contingency Table (The Learning Machine, 2021).

	Actually positive: y=1	**Actually negative: y=−1**
Predicted positive: $\hat{y} = \mathbf{1}$	True positive (TP)	False negative (FN)
Predicted negative: $\hat{y} = \mathbf{1}$	False positive (FP)	True negative (TN)

- **True Positive**: This is the number of positive samples that were classified positive by the model. This is the number of samples where:

$$\hat{y}l = 1 \; and \;\; yl = 1 \qquad Eqn\ 10$$

- **True Negatives:** This is the number of negatives samples which were classified negative by the network. This is the number of samples where:

$$\hat{y}l = \; -1 \; \& \; yl = -1 \qquad Eqn\ 11$$

- **False Negatives:** This is the number of negative samples that were not correctly classified as negative. This is the number of samples where:

$$\hat{y}l = \; -1 \; \& \; yl = 1 \qquad Eqn\ 12$$

- **False Positive:** This is the number of positive samples that were not correctly classified as positive. This is the number of samples where:

$$\hat{y}l = \; 1 \; \& \; yl = -1 \qquad Eqn\ 13$$

Accuracy: Accuracy is the number of predictions that are correct. It is the number of true positive and true negatives over the whole population. Accuracy can be used to evaluate binary and multiclass classification but in cases where the classes are imbalanced, multi-labelled class scenarios, or anomaly detection classification scenarios, accuracy is not suitable.

$$Accuracy = \frac{Number\ of\ Correct\ Predcitions}{P+N} \qquad Eqn\ 14$$

$$Accuracy = \frac{TP+TN}{TP+FP+FN+TN} \qquad Eqn\ 15$$

Recall: Recall or Sensitivity is the measurement of samples that were correctly classified to be positives over the number of samples that are positive. It computes the sensitivity of the model in identifying abnormalities in samples.

$$Recall = \frac{TP}{TP+FN} \qquad Eqn\ 16$$

Precision: Precision is the number of TP against the total number of positives. It is the measure of how precisely the network identifies the positive samples by avoiding wrong classification of the negative samples as positive [56].

$$Precision = \frac{TP}{TP+FP} \qquad Eqn\ 17$$

Specificity: Specificity or true negative rate is the number of TN against the total number of negatives. It is the measure of how precisely the network identifies the negative samples by avoiding wrong classification of the positive samples as negative [56].

$$Specificity = \frac{TN}{TN+FP} \qquad Eqn\ 18$$

F1 Score: F1 score is the aggregate score between the Recall and Precision scores. It can be computed as the harmonic mean of precision and recall.

$$F1 = \left(\frac{Recall+Precision}{2}\right)^{-1} \qquad Eqn\ 19$$

$$= \frac{2}{\frac{1}{Recall}+\frac{1}{Precision}} \qquad Eqn\ 20$$

CHAPTER 5
RESULTS AND DISCUSSION

5.1 Introduction

The results obtained in this study for the 2 and 4 Classification task using 5 CNN models are presented and discussed in this chapter.

Figure 24: Key for understanding results

Fig 24 gives the key for understanding the results shown below. On the graphs shown for each model, the blue line represents the smoothed training accuracy, the black lines represent the validation accuracy in the first graph and the validation loss in the second graph. The red line signifies the training loss. In the first graph, training accuracy is plotted against validation loss and in the second graph, training loss is plotted against validation loss. The accuracy and loss are represented on the vertical axis for the first and second graph of each model respectively. The number of iteration and epoch is represented on the horizontal axis for both graphs for each model. The training accuracy is the classification accuracy computed on each individual mini-batch. The smoothed training accuracy is computed by applying a smoothing algorithm to the training accuracy. It makes it easier to spots trends as it is less noisy than the unsmoothed accuracy. Validation accuracy is the Classification accuracy on the entire validation set. Training loss signifies the loss on each mini-batch. The smoothed training loss represents the smoothed version of the training loss, and validation loss represents the loss on the validation set.

The confusion matrix shows the number of images successfully and unsuccessfully classified by the model. The blue boxes represent the number of validation images classified correctly and the light red boxes represent images wrongly classified in each HPR stage by the model. The confusion matrix was used to calculate the accuracy, precision, specificity and sensitivity scores.

5.2 Classification into Four Classes (Mild HP, Moderate HP, No Visible HP, Severe HP)

Figures 29 through 38 present classification results using the five network architectures discussed in the previous chapter including the confusion matrices for the case of four classes.

5.2.1 ResNet-101

Figure 25: Validation Accuracy and Loss Graph ResNet-101 (4 Classes).

Resnet-101 has given a validation accuracy of 91.61%, precision of 90.43%, sensitivity of 87.31% and a specificity of 95.57% for the 4-classification task. It took about 182 minutes for training and validation loss started from 1.6 and reduced to 0.2 by the end of the final iteration. Resnet-101 had the second highest accuracy when compared to the other models implemented. The confusion matrix is shown below in Fig. 30.

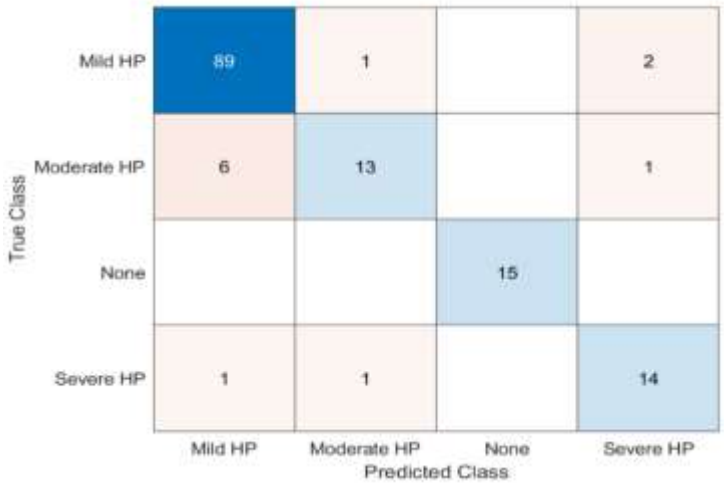


Figure 26: Confusion Matrix ResNet-101101 (4 classes).

5.2.2 GoogleNet

Figure 27: Validation Accuracy and Loss Graph GoogleNet (4 Classes).

GoogleNet has given a validation accuracy of 93.01%, precision of 93.30%, sensitivity of 89.15% and a specificity of 95.96% for the 4-Classification task. It took about 39 minutes for training and validation loss started from 4.99 and reduced to 0.01 by the end of the final iteration. The model

achieved the highest accuracy for the 4-classification task. The confusion matrix is shown below in Fig. 32.

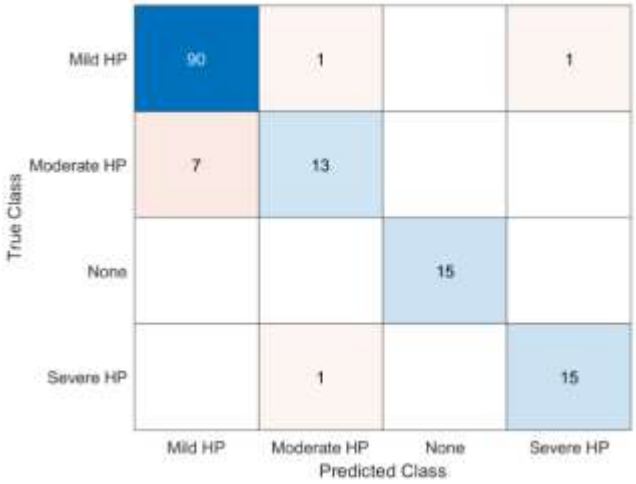


Figure 28: Confusion Matrix for GoogleNet classification results (4 Classes).

5.2.3 AlexNet

Figure 29: Validation Accuracy and Loss Graph AlexNet (4 Classes).

AlexNet has given a validation accuracy of 87.41%, precision of 84.92%, sensitivity of 87.95% and a specificity of 94.64% for the 4-Classification task. It took about 21 minutes to train and validate and it was the fastest when compared with the other models used. Validation loss started from 5.4 and reduced to 0.04 by the end of the final iteration. It achieved the fourth position in terms of validation accuracy when compared to the other models used. The confusion matrix is shown below in Fig. 34.

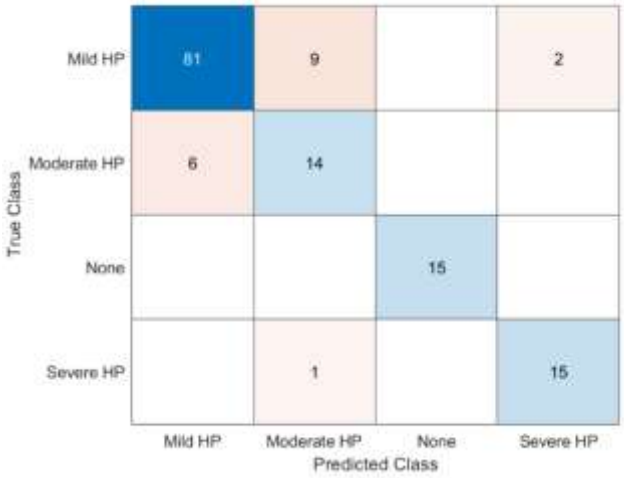


Figure 30: Confusion Matrix for AlexNet classification results (4 Classes).

