

**A TRANSFER LEARNING AND TWO-LEVEL HYPERPARAMETER  
OPTIMIZATION BASED MODEL FOR IMPROVED CLASSIFICATION OF  
DIABETIC RETINOPATHY**

**Jackson Kamiri Wambugu**

**A Thesis Submitted in Partial Fulfilment of the Requirements for the Degree of  
Master of Science in Information Technology of Murang'a University of  
Technology**

**October, 2022**

## DECLARATION

I hereby declare that this thesis is my original work and to the best of my knowledge has not been presented for a degree award in this or any other university.

---

Jackson Kamiri Wambugu  
SC401/5412/2019

---

Date

## APPROVAL

The undersigned certify that they have read and hereby recommend for acceptance of Murang'a University of Technology a thesis entitled "**A Transfer Learning and Two-Level Hyperparameter Optimization Based Model for Improved Classification of Diabetic Retinopathy**"

---

Dr. Geoffrey Mariga Wambugu  
Department of Information Technology,  
Murang'a University of Technology.

---

Date

---

Dr. Aaron Mogeni Oirere,  
Department of Computer Science,  
Murang'a University of Technology.

---

Date

## **DEDICATION**

I dedicate this thesis to my family: my wife Faith Kamiri, our lovely daughter Tatiana Kamiri, my mum Susan Kamiri, and my Brother Paul Maina. For the support they have accorded me and the time they have missed me while I was doing this research.

## **ACKNOWLEDGEMENT**

I wish to thank the almighty God for his unlimited blessings all through my life. It is thorough his mercy and favor that I have managed to achieve this progress in my life. In a special way I wish to thank the management of Murang'a University of Technology for offering me a scholarship to pursue Master of Science in Information Technology.

I acknowledge the unwavering input of my supervisors Dr. Geoffrey Mariga and Dr. Aaron Oirere, thank you for your guidance all through this research. Sincere gratitude to all the staff of the School of Computing and Information Technology (SCIT), Murang'a University of Technology (MUT) for the valuable guidance and positive critiques during my studies and progressive research presentations and seminars.

Heartfelt appreciation to the love of my life Faith Kamiri and our adorable daughter Tatiana Kamiri. You have supported me and believed in me even when I doubted myself.

## ABSTRACT

Automated diagnosis of disease from medical images using machine learning has been in rise in the recent past. One such case is the classification of diabetic retinopathy from fundus images. Diabetic Retinopathy is an eye disease that is a result of diabetes mellitus and it is major cause of blindness among people of the working age. Diabetic retinopathy has five main classes namely: No DR, Mild DR, Moderate DR, Severe DR, and Proliferative DR. Deep learning has been used previously in this field and it has proved to be better than conventional machine learning approaches. However, deep learning involves training a model from scratch thus making it to be data hungry, require high training cost, have poor generalizability, and they don't deliver high performance. Meta-learning also known as learning-to-learn is a field of machine learning which aims at improving deep learning by enabling models to improve their performance capabilities and reduce training cost. Meta-learning techniques include multi-task learning, transfer learning, self-optimization, and few-shot learning. Several transfer learning architectures pre-trained on the ImageNet dataset have been used by different researchers and they have demonstrated superior performance over deep learning. However, domain-shift generalizability and optimal performance of pre-trained architectures are major challenges facing transfer learning. This so because these models are not properly tuned for cross-domain optimality. The aim of this study was to develop an improved model for classification of diabetic retinopathy into its five classes. To achieve this, the researcher used the following approach: A VGG16 network pre-trained in ImageNet was modified such that the top-layer was rebuilt and an attention model was added. Two-level optimization was used during training in which the model was allowed to self-tune its learning rate based on the training parameters. The EyePACS dataset obtained from Kaggle repository was used in training, validating, and testing the model. The model was developed in Google Collaboratory platform using python programming language, TensorFlow, and Keras. The study achieved the following results: Accuracy 89.06%, Precision 88.9%, Recall 89.2%, F1-Score 75%, Quadratic Cohen Kappa Metric 0.84, Area Under the Curve (AUC) 93.3%. The results of the study demonstrated improved performance compared to other existing models in literature such as Qummar et al (2019), Jinfeng et al (2020), Chilukoti et al (2022), that classify diabetic retinopathy into five classes. The study concluded that leveraging on previously acquired knowledge and efficient optimization of neural networks using data driven self-optimization delivers better performance than conventional machine learning and deep learning. In future researchers can consider using reinforcement learning and transfer learning in classification of diabetic retinopathy.

## TABLE OF CONTENTS

<b>DECLARATION.....</b>	<b>i</b>
<b>DEDICATION.....</b>	<b>ii</b>
<b>ACKNOWLEDGEMENT.....</b>	<b>iii</b>
<b>ABSTRACT.....</b>	<b>iv</b>
<b>LIST OF TABLES.....</b>	<b>viii</b>
<b>LIST OF FIGURES.....</b>	<b>ix</b>
<b>ACRONYMS AND ABBREVIATIONS.....</b>	<b>xi</b>
<b>DEFINITION OF TERMS.....</b>	<b>xii</b>
<b>CHAPTER ONE: INTRODUCTION.....</b>	<b>1</b>
1.1 Background of the Study.....	1
1.2 Problem Statement.....	5
1.3 Objectives.....	6
1.3.1 General Objective.....	6
1.3.2 Specific Objectives.....	6
1.4 Research Questions.....	6
1.5 Significance of the study.....	6
1.6 Scope of the study.....	7
1.7 Limitations of the Study.....	8
1.8 Contributions of the Thesis.....	8
1.9 Organization of the Thesis.....	8
<b>CHAPTER TWO: LITERATURE REVIEW.....</b>	<b>10</b>
2.1 Introduction.....	10
2.2 Diabetic Retinopathy.....	10
2.3 Empirical Studies.....	15
2.3.1 Detection using Convolutional Neural Network.....	15
2.3.2 Using Mask RCNN and Transfer Learning.....	16
2.3.3 Convolutional Neural Network.....	17
2.3.4 Transfer Learning and Small Dataset.....	18
2.3.5 Meta-Learning with Noisy Data.....	19
2.3.6 A deep Learning Ensemble Approach.....	21
2.3.7 Ensemble Framework of Deep CNNs.....	22
2.3.8 Deep Learning Approach.....	24
2.3.9 Detection Using VGG-NIN a Deep Learning Architecture.....	25