5.2.4 VGG-19

Figure 31: Validation Accuracy and Loss Graph VGG-19 (4 Classes).

VGG-19 has given a validation accuracy of 90.21%, precision of 88.78%, sensitivity of 86.66% and a specificity of 94.87% for the 4-Classification task. It took about 370 minutes for training and validation loss started from 2.7 and reduced to 0.28 by the end of the final iteration. It obtained the third highest accuracy out of the 5 models implemented. The confusion matrix is shown below in Fig. 36.

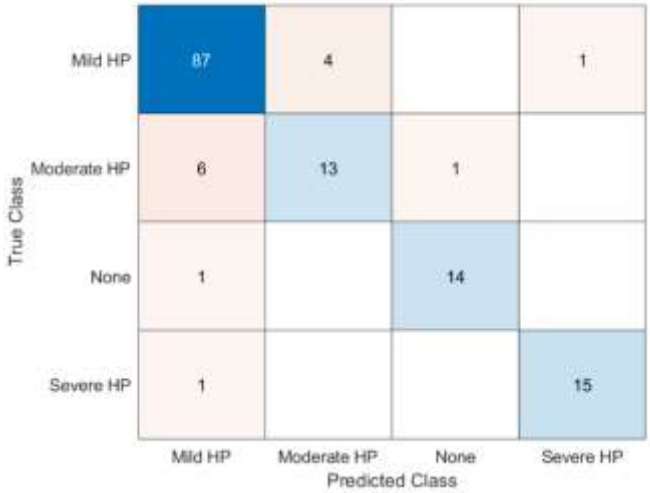


Figure 32: Confusion Matrix for VGG-19 classification results (4 Classes).

5.2.5 Xception

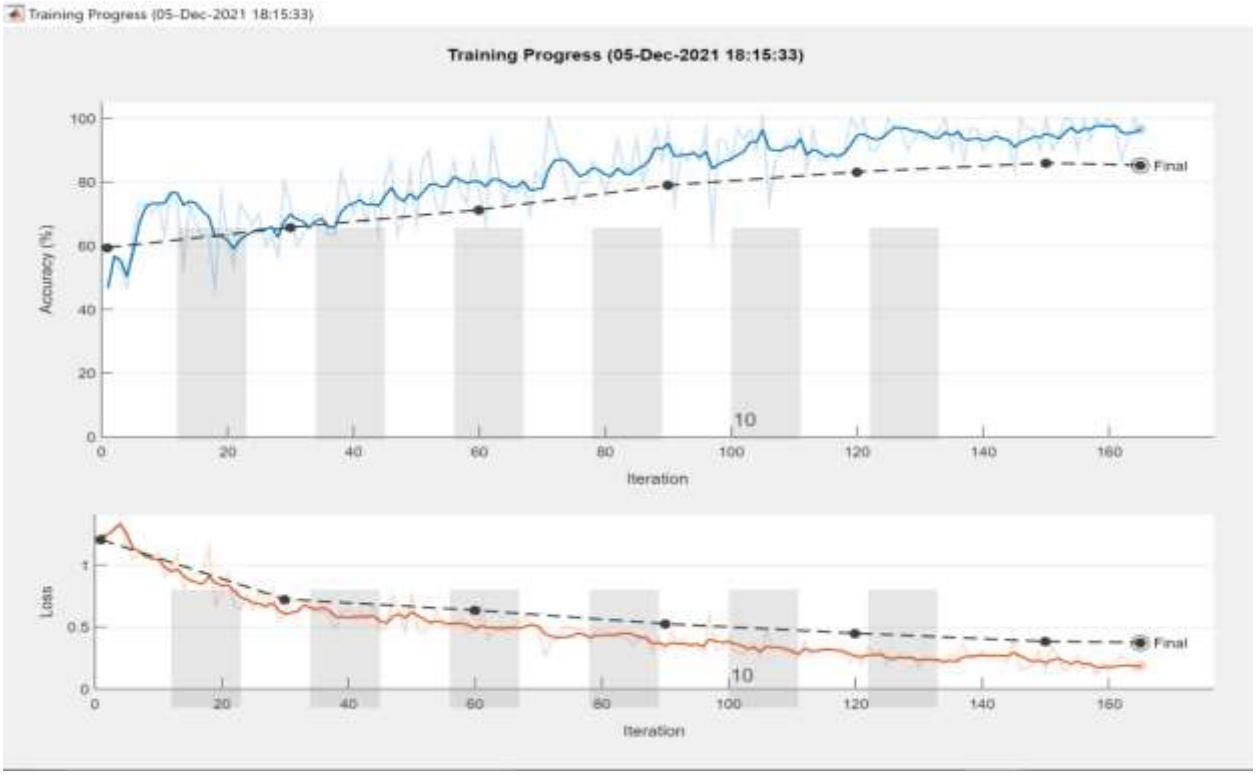


Figure 33: Validation Accuracy and Loss Graph Xception (4 Classes).

Xception has given a validation accuracy of 85.31%, precision of 85.30%, sensitivity of 88.22% and a specificity of 92.87% in the 4-Classification task. It took about 275 minutes for training and validation loss started from 8.0 and reduced to 1.38 by the end of the final iteration. This model had the lowest accuracy when compared to the 5 models implemented. The confusion matrix is shown below in Fig. 38.

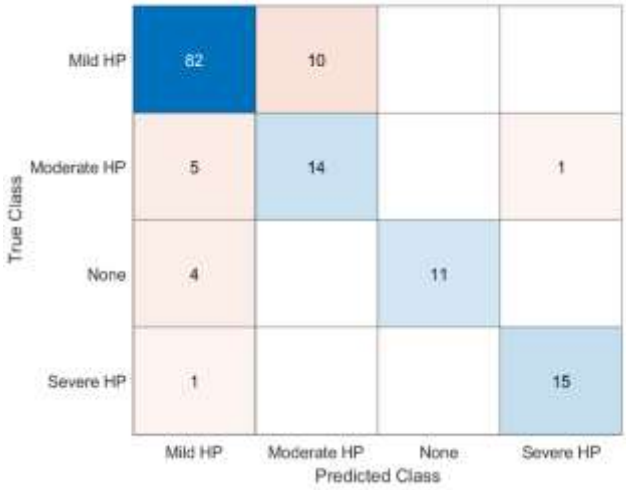


Figure 34: Confusion Matrix for Xception classification results (4 Classes).

5.3 Classification into Two Classes (Normal, Abnormal)

The 2-Classification task was implemented to evaluate how well the model performs in classifying between normal and abnormal. Figures 39 through 48 present classification results using the five network architectures discussed in the previous chapter including the confusion matrices for the case of two classes.

5.3.1 ResNet-101

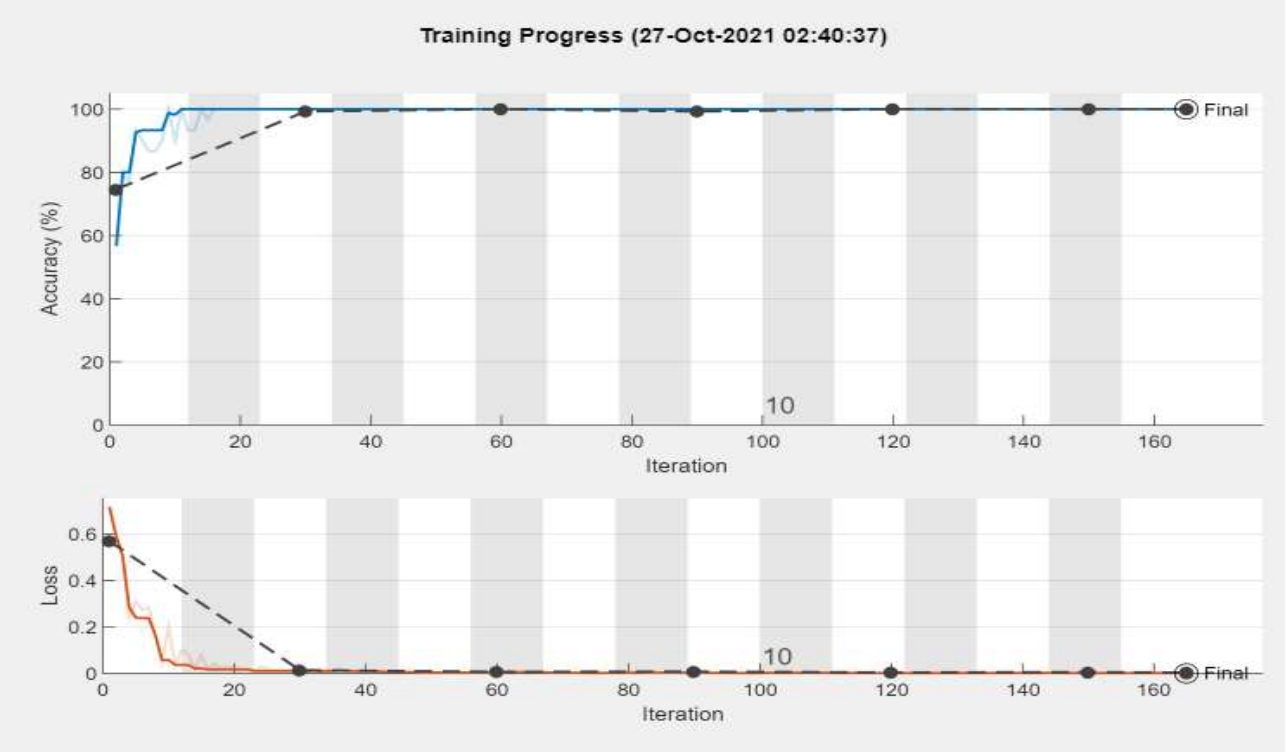


Figure 35: Validation Accuracy and Loss Graph ResNet-101 (2 Classes).