2.3.10 Transfer Learning from Pre-trained CNN Models.....	26
2.3.11 Transfer Learning Based Robust Automatic Detection .....	27
2.4 Machine Learning Model Development Approaches .....	28
2.4.1 Data Management .....	29
2.4.2 Model Training .....	31
2.5 Meta-Learning .....	37
2.6 Meta-Learning Approaches .....	40
2.6.1 Multi-Task Learning .....	41
2.6.2 Transfer Learning .....	43
2.6.3 Hyperparameter optimization .....	51
2.6.4 Ensemble Learning .....	52
2.7 Model Evaluation.....	55
2.7.1 Confusion Matrix.....	56
2.7.2 Quadratic Weighted Kappa Metric (QK).....	60
2.8 Conceptual Framework.....	61
2.9 Summary.....	62
<b>CHAPTER THREE: METHODOLOGY.....</b>	<b>64</b>
3.1 Introduction.....	64
3.2 Research Process.....	64
3.3 Research Design .....	65
3.4 Identification of Best Architecture for Transfer learning Model.....	66
3.5 Data Sourcing .....	70
3.6 Model Development .....	72
3.7 Model Validation.....	81
3.8 Model Testing and Evaluation.....	81
3.9 Tools and Techniques used.....	82
3.10 Ethical Considerations .....	83
3.11 Summary.....	83
<b>CHAPTER FOUR: RESULTS AND DISCUSSION.....</b>	<b>84</b>
4.1 Introduction.....	84
4.2 Identification of Best Architecture for Transfer learning Model.....	84
4.2.1 Model Parameters .....	84
4.2.2 Comparison of the performance achieved by the Models.....	85
4.3 Modified Model Architecture .....	90
4.4 Modified Model Algorithm .....	91

4.5 Summary of hyperparameters after Tuning .....	92
4.6 Results of the improved model's performance .....	93
4.6.1 Top layer Training .....	94
4.6.2 Full Model Session one.....	95
4.6.3 Full model Second Session .....	97
4.6.4 Full model Third Session .....	99
4.7 Comparison of the Results with the existing models .....	100
4.8 Discussion.....	101
4.8.1 Identification of Best Architecture for Transfer learning Model.....	101
4.8.2 Top layer only.....	103
4.8.3 Full model Session one .....	104
4.8.4 Full Model Session Two .....	104
4.8.5 Full Model Session Three .....	105
4.9 Summary.....	107
<b>CHAPTER FIVE: CONCLUSION, RECOMMENDATIONS AND FUTURE</b>	
<b>WORK .....</b>	<b>108</b>
5.1 Conclusion .....	108
5.2 Recommendation .....	110
5.2.1 The Ministry of health to collect and maintain a database of medical datasets ..	110
5.2.2 Adoption of developed models .....	110
5.2.3 Facilitation of postgraduate students .....	111
5.3 Future work.....	111
5.3.1 Use of another type of machine learning .....	111
5.3.2 Increase Multi-task capabilities of the model .....	112
5.3.2 Creating a G.U.I to facilitate deployment of this model .....	112
<b>REFERENCES.....</b>	<b>113</b>
<b>APPENDICES .....</b>	<b>130</b>



## LIST OF TABLES

Table 2.1: International Clinical Diabetic Retinopathy Disease Severity Scale.....	12
Table 3.1: Dataset Distribution .....	71
Table 4.1: Comparative Analysis of Models' parameters .....	85
Table 4.2: Comparison of the four models in both deep learning and Transfer learning accuracies.....	86
Table 4.3: Final Hyperparameters Obtained After Tuning .....	93
Table 4.4: Results of Training the Top Layer Only.....	94
Table 4.5: Results of the First Session of Training the Full Model.....	96
Table 4.6: Results of Training the Full Model for the Second Session .....	98
Table 4.7: Results of Training the Full Model for the Third Session .....	99
Table 4.8: Comparison of Existing Models to the Improved Model .....	101

## LIST OF FIGURES

Figure 2.1: Fundus Images showing Stages of Diabetic Retinopathy .....	13
Figure 2.2: SVM Linear Classifier.....	33
Figure 2.3: Deep Neural Network.....	36
Figure 2.4: Meta-Learning .....	40
Figure 2.5: Hard parameter Sharing in Multi-task Learning .....	42
Figure 2.6: Soft Parameter Sharing.....	43
Figure 2.7: DenseNet Architecture .....	46
Figure 2.8: Residual Learning Block.....	47
Figure 2.9: VGG Architecture .....	50
Figure 2.10: Stacking .....	54
Figure 2.11: Bagging .....	55
Figure 2.12: Confusion Matrix .....	56
Figure 2.13: Conceptual Framework .....	62
Figure 3.1: Research Process .....	65
Figure 3.2: Code for importing VGG16 Architecture .....	72
Figure 3.3: Layer-by-layer VGG16 Architecture before layers' Modification .....	73
Figure 3.4: Model Development and Training .....	77
Figure 3.5: Two-Level Optimization .....	79
Figure 4.1: EfficientNetB0 Deep Learning and Transfer Learning Accuracies .....	87
Figure 4.2: DenseNet169 Deep Learning and Transfer Learning Accuracies.....	88
Figure 4.3: ResNet50 Deep Learning and Transfer Learning Accuracy Curves .....	89
Figure 4.4: VGG16 Deep Learning and Transfer Learning Accuracy Curves .....	90
Figure 4.5: Modified Model Architecture.....	91

Figure 4.6 Top Layer Training and Validation Curve .....	95
Figure 4.7: Full Model Session 1 Training and Validation Curves .....	97
Figure 4.8: Full Model Session Two training and validation curves .....	98
Figure 4.9: Full Model Session Three training and validation curves .....	100

## ACRONYMS AND ABBREVIATIONS

<b>AUC</b>	Area Under the Curve
<b>CNN</b>	Convolutional Neural Network
<b>DenseNet</b>	Densely Connected Neural Networks
<b>DL</b>	Deep Learning
<b>DR</b>	Diabetic Retinopathy
<b>FN</b>	False Negative
<b>FP</b>	False Positive
<b>NACOSTI</b>	National Commission for Science, Technology and Innovation
<b>NIN</b>	Network in Network
<b>NPDR</b>	Non-proliferative diabetic retinopathy
<b>PDR</b>	Proliferative Diabetic Retinopathy
<b>QWK</b>	Quadratic Weighted Kappa Metric
<b>ResNet</b>	Residual Network
<b>SDG</b>	Stochastic Gradient descent
<b>TL</b>	Transfer Learning
<b>TN</b>	True Negative
<b>TP</b>	True Positive
<b>VGGNet</b>	Visual Geometry Group Network

## DEFINITION OF TERMS

**Diabetic Retinopathy:** Diabetic retinopathy (DR) is an eye disease that is a result of diabetes mellitus. It is characterized with damaged blood vessels in the retina, swollen or leaking vessels, some close thus stopping blood from passing through them, and abnormal vessels can grow in the retina.

**Domain-shift Generalizability:** Refers to the model's ability to adapt properly to a new but related task which belongs to a different domain from which the model was pre-trained on.

**Generalizability:** the model's ability to adapt properly to new, previously unseen data, drawn from same distribution as the one used to create the model.

**Meta-Learning (learning-to-learn):** it is a field of Artificial intelligence that aims at improving the conventional deep learning approaches by providing a collection of techniques that facilitate a model to learn from multiple tasks, leverage on previously acquired knowledge, and perform self-optimization.

**Multi-task Learning:** This refers to a learning approach that aims at using shared knowledge among tasks to jointly learn from the multiple tasks. It also involves leveraging on the knowledge acquired from these multiple tasks to improve a model's performance on an individual task.

**Transfer Learning:** Transfer learning is part of meta-learning that focuses on transferring knowledge that is learned from a task A to a new related task B.

## CHAPTER ONE

### INTRODUCTION

#### 1.1 Background of the Study

Machine learning is one of the technologies that has been highly applied in solving some of the world's major challenges. One of the areas in which machine learning has been widely applied is in healthcare [1]. Machine learning is a subset of Artificial Intelligence that provides systems with the ability to learn without being explicitly programmed [2]. Therefore, once machine learning models are provided with training data, they can be able to learn from the data and make decisions based on what the models have learned. This is unlike traditional programming which is rule-based and programs act as per the conventions of the rules.

Machine learning is categorized into four main branches which according to [2] are: first, supervised learning, this uses labeled data to train and test the machine learning algorithm. Second, unsupervised learning, this involves training and testing machine learning models with unlabeled data. Third, reinforcement learning, this one uses a reward system in which the model learns as it goes by using trial and error. A series of successful outcomes are reinforced to come up with the final model. Fourth, semi supervised learning, this is a learning problem that involves a large number of unlabeled examples and a small number of labeled examples [2].

Machine learning can be used to perform different sets of activities which include the following: First, regression which is a supervised learning task which focuses on predicting the value of label based on a set of features[3]. The dependency of a label on the training data in regression is determined by how the label will change as the value of the features are varied. Regression can be used in tasks such as predicting the price of commodities in future. Second, clustering which is an unsupervised learning

task which groups instances of data into clusters that contain similar characteristics[4]. Clustering can be used in tasks such as market segmentation.

Third classification which is a supervised learning task which is used to determine a class or category that an instance belongs to. Classification can be grouped into either multi-class classification or binary classification. Binary classification involves predicting which of two categories an instance belongs to[5]. Therefore, it only classifies instances into two categories only. A sample application of binary classification is detecting if a patient is +ve for certain ailment or not. Multi-class classification is an advancement of binary classification that involves classifying instances into more than one classes. Multi-class classification therefore enables data analyst to classify data into different dimensions[6]. For instance, it can be used to detect if a patient has a certain illness or not and then if the patient has the illness it further classifies the illness into different stages of the illness[6].

The medical field has been a major area in which machine learning researchers have developed solutions for. Deep learning which involves use of neural networks has especially proofed to be superior in medical imaging task. This involves diagnosing and classifying medical conditions from medical images. Some of the deep learning solutions that have been developed include: Brain tumor detection[7], cancer detection from medical images[8], diabetic retinopathy detection[9], and CT-scan images analysis to determine extent of damage[10].

Developing deep learning models involves the following series of steps: First, choosing the dataset to use. Second, performing data preprocessing. Third, divide the dataset into training, validation, and testing sets. Fourth, model training using the training dataset. Fifth, validating the model. Sixth, testing and evaluating the model

[11] [12]. The above-described approach also known as the conventional approach has three main problems that make it perform poorly.