Resnet-101 has given a validation accuracy of 100%, precision of 100%, sensitivity of 100% and a specificity of 100% for the 2-Classification task. It took about 258 minutes for training and validation loss started from 0.712 and reduced to 0.002 by the end of the final iteration. The confusion matrix is shown below in Fig. 40.

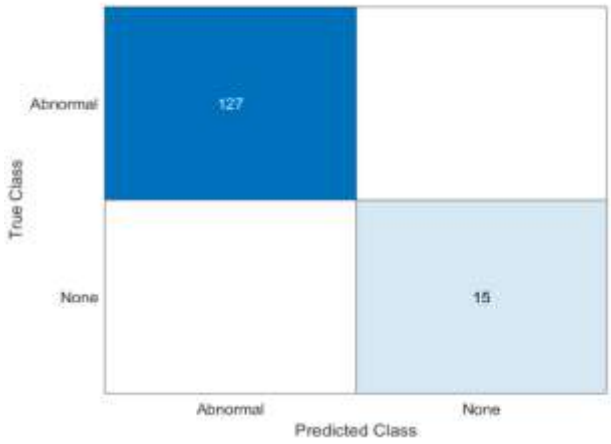


Figure 36: Confusion Matrix for ResNet-101 classification results (2 Classes).

5.3.2 GoogleNet

Figure 37: Validation Accuracy and Loss Graph GoogleNet (2 Classes).

GoogleNet has given a validation accuracy of 100%, precision of 100%, sensitivity of 100% and a specificity of 100% for the 2-Classification task. It took about 39 minutes as training time. Validation loss started from 1.67 and reduced to 0.003 by the end of the final iteration. The confusion matrix is shown below in Fig. 42.

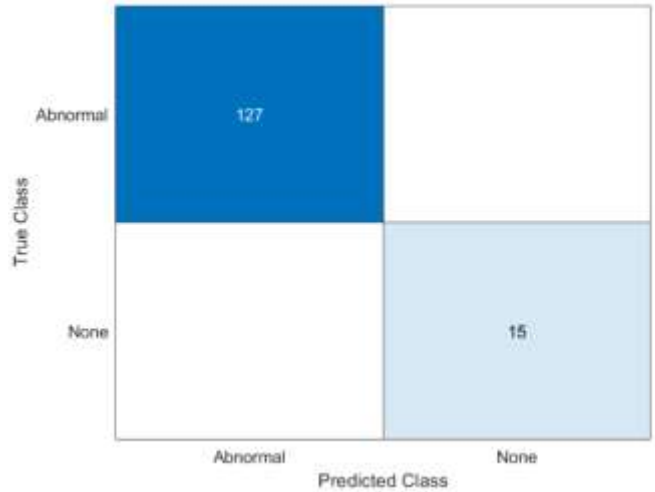


Figure 38: Confusion Matrix for GoogleNet classification results (2 Classes).

5.3.3 AlexNet

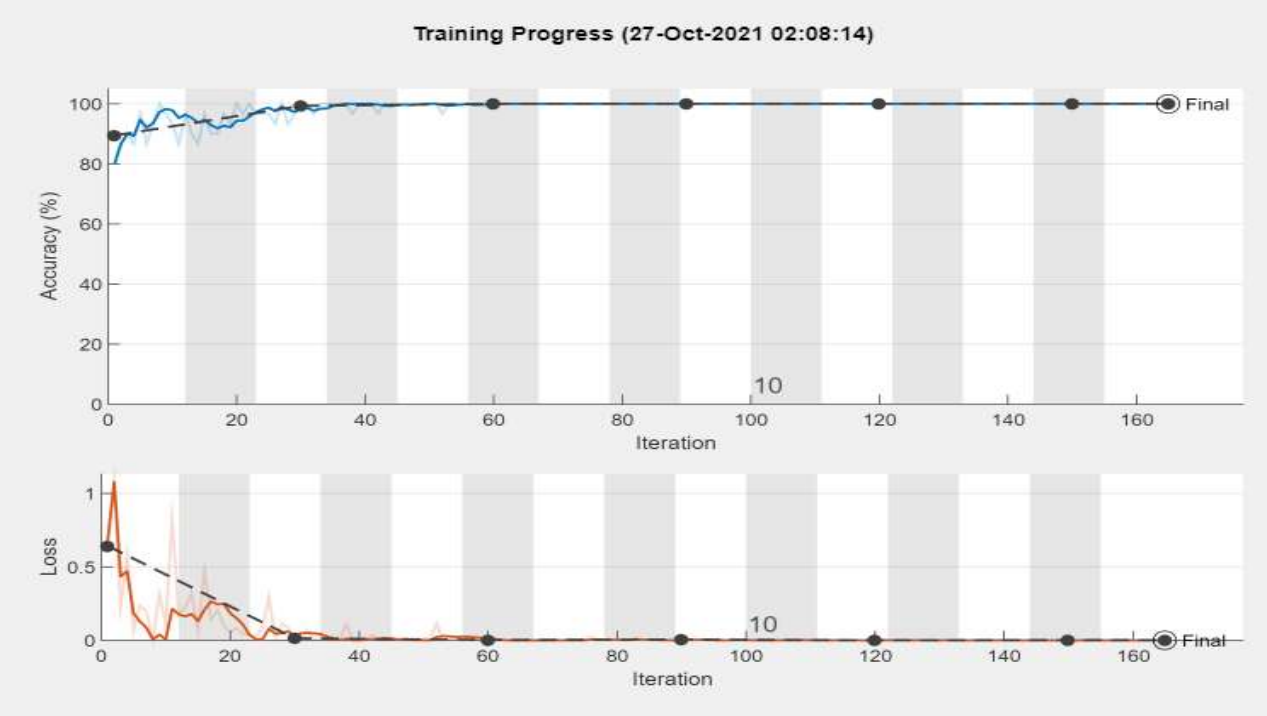


Figure 39: Validation Accuracy and Loss Graph AlexNet (2 Classes).

AlexNet has given a validation accuracy of 100%, precision of 100%, sensitivity of 100% and a specificity of 100% for the 2-Classification task. It took about 25 minutes for training. Validation loss started from 0.67 and reduced to 0.002 by the end of the final iteration. The confusion matrix is shown below in Fig. 44.

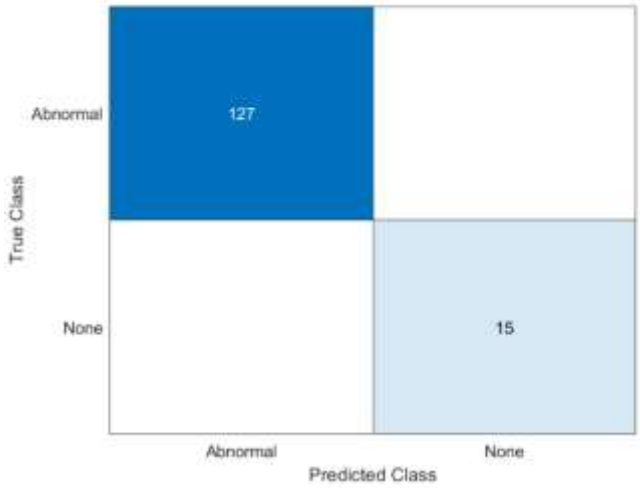


Figure 40: Confusion Matrix for AlexNet classification results (2 Classes).