First, it requires a lot of training data to be available to learn effectively. Where large volumes of training data are not available the models trained using the conventional approach performs poorly. Secondly, the choice of hyper-parameters also affects how the model's performance. Poor choice of hyperparameters results in poor performance [13] [14]. Third, conventional approaches such as deep learning require high computational power thus limiting scalability of conventional models. Fourth, the conventional approaches lack the capabilities to learn from different tasks and transfer the acquired knowledge to solving a new task[13].

To address these problems, the concept of Meta-learning was introduced. Meta-learning is also known as learning-to-learn. It is a field of machine learning that aims at improving the conventional deep learning approaches where tasks are performed from scratch using a fixed learning algorithm[13]. Meta-learning focuses on improving the algorithm itself so that the model can achieve optimal performance with limited resources compared to other machine learning approaches. This provides an avenue of solving the key machine learning challenges which include need for large volumes of data, performance, generalization, and computational bottlenecks [14].

Meta-learning has the following variants; First, multi-tasking learning, which deals with training the algorithm with multiple sets of training data. Second, transfer learning which involves transferring the knowledge obtained from multitask learning into solving a new task. Third, Hyper-parameter optimization, this deals with obtaining the optimal hyperparameters of an algorithm. Fourth, learning to learn collectively which includes bagging, boosting, and stacked generalization [13][14]



[15][16]. Scholars, such as [13] have demonstrated that by using Meta-Learning techniques, a state-of-the-art performance of the algorithm is achieved.

Transfer learning has attracted many researchers such as [17] [18], due to its capabilities of enabling a model to leverage on previous knowledge. Transfer learning transfers knowledge acquired from multitask learning in form of weights to solving new tasks. For transfer learning to be successful there is need for optimal hyperparameter tuning so that the model can easily achieve good performance with minimal training effort [19]. This brings value to fields such as Diabetic retinopathy classification where high performance is a necessity yet huge volumes of training data are not available.

Diabetic retinopathy (DR) is an eye disease that is a result of diabetes. It is characterized with damaged blood vessels in the retina, swollen or leaking vessels, some close thus stopping blood from passing through them, and abnormal vessels can grow in the retina. Eventually these changes can result to lose of vision [20]. DR has five categories namely: No DR, Mild DR, Moderate DR, Severe DR, and Proliferative DR [21]. Deep learning has been used by other researchers such as [22] for classification of diabetic retinopathy. Deep learning models registered low performance due to learning from scratch. Transfer learning models have proved superiority over deep learning as demonstrated by [18][17][23]. However, the existing transfer learning models have not achieved domain-shift generalizability due to lack of optimal tuning of hyperparameters to achieve optimal performance of the models.

In Africa, a study conducted in Zambia's copper belt province revealed that out of the 2689 participants, 52% showed evidence of Diabetic retinopathy. Among these positive patients, 36% had sight threatening diabetic retinopathy while 7% had

proliferative diabetic retinopathy[24]. Another study was conducted in Sudan by [25]. The aim was to determine the prevalence of diabetic retinopathy among people with diabetes. Out of 316 participants screened 261 (82.6%) had diabetic retinopathy. Of the 82.6% 39.9% had proliferative diabetic retinopathy[25]. A study conducted in Nakuru county, Kenya by [26] reported that the estimated prevalence of diabetic retinopathy among people with diabetes mellitus is 224.7 persons for every 1000 persons. The study also estimated the prevalence of Diabetic retinopathy among people without diabetes mellitus is 15.8 per 1000 persons[26].

## **1.2 Problem Statement**

Transfer learning is part of meta-learning that deals with transferring knowledge acquired from multitask learning into solving a new task[13][19]. Transfer learning has been applied in various image processing tasks with one of them being classification of diabetic retinopathy[17][18][19]. Ideally, a model based on transfer learning should be able to leverage on previously acquired knowledge into solving the task of classification of diabetic retinopathy with minimal training overheads and achieve good performance.

However, this is not the case since domain shift generalizability and optimal performance of pre-trained architectures are major challenges facing transfer learning. This so because existing transfer learning models for detection of diabetic retinopathy such as [17][18][19] are not efficiently tuned for cross-domain optimality.

Also, performance of existing diabetic retinopathy detection models such as [17][18][19] [27] & [28] are challenged in self-optimization, increased training overhead due to many trainable parameters and failure to consider all classes of diabetic retinopathy. Therefore, there is need to develop a model that utilizes data

driven self-optimization to achieve domain-shift generalizability and improved performance.

### **1.3 Objectives**

#### **1.3.1 General Objective**

The general objective of this study was to develop an improved diabetic retinopathy classification model using transfer learning approach and hyperparameter optimization.

#### **1.3.2 Specific Objectives**

- i. To analyze current deep learning and transfer-learning models used for classification of diabetic retinopathy with the aim of selecting the best architecture.
- ii. To redesign selected model architecture and tune training hyperparameters with the aim of improving performance.
- iii. To validate and test the improved model.

### **1.4 Research Questions**

- i. What are the existing transfer learning and deep learning models that have been used for classification of diabetic retinopathy?
- ii. How can the model architecture be redesigned and training hyperparameters tuned to improve performance?
- iii. How can the improved model be validated and tested?