5.3.4 VGG-19

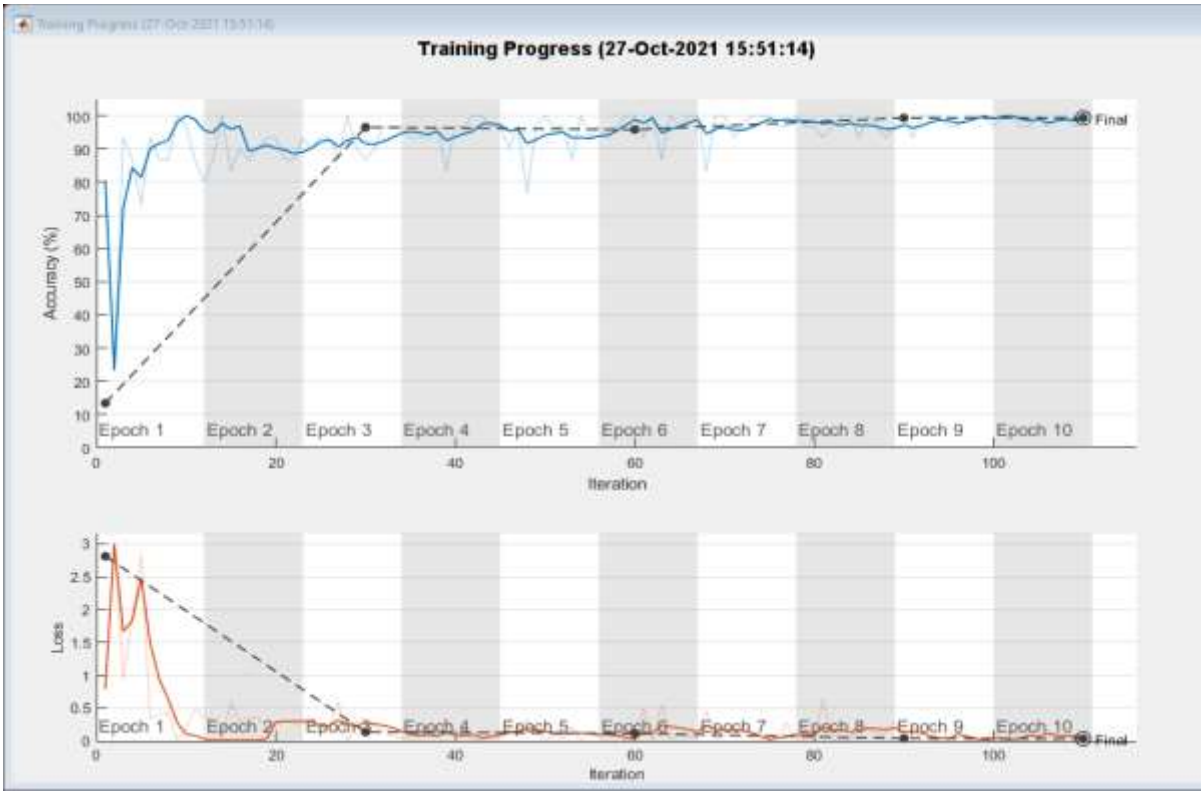


Figure 41: Validation Accuracy and Loss Graph VGG-19 (2 classes).

VGG-19 has given a validation accuracy of 99.30%, precision of 99.61%, sensitivity of 196.67% and a specificity of 96.67%. It took about 108 minutes for training. The confusion matrix is shown below in Fig. 46.

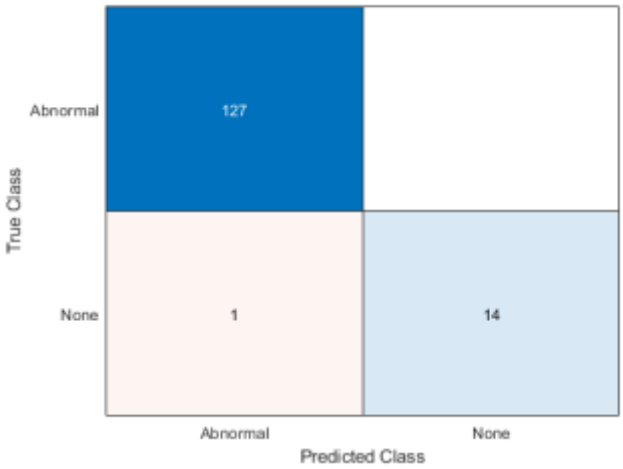


Figure 42: Confusion Matrix for VGG-19 classification results (2 Classes).

5.3.5 Xception

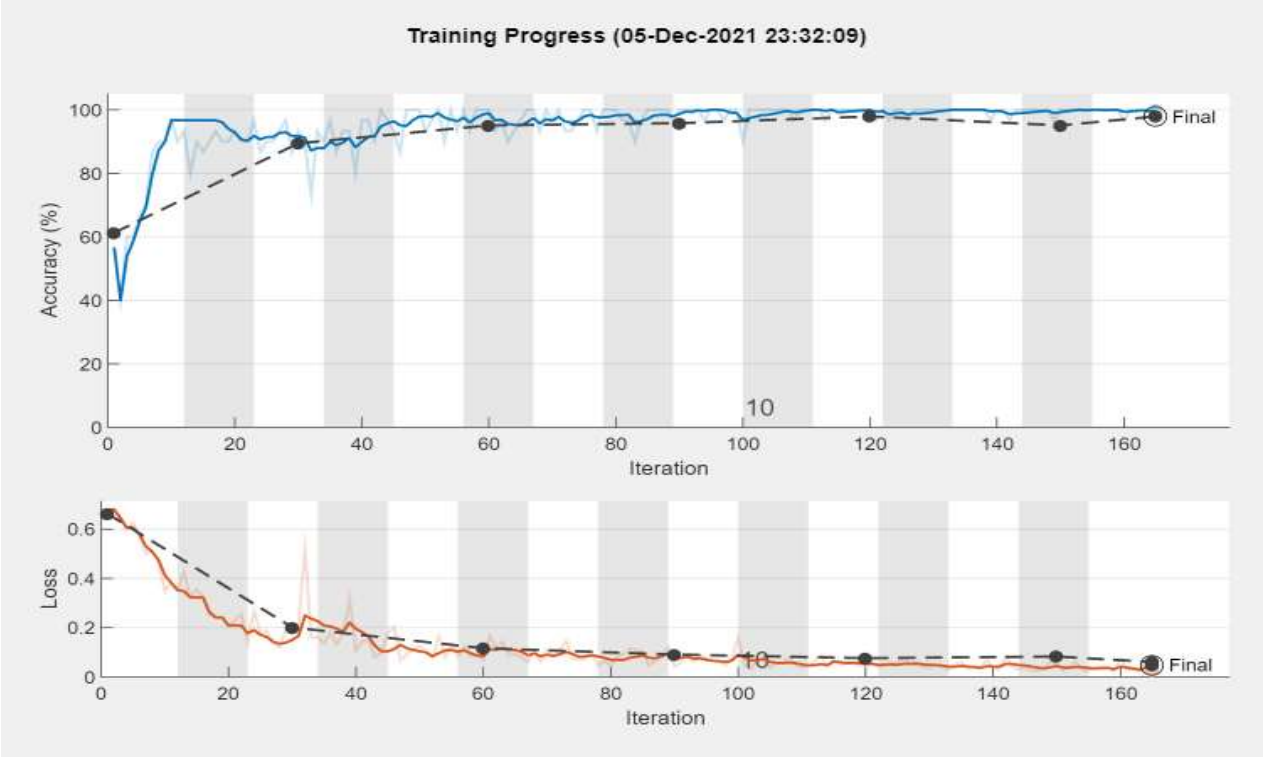


Figure 43: Validation Accuracy and Loss Graph Xception (2 Classes).

Xception has given a validation accuracy of 97.89%, precision of 98.85%, sensitivity of 90% and a specificity of 90%. It took about 229 minutes for training and validation loss started from 0.67 and reduced to 0.25 by the end of the final iteration. The confusion matrix is shown below in Fig. 48

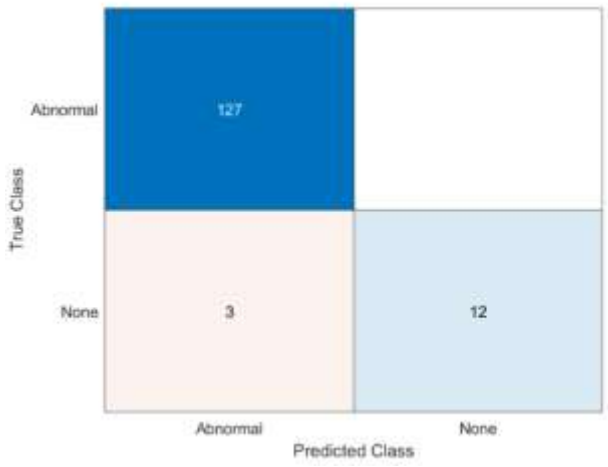


Figure 44: Confusion Matrix for Xception classification results (2 Classes).

5.4 General Comparison between Architectures (4 Classes)

Table 8 compares the performance of the five network architectures in terms of validation accuracy, precision, sensitivity and specificity for the case of four classes while Fig. 49 presents the results in bar plots. The comparisons for the 2 class case is presented in Table 9 with the corresponding bar plot in Fig. 50.

Table 7: Evaluation metrics (4 classes).

Networks	Validation Accuracy %	Precision %	Sensitivity %	Specificity %	Training Time (min)	Computational time (millisecs)
ResNet-101	91.61	90.43	87.31	95.57	183	264.56
GoogleNet	93.01	93.3	89.15	95.96	39	204.89
AlexNet	87.41	84.92	87.95	94.64	21	323.20
VGG-19	90.21	88.78	86.66	94.87	370	255.91
XCeption	85.31	85.30	88.22	92.87	275	281.95

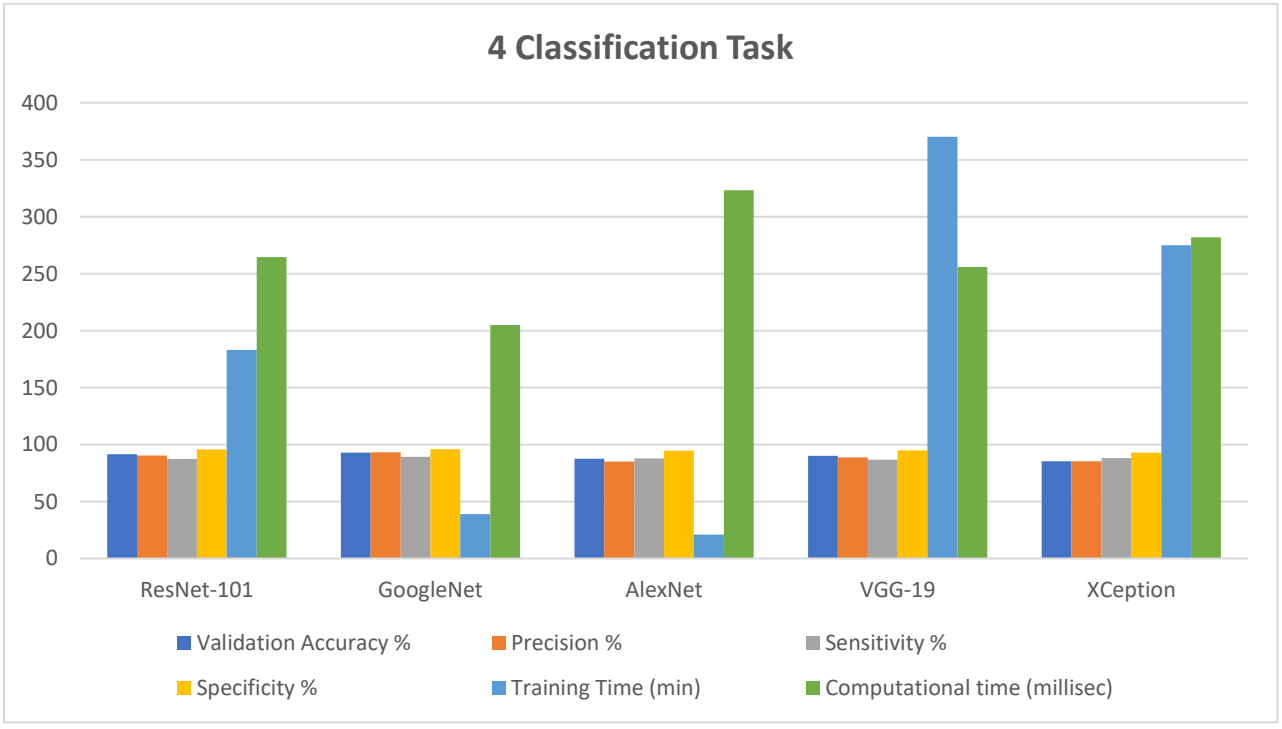


Figure 45: plot representation showing evaluation metrics (4 classes).

The Bar Graph gives a representation of the comparison between the evaluation metrics for the pretrained models used in the 4 classification task.

5.5 General Comparison between Architectures (2 Classes)

Table : Evaluation metrics (2 classes)

Networks	Validation Accuracy %	Precision %	Sensitivity %	Specificity %	Training Time (min)	Computational time (millisecs)
ResNet-101	100	100	100	100	258	318.16
GoogleNet	100	100	100	100	39	196.38
AlexNet	100	100	100	100	25	226.55
VGG-19	99.30	99.61	96.67	96.67	108	277.17
Xception	97.89	98.85	90	90	229	367.55

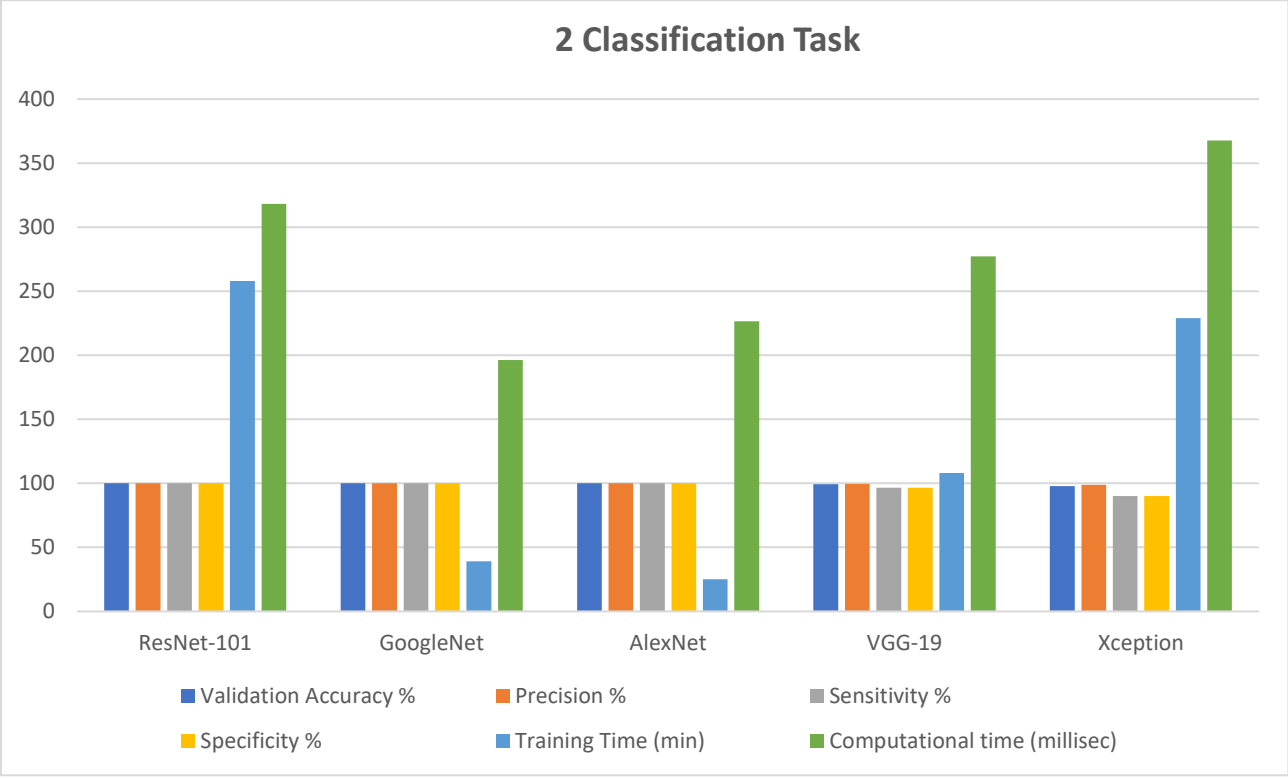


Figure 46: Bar plot representation of evaluation metrics (2 classes).

The bar plot gives a representation of the comparison between the evaluation metrics for the pretrained models used in the 2-classification task.

5.6 Discussion

In the case of 4-class classification, GoogleNet achieved the highest accuracy at 93.01% when compared with the other pretrained models used. GoogleNet had a training time of 39 minutes and the minimum computational time of 204.89 milliseconds. ResNet-101 achieved the next highest accuracy at 91.61% with a training time of 183 minutes and computational time of 264.56 milliseconds. Xception with an accuracy of 85.31% had a training time of 275 minutes and a computational time of 281.95 milliseconds. VGG-19 was the third highest with an accuracy of 90.21% but had the longest training time and a computational time of 255.91 milliseconds. AlexNet had an accuracy of 87.41% though it had the minimum training time and maximum computational time of 323.20 milliseconds. As compared with the study reported in [6] where the authors achieved an accuracy of 92.6%, specificity of 91.5% and sensitivity of 90.5% in the 5-class classification task of HPR, GoogleNet had an accuracy of 93.01%, specificity of 95.96% and a sensitivity of 89.15% in the 4-class classification task of HPR implemented in the current study.

For the 2-class classification, ResNet-101, GoogleNet and AlexNet all offered perfect performance at 100% with a computational score of 318.16, 196.38, 226.55 milliseconds respectively. VGG-19 and Xception also perform well with an accuracy of 99.30% and 97.89% and a computational time of 277.17 and 367.55 milliseconds respectively. When comparing for both classifications (2 classes and 4 classes), GoogleNet excels above other models with regards to computational time and validation accuracy as well. The results of the 2-classification task when compared to [40] [41] [42] [43] [45] has given better accuracy.

Two important performance metrics in health care dataset are sensitivity and specificity. Sensitivity tells how well the model is able to correctly identify that a person has a particular disease. Specificity is the ability of the model to classify persons without HPR lesions as negative. This is particularly important because it is critical for the model to identify a person who has the disease as positive for the disease and a person who doesn’t have the disease as negative for the disease. The 5 models used in the current study offered an average accuracy of 88.51% and the models offered an average of 87.86% in sensitivity in the case of 4-class classification. Overall, the models have a high specificity

and sensitivity value and this shows that they are able to correctly identify images with and without pathology features. The models also give good precision and accuracy scores. The Matlab implementation of each network is presented in the Appendix section.

The current study considers only three levels of HPR: Mild, moderate and severe which most literatures agree is simplified for clinical practice. While there is a recent approach that considered more severity types. For example, the study in [6] considered four levels of severity. It is, however, still considered easier for ophthalmologists to classify HPR into mild, moderate and severe than using other forms of classification. The number of research carried in literatures with an attempt to classify HPR into multiple classes based on the severity is still very low. The classification scheme proposed in the current study is just only one of the very few studies already reported in the literature.

Only datasets that contain HPR cases have been considered in the current study. There are other studies reported in the literature and considered other types of retinopathies such as diabetic retinopathy. It will still be interesting to check the performance of the proposed scheme in correctly classifying other retinopathies other than HPR including diabetic retinopathy, age related macular degeneration, retinopathy of prematurity and the like which is beyond the scope of the current study. One would doubt if a robust classification scheme could be developed that considers classifying different types of retinopathies into stages.

The current study considered and tested the efficacy of other architectures other than the 5 CNN architectures mentioned in the methodology as well as the results section. Architectures including SqueezeNet, Inception-v3 and the like offered inferior results to the results obtained using the 5 architectures adopted in the current study, even though the results are not documented here.