### **1.5 Significance of the study**

This study is important because it brings on board a new model that improves on the automatic classification of diabetic retinopathy by utilizing two levels of optimization to optimize the model hyperparameters. The findings of the study show that the model

delivers more reliable performance compared to other models in related works. This improvement reduces chances of misdiagnosis and misclassification. The high specificity achieved by the model compared to others in literature is a clear indication that the model has very high capabilities of ensuring that majority of the positive patients will be identified as such. The new model is also better than the existing ones since it has better domain generalizability. Therefore, in case there is need for the model to be used for detecting other Eye complications associated with diabetic retinopathy, the model will require only minimal training to achieve high performance.

### **1.6 Scope of the study**

The research is domiciled in the field machine learning, subfield of meta-learning. The study is confined within transfer learning as one of the major approaches of meta-learning. The first objective focused on analyzing the existing works and choosing a suitable architecture that can be used for transfer learning. The second objective was confined on redesigning the chosen architecture and tuning its training hyperparameters to improve performance. The aim of the model was to achieve superior performance in classification of Diabetic retinopathy compared to existing models in literature.

The third objective was confined on validating the model using simulation. The researcher opted to use simulation in validation due to the following reasons: First, validation using simulation is the standard practice in machine learning since other researchers such as [9][17][18][23] have used the same approach. Second, validation by simulation is less costly compared to others such as engaging experts. It also focused on evaluating the model's performance using Precision, Recall, Quadratic Weighted Kappa Metric, Accuracy, F1-score, and Area Under the Curve as the Evaluation Metrics.

## **1.7 Limitations of the Study**

The researcher faced the challenge of acquiring a balanced labeled dataset with all the five classes of diabetic retinopathy. The dataset that was obtained which is the EyePACS dataset was quite imbalanced. Therefore, the dataset if used without modification would have skewed learning towards the majority class. To overcome this challenge, the researcher augmented the minority classes using TensorFlow library and reduced the data imbalance gap.

## **1.8 Contributions of the Thesis**

The contributions made by this thesis are highlighted below:

- i. A new improved model was developed which achieves better performance compared to the existing models
- ii. Model optimization was done using two-level optimization thus allowing the model to autotune some of its hyperparameters based on the training data.
- iii. Domain-shift generalizability was achieved as evidenced by the model's superior performance in the task as compared to existing models in literature.

## **1.9 Organization of the Thesis**

This thesis is organized into five chapters as highlighted below;

Chapter one discusses the background of the study, the problem statement, the research objectives, research questions, significance of the study, scope of the study, limitations of the study, conceptual framework, and the contributions of the thesis. Chapter Two consists of the literature review. The chapter discusses the field of study in details and analyzes the previous studies that had been done in this area. Chapter three discusses the methodology that the researchers used to achieve the objectives of the study. Chapter four consist of the results obtained from the study and their

discussion. Chapter five contains the conclusion, recommendations and future works.

The references and the Appendices follow after chapter five.

## **CHAPTER TWO**

### **LITERATURE REVIEW**

#### **2.1 Introduction**

This chapter reviews the literature on previous works done by other researchers in this field. The aim of the chapter was to analyze what other scholars in the field have discovered.

#### **2.2 Diabetic Retinopathy**

Diabetic retinopathy (DR) is an eye disease that is a result of diabetes mellitus and it's a major cause of blindness among people who are of working-age[21]. It is most common microvascular complication and it is more prevalent among people with type I diabetes mellitus than those with type II diabetes mellitus[20]. It is characterized with damaged blood vessels in the retina, swollen or leaking vessels, some close thus stopping blood from passing through them, and abnormal vessels can grow in the retina[22]. Thus, eye fundus images are used to detect diabetic retinopathy and even determine the stage at which the complication is at[21].

Clinically diabetic retinopathy can be grouped into two classes namely; proliferative DR (PDR) and non-proliferative DR (NPDR). Excessive sugar levels are main cause NPDR and they start by affecting the tiny blood vessels in the eye's retina[18]. This makes the blood vessels to swell and fluids to leak from them which results to the body lacking oxygen and nutrients. Proliferative DR on the other hand is an advanced stage which is caused by more swelling and leaking. This is a very hazardous stage that can result to permanent vision loss[18].

NPDR is further classified into Mild DR, Moderate DR, and Severe DR. "Mild NPDR exhibits only microaneurysms, there is a small round red spot at the end of the blood capillary. Moderate NPDR presents with additional signs of impaired vessel integrity

and vessel occlusion, including dot and more than five microaneurysms occurs with flame-shaped hemorrhages[23], hard exudates, and cotton wool spots. Severe NPDR is accompanied by more distinct features of retinal ischemia, such as venous beading and intra-retinal microvascular abnormalities (IRMAs) that are adjacent to non-perfusion areas”[20] and there are more than 20 intraretinal hemorrhages[23]. “PDR, is a more advanced stage of DR, is characterized by neovascularization. During this stage, the patients may experience severe vision impairment when the new abnormal vessels bleed into the vitreous (vitreous hemorrhage) or when tractional retinal detachment is present”[21].

Diabetic retinopathy can therefore be classified into five classes namely; No DR, mild DR, Moderate DR, Severe DR, Proliferative DR. These classes are used in medical imaging of eye fundus images [22]. “The fundus is the inner surface of the eye, which is nearly opposite to the lens and includes the macula, retina, fovea, optic disc, and posterior pole” [23]. “The distinguishing factor between these sub-categories is the presence of micro-aneurysms number and intensity (MA). Table 2.1 shows clinical diabetic retinopathy severity scale[21].



**Table 2.1: International Clinical Diabetic Retinopathy Disease Severity Scale**

<b>Disease Severity Level</b>	<b>Observable Findings Upon Dilated Ophthalmoscopy</b>
No apparent retinopathy	No abnormalities
Mild NPDR	Microaneurysms only
Moderate NPDR	More than just microaneurysms but less severe NPDR
Severe NPDR	Any of the following: <ul style="list-style-type: none"> <li>• More than 20 intraretinal hemorrhages in each of four quadrants</li> <li>• Definite venous beading in two or more quadrants</li> <li>• Prominent Intraretinal microvascular abnormalities (IRMA) in one or more quadrants</li> </ul>
PDR	One of either: <ul style="list-style-type: none"> <li>• Neovascularization</li> <li>• Vitreous/preretinal hemorrhage</li> </ul>

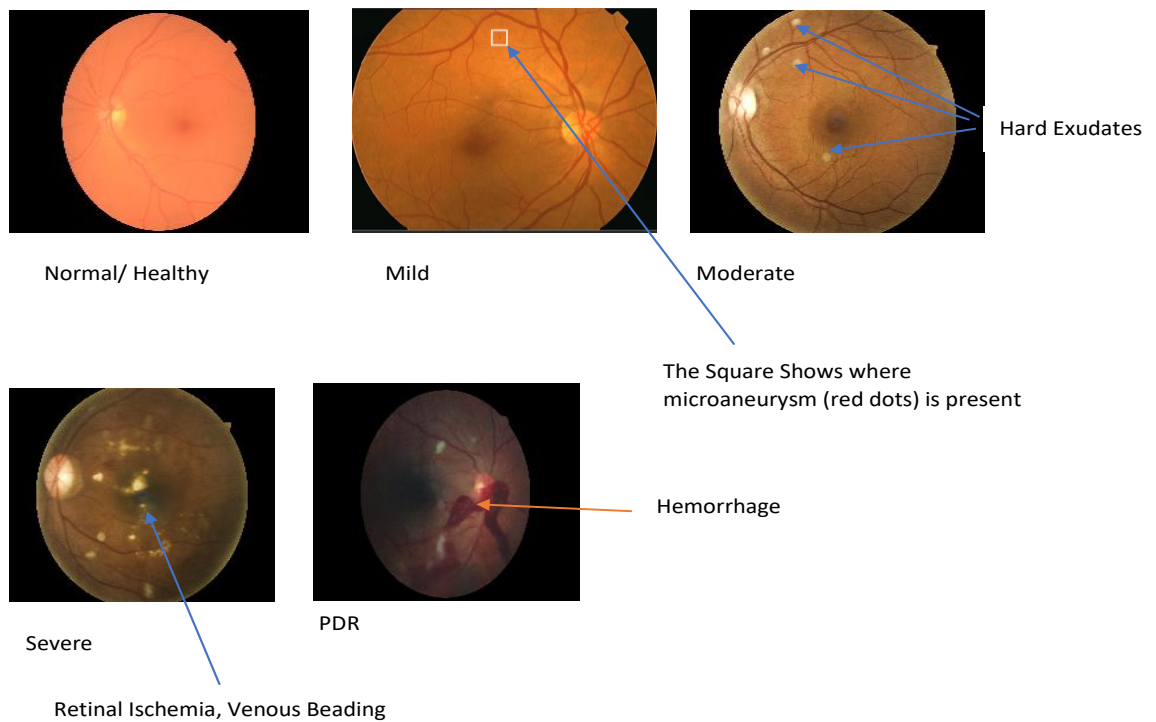
Computer vision involves analyzing fundus images to extract critical features that distinguish one stage of diabetic retinopathy from another. These features for each stage of diabetic retinopathy include the following: Level\_0 healthy: the fundus images do not show any signs of diabetic retinopathy. Level\_1 Mild Diabetic Retinopathy: Microaneurysms which are red dots as a result of capillary dilation[23] [20]. Level\_2 Moderate diabetic retinopathy: More than five Microaneurysms, cotton

wool spots, and hard exudates. Hard exudates consist of lipid and proteinaceous materials that leak from the impaired blood-retinal barrier[23][20].

Level\_3 Severe Diabetic Retinopathy: Venous beading which is the irregular constriction or dilatation of venules in the retina. More than 20 intraretinal hemorrhage and formation of shunt vessels which appear as abnormal branching of blood vessels[23][20]. Level-4 Proliferative Diabetic Retinopathy:

Neovascularization which is the formation of abnormal blood vessels, Vitreous hemorrhage which is the bleeding that appears in the gel like part of the eye[20][23].

Figure 2.1 shows samples of eyes fundus images for each stage of diabetic retinopathy[23][29]. The arrows in the images point to the specific features that differentiate between the various classes of diabetic retinopathy.



**Figure 2.1: Fundus Images showing Stages of Diabetic Retinopathy (Source; [23] [29])**

Artificial intelligence and more specifically machine learning and deep learning have been applied in automated detection of diabetic retinopathy such as in [30] [31] [32] [19][28]. Automated detection of diabetic retinopathy involves training a neural network using labeled fundus images. Automated DR detection models are more powerful than human beings since they are able to classify multiple images within a short period of time, and without exhaustion. They are also scalable and can improve their classification accuracy, specificity, and sensitivity over time[33].

According to a study conducted by [26] diabetic retinopathy is prevalent in Kenya especially among people with diabetes mellitus. This poses a challenge to many Kenyans since a study conducted by [26] revealed that there is no good integration of diabetic retinopathy care in government policy. This makes it difficult for patients to manage diabetic retinopathy in Kenya. The study recommends that the government can make innervations through policies such as: first, integrating diabetic retinopathy with other top priority diseases in the country such as HIV, Malaria, and Tuberculosis[26]. Second, the costs of tests and medicine for DR to be included in the National Hospital Insurance Fund cover. Third, maintain an integrated electronic health information system to easily survey the prevalence of DR[26].