Only datasets that contain HPR cases have been considered in the current study. There are other studies reported in the literature and considered other types of retinopathies such as diabetic retinopathy. But such issues are beyond the scope of the current study.

CHAPTER 6

CONCLUSION AND RECOMMENDATION

6. 1 Conclusion

In this thesis work, we have employed transfer learning in classifying our dataset. Our results overall are acceptable. The pretrained models show that with proper tuning of training parameters like the learning rate, number of epochs. Batch size, type of optimizer, activation and loss function, they can be used in HPR classification.

6.2 Limitation

Some of the limitations are listed below:

- There was a limitation on the size of data acquired. Usually for deep learning, the larger the data size the better. For our case, our data size was quite small.
- We also encountered the issue of unbalanced data size as the amount of data for each class was disproportionate. Ideally each class size should be the same for better classification.

6.3 Future work

One possible future work would be the application of deep learning in classifying HPR and diabetic retinopathy (DR) into stages as they both have similar pathological features. This would be really useful for eye specialists in the healthcare facility especially in differentiating between HPR and DR because of similar lesions they share. Another future work would be to increase the HPR database for better classification results. That way the performance of the proposed classification scheme could be checked on a large size database. As a recommendation, the effect of resistance to noise on the performance metric of the models used in this study needs to be evaluated.

References

[1] CDC, “High Blood Pressure,” 22 October 2020. [Online]. Available: https://www.cdc.gov/bloodpressure/index.htm#:~:text=High%20blood%20pressure%20increases%20the,not%20have%20it%20under%20control..

[2] WHO, “Hypertension,” 13 September 2019. [Online]. Available: https://www.healio.com/news/cardiology/20131010/ct0813ikram_10_3928_1081_597x_20130801_09_1305083.

[3] P. Kearney, M. Whelton, K. Reynolds, P. Muntner, P. Whelton and J. He, “Global burden of Hypertension: analysis of worldwide data,” *Lancet,* pp. 217-223, 2005.

[4] T. Y. Wong and P. Mitchell, “Hypertensive Retinopathy,” *The new england journal of medicine,* vol. 351, pp. 2310-2317, 2004.

[5] O. Yi-Ting, Y. W. Tien, K. Ronald, E. K. Barbara, l. M. Pau, A. R. Sharrett, D. J. Couper and M. K. Ikram, “Hypertensive Retinopathy and Risk of Stroke,” *AHA Journals,* pp. 706-711, 2013.

[6] Q. Abbas, I. Qureshi and M. E. A. Ibrahim, “An Automatic Detection and Classification System of Five,” *MDPI,* vol. 6936, no. https://doi.org/10.3390/s21206936, p. 21, 2021.

[7] K. Takayama, H. Kaneko, Y. Ito, K. Kataoka, T. Iwase, T. Yasuma, T. Matsuura, T. Tsunekawa, H. Shimizu, A. Suzumura, E. Ra, T. Akahori and H. Terasaki, “Novel Classification of Early-stage Systemic Hypertensive Changes in Human Retina Based on OCTA Measurement of Choriocapillaris,” *Science report,* vol. 8, no. https://doi.org/10.1038/s41598-018-33580-y, p. 15163, 2018.

[8] E. K. Aissopou, M. Papathanassiou, E. G. Nasothimiou, G. D. Konstantonis, N. Tentolouris, P. G. Theodossiadis, T. G. Papaioannou, P. P. Sfikakis and A. D. Protogerou, “The Keith-Wagener-Barker and Mitchell-Wong grading systems for hypertensive retinopathy: association with target organ damage in individuals below 55 years,” *National Library of Medicine,* Vols. (11):2303-9, no. doi: 10.1097/HJH.0000000000000702., p. Nov 33, 2015.

[9] Nvision, “ANATOMY OF THE EYE: EYE STRUCTURE & MORE,” 2020. [Online]. Available: https://www.nvisioncenters.com/education/eye-structure-anatomy/.

[10] M. Navid and K. Yasir Al, “Neuroanatomy, Retina,” 11 August 2021. [Online]. Available: https://www.ncbi.nlm.nih.gov/books/NBK545310/.

[11] EXCIMER, “Structure of the retina,” [Online]. Available: https://en.excimerclinic.ru/retina/structure/.

[12] G. Chaine and E. Kohner, “ La rétinopathie hypertensive [Hypertensive retinopathy],” *J Fr Ophtalmol,* pp. 995-1005, 1983.

[13] P. Modi and T. Arsiwalla, “Hypertensive Retinopathy,” Jan 2020. [Online]. Available: https://www.ncbi.nlm.nih.gov/books/NBK525980/.

[14] A. Grosso, F. Veglio, M. Porta, F. Grignolo and T. Wong, “Hypertensive retinopathy revisited: some answers, more questions,” *Br J Ophthalmol,* p. 89:1646–1654., 2005.

[15] A. S. Jalal, “A survey on automated microaneurysm detection in Diabetic Retinopathy retinal images,” Mar 2013. [Online]. Available: https://www.researchgate.net/figure/Normal-Retinal-Fundus-Image_fig1_261037783.

[16] Hipertency, “Fundoscopy Hypertension Grade,” 10 June 2016. [Online]. Available: https://all-about-hipertency.blogspot.com/2016/06/fundoscopy-hypertension-grade.html.

[17] P. J. Saine and M. E. Tyler, “Fundus Photography Overview,” in *Ophthalmic Photography: Retinal Photography, Angiography, and Electronic Imaging*, Butterworth-Heinemann Medical, p. ISBN: 0750673729.

[18] C. R. Baumal, “Imaging in Diabetic Retinopathy,” in *Current Management of Diabetic Retinopathy*, Boston, Elsevier, 2018.

[19] B. Maryam, H. Muhammad and F. Anam, “Application of deep learning for retinal image analysis: A review,” *Elsevier,* 2019.

[20] C. Lamirel, “Optical Coherence Tomography,” Elsevier Ltd, 2014.

[21] A. Carpenter, “What Is Deep Learning?,” Code Academy, 28 September 2021. [Online]. Available: https://www.codecademy.com/resources/blog/what-is-deep-learning/. [Accessed 24 January 2022].

[22] B. Ekaba, “Building machine learning and deep learning models on google cloud platform,” in *Building machine learning and deep learning models on google cloud platform*, New York, Apress Media, LLC , 2019, pp. 423-441.

[23] S. Sumit, “A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way,” 18 December 2018. [Online]. Available: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53.

[24] IBM Cloud Education, “Convolutional Neural Networks,” [Online]. Available: https://www.ibm.com/cloud/learn/convolutional-neural-networks.

[25] J. Brownlee, “A Gentle Introduction to Transfer Learning for Deep Learning,” 16 September 2019. [Online]. Available: https://machinelearningmastery.com/transfer-learning-for-deep-learning/.

[26] N. Donges, “What Is Transfer Learning? Exploring the Popular Deep Learning Approach,” 16 June 2019. [Online]. Available: https://builtin.com/data-science/transfer-learning.

[27] O. G. Yalçın, “4 Pre-Trained CNN Models to Use for Computer Vision with Transfer Learningg,” 23 September 2020. [Online]. Available: https://towardsdatascience.com/4-pre-trained-cnn-models-to-use-for-computer-vision-with-transfer-learning-885cb1b2dfc.

[28] KUFE, “Artificial Neural Network (ANN),” 2010. [Online]. Available: http://www.cs.kumamoto-u.ac.jp/epslab/ICinPS/Lecture-2.pdf.

[29] S. Doshi, “Various Optimization Algorithms For Training Neural Network,” 13 January 2019. [Online]. Available: https://towardsdatascience.com/optimizers-for-training-neural-network-59450d71caf6.

[30] J. Brownlee, “How to Choose an Activation Function for Deep Learning,” 22 January 2021. [Online]. Available: https://machinelearningmastery.com/choose-an-activation-function-for-deep-learning/.

[31] S. Saxena, “Introduction to Batch Normalization,” 9 March 2021. [Online]. Available: https://www.analyticsvidhya.com/blog/2021/03/introduction-to-batch-normalization/.

[32] A. Gandhi, “Data Augmentation | How to use Deep Learning when you have Limited Data — Part 2,” Nanonets, July 2021. [Online]. Available: https://nanonets.com/blog/data-augmentation-how-to-use-deep-learning-when-you-have-limited-data-part-2/. [Accessed October 2021].

[33] A. C. d. M. Lima, L. B. Maia, P. T. C. d. Santos, G. B. Junior, J. D. S. d. Almeida and A. C. d. Paiva, “Evolving Convolutional Neural Networks for Glaucoma Diagnosis,” *Brazilian Journal of health Review,* pp. DOI:10.34119/bjhrv3n4-160, 2020.

[34] B. Y. BIZUNEH, “DETECTION AND GRADING OF TRACHOMA USING DEEP CONVOLUTIONAL NEURAL NETWORK,” Bahir Dar, Ethiopia, 2019.

[35] S. Kadam, V. Pradhan, K. Kate and J. M, “CONVOLUTION NEURAL NETWORK BASED METHOD FOR AUTOMATIC DETECTION OF DIABETIC EYE DISEASE USING THERMAL IMAGES,” *International Research Journal of Engineering and Technology (IRJET),* pp. Volume: 07 Issue: 05 Page 3048 - 3051, 2020.