Nkumbe et al [34] in a study to determine prevalence of diabetic retinopathy in Kenya records that; out of 171 eye images obtained for the study, only 92 eye fundus images were in good quality for analysis. The poor quality of fundus images was as a result of accidental opening of the fundus camera by clinicians as well as media opacities[34]. Therefore, there is need for the government through the ministry of health to facilitate the collection of fundus images so that automatic diagnosis of the disease can be possible.

## **2.3 Empirical Studies**

This section looks at previous works that have been done in the use of transfer learning and deep learning models for detection of diabetic retinopathy. The section also analyzes the techniques that the researchers used, the results they obtained, and the outstanding gaps in their work. The section forms part of objective one of this research.

### **2.3.1 Detection using Convolutional Neural Network**

Thiagarajan et al.,[31] proposed a model that uses Convolutional Neural Network (CNN) to detect diabetic retinopathy. The researcher records that the motivation towards the use of CNN was the automatic feature selection capabilities of CNN. The study used the Indian Diabetic retinopathy dataset which consists of 413 training and validation images (80% and 20% respectively) and 103 testing images. Data pre-processing involved loading the images using opencv and resizing them to 256\*256 dimension [31].

Data augmentation was done using keras and it involved horizontal flipping, scaling, zooming in, cropping, and translation. Model training involved a CNN with convolution, dropout, and max pooling layers. The study used batch normalization preceding activation layers, ReLU, Sigmoid scaled Exponential Linear Units (SeLU), and softmax were used as the activation functions. Binary cross-entropy was used as the loss function, and a series of optimizers. Grey-level Co-occurrence matrix was used to extract features in basic machine learning algorithms and managed to gain an accuracy of 48.78% with logistic regression [31]. The highest accuracy gained by the model was 80.036% with 4 layer Convolutional Neural Network (CNN), Adam optimizer, and softmax activation function, learning rate of 0.001, and binary cross entropy loss function[31].

Although the study demonstrated that deep learning performs better than ordinary Machine learning in this task. The model is limited in that it performs binary classification yet diabetic retinopathy classification is a multi-class classification task with five classes. This means that the model can only classify fundus images as either DR+Ve or DR-Ve. Thus it is unable to classify the DR+Ve fundus images into their respective classes which are either Mild DR, Moderate DR, Severe DR, or Proliferative DR. Also, the researcher records that automatic hyperparameter tuning can be done using meta-learning to achieve better performance [31]

### **2.3.2 Using Mask RCNN and Transfer Learning**

Shenavarmasouleh et al.,[32] proposed a hybrid DRDr II and recurrent neural network for early detection of diabetic retinopathy. The research used a public dataset from kaggle which had more than 35000 fundus images. A pretrained model of DRDr was used to perform transfer training. Data pre-processing was done using opencv. It involved cropping the images, removing extra blank space around them. The eyes were then transformed into perfect circles, gaussian blur was used to normalize the contrast level on each image [32].

In data preprocessing the researchers grouped the top two most categories together and the bottom two categories together to create two three classes rather than the initial five classes. The researcher records that this was informed by the fact that there is close mapping between these categories of data. The researchers used a feed forward neural network with 2 hidden states with 75 nodes, softmax activation function, adam optimizer with a learning rate of 0.001, and 100 training epochs. The model achieved an accuracy of 92% [32].

Although this model achieved a good accuracy, the model did not achieve the goal of classifying fundus images into the five classes of Diabetic Retinopathy since the researcher combined the first two and the last two classes thus resulting into a multi-class classification of three classes rather than five classes. Secondly, the model did not perform inner-loop optimization thus the results achieved by this model are not optimal. Third, this model used an imbalanced dataset, therefore, the accuracy obtained is not reliable other evaluation measures such as Recall, F1-score, and precision would have been used. The researcher would have also considered reducing the data imbalance gap. The limitations in this model make it unfit for classification of diabetic retinopathy since it cannot separate between mild and moderate classes as well as severe and proliferative diabetic retinopathy. This inefficiency may greatly impact disease management process.

### **2.3.3 Convolutional Neural Network**

Pratt et al [35] proposed a CNN model that aimed at automatically detecting and classifying diabetic retinopathy. The network architecture comprised of 10 convolutional blocks and three fully connected layers. Each convolutional layer had a max pooling layer and batch normalization. The first layers learn edges while the last layers learn classification features such as exudate. The researchers[35] also used weighted class weights relative to the number of images in each class to avoid overfitting. Drop out was also used in the dense layers while L2 regularization was used in the convolution layers[35].

The researchers used a dataset with 80,000 images obtained from Kaggle out of which 5000 images were used for validation. Further the researcher used Nvidia GPU which made it possible for the researchers to train faster and with more images[35]. Data pre-processing involved color normalization, data augmentation, and resizing the images

to 512\*512 pixels. Training involved first pre-training the network using 10,290 images for 120 epochs and then training with 78,000 images for 20 epochs. Stochastic gradient descent with Netsrov momentum was used. The pre-training phase took 350 hours to achieve an accuracy of 60%[35].

The model achieved an accuracy of 75%, specificity 95%, 30% sensitivity,[35] and F1-score 41.6%[17]. The high specificity of the model was a tradeoff of lower sensitivity. Specificity is the total number of patients identified as not having diabetic retinopathy out of total number of instances without diabetic retinopathy. Sensitivity on the other hand is the total number of patients identified as having diabetic retinopathy out of total number of instances with diabetic retinopathy. Accuracy is class-wise classification[9].