[36] W. Xiaoliang, L. Yongjin, W. Yujuan and C. Wei-Bang, “Diabetic Retinopathy Stage Classification using Convolutional Neural Networks,” *IEEE,* pp. 465-471, 2018.

[37] N. M N and R. A Angel, “Diabetic Retinopathy Stage Classification using CNN,” *International Research Journal of Engineering and Technology (IRJET),* pp. 5969-5974, 2019.

[38] D. S. W. Ting, L. R. Pasquale, L. Peng, J. P. Campbell, A. Y. Lee, R. Raman, G. S. W. Tan, L. Schmetterer, P. A. Keane and T. Y. Wong, "Artificial intelligence and deep learning in ophthalmology," *Br J Ophthalmol,* p. 103:167–175., 2019.

[39] K. T. Bambang, B. Widodo and A. Edi, "The Classification of Hypertensive Retinopathy using convolutional neural network," in *Procedia Computer Science*, Bali, Indonesia, 2017.

[40] A. Rahmat and Basari, "Detection of hypertensive retinopathy using principal component analysis (PCA) and backpropagation neural network methods," in *3rd Biomedical Engineering's Recent Progress in Biomaterials, Drugs Development, and Medical Devices*, 2019.

[41] K. Noronha, N. K.T and K. P. Nayak, "Support System for the Automated Detection of," *International Journal of Computer Applications,* vol. (0975 – 8887), pp. 7 - 11, 2012.

[42] Q. Abbas and M. E. A. Ibrahim, "DenseHyper: An automatic recognition system for detection of hypertensive retinopathy using dense," *Springer Link,* vol. Multimed Tools Appl 79, no. https://doi.org/10.1007/s11042-020-09630-x, p. 31595–31623 , 2020.

[43] M. F. Syahputra, C. Amalia, R. F. Rahmat, D. Abdullaht, D. Napitupulu, M. I. Setiawan, Nurdin, W. Albra and U. Andayani, "Hypertensive retinopathy identification through retinal fundus image using backpropagation neural network," *Journal of Physics Conference Series,* vol. 978(1):012106, no. 10.1088/1742-6596/978/1/012106, 2018.

[44] G. Dai, W. He, L. Xu, E. E. Pazo, T. Lin, S. Liu and C. Zhang, "Exploring the effect of hypertension on retinal microvasculature using deep learning on East Asian population," *PLOS ONE,* pp. https://doi.org/10.1371/journal.pone.0230111: 1-13, 2020.

[45] A. Shahzad, a. usman, H. Taimur, Y. Ubaidullah and B. Imran, "Data on Fundus Images for Vessels Segmentation, Detection of Hypertensive Retinopathy, Diabetic Retinopathy and Papilledema," 27 March 2020. [Online]. Available: https://data.mendeley.com/datasets/3csr652p9y/2.

[46] S. G. Vázquez, N. B. B. Cancela, M. G. Penedo, M. Rodríguez-Blanco, M. Pena, G. C. d. Tuero, M. A. Barceló and M. Saez, "Improving retinal artery and vein classification by means of a minimal path approach"," *Machine Vision and Applications,* pp. 919-930, 2013.

[47] Y. Zheng, C. K. Yang and A. Merkulov, "Breast cancer screening using convolutional neural network and follow-up digital mammography," in *Computational Imaging III PROCEEDINGS VOLUME 10669*, Florida, 2018.

[48] Mathworks, "resnet101," 2021. [Online]. Available: https://www.mathworks.com/help/deeplearning/ref/resnet101.html.

[49] Y. Tong, W. Lu, Q.-q. Deng, C. Chen and Y. Shen, “Automated identification of retinopathy ofprematurity by image-based deep learning,” *Eye and Vision,* pp. 1-12, 2020.

[50] P. Jay, “Understanding and Implementing Architectures of ResNet and ResNeXt for state-of-the-art Image Classification: From Microsoft to Facebook,” 7 February 2018. [Online]. Available: https://medium.com/@14prakash/understanding-and-implementing-architectures-of-resnet-and-resnext-for-state-of-the-art-image-cf51669e1624.

[51] J. Wei, “AlexNet: The Architecture that Challenged CNNs,” 3 July 2019. [Online]. Available: https://towardsdatascience.com/alexnet-the-architecture-that-challenged-cnns-e406d5297951.

[52] I. Sutskever, A. Krizhevsky and G. E. Hinton, “ImageNet Classification with Deep Convolutional,” *Advances in Neural Information Processing Systems 25,* pp. https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf, 2012.

[53] M. Fabien, “XCeption Model and Depthwise Separable Convolutions,” 20 March 2019. [Online]. Available: https://maelfabien.github.io/deeplearning/xception/#.

[54] F. Chollet, “Xception: Deep Learning with Depthwise Separable Convolutions,” *IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* pp. 1251-1258, 2017.

[55] J. Brownlee, “Tour of Evaluation Metrics for Imbalanced Classification,” 8 January 2020. [Online]. Available: https://machinelearningmastery.com/tour-of-evaluation-metrics-for-imbalanced-classification/.

[56] The Learning Machine, “Evaluation metrics for classification: Machine Learning,” 2021. [Online]. Available: https://the-learning-machine.com/article/ml/evaluation-metrics-for-classification?gclid=Cj0KCQjwiNSLBhCPARIsAKNS4_fK_JQomfBxAJ8bxfCdo-27Xa-I0dwVtMMLwFOU4RS5Jk_ZaYyf6iYaAtwjEALw_wcB.

[57] R. Nall, “A guide to the spinal cord: Anatomy and injuries,” 12 November 2019. [Online]. Available: https://www.medicalnewstoday.com/articles/326984.

[58] Cleveland Clinic, “Spine Structure and Function,” 1 February 2015. [Online].

[59] S. Dockins, “INTERVERTEBRAL DISCS IN YOGA: MANAGING MECHANICAL FORCES,” 16 January 2020. [Online]. Available: https://www.stacydockins.com/posture-lab/2020/1/16/protecting-lumbar-discs-managing-mechanical-forces.

[60] M. M. O. J, H. A. B, M. P. Susai and C. Zizhou, “Analysis of soft tissue in cervical traction therapy using finite element method,” in *2nd International conference on Advances in Mechanical Engineering*, 2018.

[61] K. Evangeline, S. P. Angeline and D. Raja, “A biomechanical study of T12-L2 in thoracolumbar region of spine affected by tuberculosis,” in *IOP Conference Series: Materials Science and Engineering*, 2018.

[62] M. Minárová and J. Sumec, “Stress - Strain Response of the Human Spine Intervertebral Disc As an Anisotropic Body. Mathematical Modeling and Computation.,” *Open Phys. ,* p. 4:426–435, 2016.

[63] C. Badii, “Making Sense of Hypertensive Retinopathy,” 31 March 2017. [Online]. Available: https://www.healthline.com/health/hypertensive-retinopathy.

[64] A. D. Henderson, B. B. Bruce, N. J. Newman and V. Biousse, “Hypertension-related eye abnormalities and the risk of stroke,” *HHS Public Access - Rev Neurol Dis,* pp. 8(1-2): 1-9, 2012.

[65] T. Y. Wong, “Hypertesive Retinopathy: The Cardiovascular Connection,” 16 March 2007. [Online].

[66] T. Y. Wong and R. Mclntosh, “Hypertensive retinopathy signs as risk indicators of cardiovascular morbidity and mortality,” *British Medical Bulletin,* pp. 57-70, January 2005.

[67] M. P, W. JJ, W. TY, S. W, K. R and L. SR, “Retinal microvascular signs and risk of stroke and stroke mortality,” 11 October 2005. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/16217050/.

[68] R. O. Mujeeb, “ Introduction to artifical neural network and machine learning,” Palakkad, 2012.

[69] K. Preeti, L. Shreenivas and D. Makarand, “Artificial Neural Networks for Construction Management: A Review,” India, 2017.

[70] K. Goyal, “The 7 Types of Artificial Neural Networks ML Engineers Need to Know,” 17 December 2019. [Online]. Available: https://www.upgrad.com/blog/types-artificial-neural-networks-in-machine-language/.

[71] P. Saini and A. Tiwari, “What is Image Processing? Explain fundamental steps in Digital Image Processing.,” 2020. [Online]. Available: https://www.ques10.com/p/33595/what-is-image-processing-explain-fundamental-steps/.

[72] U. G. Abbasi and M. U. Akram, “Classification of Blood Vessels as Arteries and Veins for Diagnosis of Hypertensive Retinopathy,” *IEEE,* pp. 978-1-4799-5241-0114/$3100, 2014.

[73] N. L. Stokoe and R. W. D. Turner, “NORMAL RETINAL VASCULAR PATTERN*,” *Brit. J. Ophthal,* pp. 21-40, 1965.

[74] K. Amit, K. Navneet and S. Prabhjit, “Eye Diabetic Retinopathy by Using Deep Learning,” *International Research Journal of Engineering and Technology (IRJET),* pp. 2504 - 2508, 2018.

[75] L. D. Hubbard, R. J. Brothers, W. N. King, L. X. Clegg, R. Klein, L. S. Cooper, A. R. Sharrett, M. D. Davis, J. Cai and A. R. i. Communities, “Methods for Evaluation of Retinal Microvascular Abnormalities Associated with Hypertension/Sclerosis in the Atherosclerosis Risk in Communities Study,” *Ophthalmology Volume 106,,* pp. 2269-2280, 1999.

[76] G. DaiI, W. He, L. Xu, E. E. Pazo, T. Lin, Shasha and C. Zhang, “Exploring the effect of hypertension on retinal microvasculature using deep learning on East Asian population,” 5 March 2020. [Online]. Available: https://doi.org/10.1371/journal.pone.0230111.

[77] S. Das, “CNN Architectures: LeNet, AlexNet, VGG, GoogLeNet, ResNet and more…,” 16 November 2017. [Online]. Available: https://medium.com/analytics-vidhya/cnns-architectures-lenet-alexnet-vgg-googlenet-resnet-and-more-666091488df5.

[78] M. Park, “Artificial Intelligence Trends of 2021,” 22 September 2021. [Online]. Available: https://cielo24.com/2021/09/artificial-intelligence-trends-of-2021/.

[79] R. Karim, “Illustrated: 10 CNN Architectures,” 29 July 2019. [Online]. Available: https://towardsdatascience.com/illustrated-10-cnn-architectures-95d78ace614d#e971.

[80] R. Quiza and J. P. Davim, “Computational methods and optimization,” in *Machining of hard materials*, London, Springer-Verlag, 2011, pp. 177-208.

[81] P. Kumar and K. Verma, “Use of Fuzzy Set and Neural Network to Extract Fingerprint Minutiae Points and Location,” June 2009. [Online]. Available: https://www.researchgate.net/publication/268363154.

[82] P. Mehrani and J. K. Tsotsos, “Early recurrence enables figure border ownership,” January 2019. [Online]. Available: https://www.researchgate.net/publication/330304826_Early_recurrence_enables_figure_border_ownership.

[83] G. Anuradha, “Artificial Neural Networks An Introduction G Anuradha Learning,” 2014. [Online]. Available: https://slidetodoc.com/artificial-neural-networks-an-introduction-g-anuradha-learning-2/.

Appendix

ResNet-101

Import Data

Import training and validation data.

```
imds = imageDatastore("C:\Users\HP\Desktop\Hypertensive Retinopathy\Grading\Grader 2\","IncludeSubfolders",true,"LabelSource","foldernames");
[imdsTrain, imdsValidation] = splitEachLabel(imds,0.7,"randomized");
% Resize the images to match the network input layer.
augimdsTrain = augmentedImageDatastore([224 224 3],imdsTrain);
augimdsValidation = augmentedImageDatastore([224 224 3],imdsValidation);
```

Set Training Options

Specify options to use when training.

```
opts = trainingOptions("sgdm",...
    "ExecutionEnvironment","auto",...
    "InitialLearnRate",0.001,...
    "MaxEpochs",15,...
    "MiniBatchSize",30,...
    "Shuffle","every-epoch",...
    "ValidationFrequency",30,...
    "Plots","training-progress",...
    "ValidationData",augimdsValidation);
```

Train the data

```
Model = trainNetwork(augimdsTrain,RNET4,opts);
```

Classification Matrix

```
[YPred,probs] = classify(Model,augimdsValidation);
accuracy = mean(YPred == imdsValidation.Labels);
display(['ResNet-101Net-101:',num2str(100*accuracy),'%'])
YTrue = imdsValidation.Labels;
figure;
confusionchart(YTrue,YPred)
```

```
tbl = countEachLabel(imds)
tb2 = countEachLabel(imdsTrain)
tb3 = countEachLabel(imdsValidation)
```

GoogleNet

Import Data

Import training and validation data.

```
imds = imageDatastore("C:\Users\HP\Desktop\Hypertensive Retinopathy\Grading\Grader 2\","IncludeSubfolders",true,"LabelSource","foldernames");
[imdsTrain, imdsValidation] = splitEachLabel(imds,0.7,"randomized");

% Resize the images to match the network input layer.
augimdsTrain = augmentedImageDatastore([224 224 3],imdsTrain);
augimdsValidation = augmentedImageDatastore([224 224 3],imdsValidation);
```

Set Training Options

Specify options to use when training.

```
opts = trainingOptions("sgdm",...
    "ExecutionEnvironment","auto",...
    "InitialLearnRate",0.001,...
    "MaxEpochs",15,...
    "MiniBatchSize",30,...
    "Shuffle","every-epoch",...
    "ValidationFrequency",30,...
    "Plots","training-progress",...
    "ValidationData",augimdsValidation);
```

Train the data

```
Model = trainNetwork(augimdsTrain,GNET4,opts);
```

Classification Matrix

```
[YPred,probs] = classify(Model,augimdsValidation);
accuracy = mean(YPred == imdsValidation.Labels);
display(['GoogleNet:',num2str(100*accuracy),'%'])
YTrue = imdsValidation.Labels;
figure;
confusionchart(YTrue,YPred)
```

```
tbl = countEachLabel(imds)
tb2 = countEachLabel(imdsTrain)
tb3 = countEachLabel(imdsValidation)
```

AlexNet

Import Data

Import training and validation data.

```
imds = imageDatastore("C:\Users\HP\Desktop\Hypertensive Retinopathy\Grading\Grader
2\","IncludeSubf         olders",true,"LabelSource","foldernames");
[imdsTrain, imdsValidation] = splitEachLabel(imds,0.7,"randomized");


% Resize the images to match the network input layer.
augimdsTrain = augmentedImageDatastore([227 227 3],imdsTrain);
augimdsValidation = augmentedImageDatastore([227 227 3],imdsValidation);
```

Set Training Options

Specify options to use when training.

```
opts = trainingOptions("sgdm",...
    "ExecutionEnvironment","auto",...
    "InitialLearnRate",0.001,...
    "MaxEpochs",15,...
    "MiniBatchSize",30,...
    "Shuffle","every-epoch",...
    "ValidationFrequency",30,...
    "Plots","training-progress",...
    "ValidationData",augimdsValidation);
```

Train the data

```
Model = trainNetwork(augimdsTrain,ALEXN4,opts);
```

Classification Matrix

```
[YPred,probs] = classify(Model,augimdsValidation);
accuracy = mean(YPred == imdsValidation.Labels);
display(['ALEXNET:',num2str(100*accuracy),'%'])
YTrue = imdsValidation.Labels;
figure;
confusionchart(YTrue,YPred)

tbl = countEachLabel(imds)
tb2 = countEachLabel(imdsTrain)
tb3 = countEachLabel(imdsValidation)
```

VGG-19

Import Data

Import training and validation data.

```
imds = imageDatastore("C:\Users\HP\Desktop\Hypertensive Retinopathy\Grading\Grader 2\","IncludeSubfolders",true,"LabelSource","foldernames");
[imdsTrain, imdsValidation] = splitEachLabel(imds,0.7,"randomized");


% Resize the images to match the network input layer.
augimdsTrain = augmentedImageDatastore([224 224 3],imdsTrain);
augimdsValidation = augmentedImageDatastore([224 224 3],imdsValidation);
```

Set Training Options

Specify options to use when training.

```
opts = trainingOptions("sgdm",...
    "ExecutionEnvironment","auto",...
    "InitialLearnRate",0.001,...
    "MaxEpochs",15,...
    "MiniBatchSize",30,...
    "Shuffle","every-epoch",...
    "ValidationFrequency",30,...
    "Plots","training-progress",...
    "ValidationData",augimdsValidation);
```

Train the data

```
Model = trainNetwork(augimdsTrain,V19NET4,opts);
```

Classification Matrix

```
[YPred,probs] = classify(Model,augimdsValidation);
accuracy = mean(YPred == imdsValidation.Labels);
display(['VGG-19:',num2str(100*accuracy),'%'])
YTrue = imdsValidation.Labels;
figure;
confusionchart(YTrue,YPred)
```

```
tbl = countEachLabel(imds)
tb2 = countEachLabel(imdsTrain)
tb3 = countEachLabel(imdsValidation)
```

Xception

Import Data

Import training and validation data.

```
imds = imageDatastore("C:\Users\HP\Desktop\Hypertensive Retinopathy\Grading\Grader 2\","IncludeSubfolders",true,"LabelSource","foldernames");
[imdsTrain, imdsValidation] = splitEachLabel(imds,0.7,"randomized");
% Resize the images to match the network input layer.
augimdsTrain = augmentedImageDatastore([227 227 3],imdsTrain);
augimdsValidation = augmentedImageDatastore([227 227 3],imdsValidation);
```

Set Training Options

Specify options to use when training.

```
opts = trainingOptions("sgdm",...
    "ExecutionEnvironment","auto",...
    "InitialLearnRate",0.001,...
    "MaxEpochs",15,...
    "MiniBatchSize",30,...
    "Shuffle","every-epoch",...
    "ValidationFrequency",30,...
    "Plots","training-progress",...
    "ValidationData",augimdsValidation);
```

Train the data

```
Model = trainNetwork(augimdsTrain,XCep44,opts);
```

Classication Matrix

```
[YPred,probs] = classify(Model,augimdsValidation);
accuracy = mean(YPred == imdsValidation.Labels);
display(['XceptionNet:',num2str(100*accuracy),'%'])
YTrue = imdsValidation.Labels;
figure;
confusionchart(YTrue,YPred)
```

```
tbl = countEachLabel(imds)
tb2 = countEachLabel(imdsTrain)
tb3 = countEachLabel(imdsValidation)
```