Although the model automated diabetic retinopathy and achieved high specificity it has several drawbacks. First, the high specificity and low sensitivity shows that the model was affected by class imbalance since it focused more in the negative class which had more instances. This shows that the model has 70% chance of misdiagnosing a patient as negative while as the patient is positive thus negating the need for automation. Second, the model still achieves relatively low accuracy since it has 25% chance of misclassifying a patient. Third, the model took long to train in the initial phase, transfer learning would have addressed this problem. Fourth, the model uses SGD which is prone to poor generalizability in global and local minima's.

#### **2.3.4 Transfer Learning and Small Dataset**

Hagos and Kant [19] developed a transfer learning model that was aimed at solving the problem of shortage of large amounts of labeled data that is usually required for deep learning. The study used a pre-trained Inception-V3 model to take advantage of

its inception models. The researchers used keras library to import the pre-trained model and then added a classifier to classify the fundus images as healthy or unhealthy. A dataset consisting of 2500 images was used for training and testing [19].

Image pre-processing was done which involved resizing and cropping the fundus images. Cropping was done using opencv in python and the images were resized to 300\*300 pixels. The researchers used the pre-trained part of the Inception V3 to do feature extraction [19]. Classifier training was done on top of the pre-trained model using the relu-activated layer, stochastic gradient descent with an ascending learning rate of 0.0005, and softmax output function. Cosine loss function was used to calculate error since it has delivered good performance in small datasets. The model was tested using 500 unseen fundus images and it achieved an accuracy of 90.9% and a loss of 3.94% [19].

Although this study tried to solve the deep learning problem of requiring huge datasets by adopting transfer learning which is an aspect of meta-learning, the model only did binary classification which classifies the fundus as either healthy or unhealthy. This is not efficient because, diabetic retinopathy is classified into five categories which are zero, Mild, Moderate, Severe, and Proliferative diabetic retinopathy. Also, the model used stochastic gradient descent which converges in the local minima and fails to converge in the global minima. The model achieved a very high loss of 3.94% which is not desirable.

### **2.3.5 Meta-Learning with Noisy Data**

Algan et al.,[28] proposed a label-noise-robust learning algorithm that uses meta-learning paradigms. The main aim of the model was to utilize meta-learning techniques to learn from noisy labels and achieve high accuracy. Training involved



three levels namely: pre-training, meta-training and conventional training. Pretraining involved using a ResNet50 architecture with model parameters pretrained on Imagenet dataset [28]. Further pre-training is done using Diabetic retinopathy dataset. In meta-training a multi-layer perceptron neural network is trained with an aim of seeking soft labels for each noisy data. Stochastic gradient descent, KL-divergence loss function, back propagation, and cross-entropy loss function were used [28].

The study used primary data with 1947 images. 200 images were used for training as correctly labeled images, 1447 were used for training as noisy-label images, while 300 were used for testing. Data pre-processing involved resizing the images to 256\*256 and the center crop to 224\*224. Random horizontal flip was used for data augmentation. The model achieved a training accuracy of 86.3% and testing accuracy of 91.4% [28].

The model performed well in dealing with noisy-label dataset, however, the model uses stochastic gradient descent optimizer which increases the variance because it only uses one example for each learning step. Also, in each iteration of the SGD, the learning step wanders around the minimum region without convergence since the noise makes it to go back and forth. This highly affects the generalizability of the model. Also, the model does binary classification of images as either healthy or unhealthy but does not classify the images into the various classes of Diabetic Retinopathy. This means that the model can only tell if a person has diabetic retinopathy or not. However, if the patient is +ve the model cannot determine the severity of DR.

### **2.3.6 A deep Learning Ensemble Approach**

Qummar et al [17] developed a model that involved an ensemble of five Transfer learning models namely: ResNet50, InceptionV3, Xception, DenseNet121, and DenseNet169. The aim of the model was to encode the rich features involved in detection of diabetic retinopathy as well as improve the classification accuracy. Qummar et al [17], noted that previous works in this area were challenged in they did not consider data imbalance and the meta-learning step of hyperparameter tuning and its implications. The researchers used the Eyepacs dataset which is publicly available Kaggle.

Data pre-processing involved resizing the images to 786\*512, then randomly cropping them to 512\*512. In order to balance the data first, up-sampling was done through augmenting images in the minority classes. Second, down-sampling was done which involved removing some instances in the majority to match with minority class sizes of 708 images per class. After down-sampling the dataset is divided into training, validation, and testing sets using the ratios of 64%, 20% and 16% respectively[17].

The researchers used Adam learning rate and the Nesterov-accelerated adaptive moment estimation to update the learning parameters. The learning rate is initially set 0.001 and then decreased by a factor of 0.1 to 0.000005 for 50 epochs with early stopping to avoid overfitting[17]. The researchers used stacking to combine the results of all different models and generate unified results. Qummar et al [17] records that in the case of imbalanced data, accuracy is a misleading metric since it is biased towards the majority class. To address this the researchers considered other metrics as F1-score, precision, recall, and ROC curve[17].

The model achieved the following results: The imbalanced dataset had: Accuracy 80.8%, Recall 51.5%, specificity 86.7%, precision 63.85%, F1-Score 53.74%[17]. The results of the down sampled data were: Accuracy 58.08%, recall 58.10%, precision 70.3%, specificity 85.5%, F1 score 53.64%. The model also achieved an average Area Under the Curve (AUC) of 0.91. This was computed by averaging Micro AUC (0.95) and Macro AUC (0.87). It was noted that the performance of the model increases with decrease in the learning rate. Also, SGD registered better performance than Adam[17].

The major drawback of the ensemble model is the high number of learnable parameters which increases computational cost and reduces the efficiency of the model [23]. Also, the model is not yet optimal since it performed relatively low in all the measures compared to the model developed in this research as shown in Table 4.8. When training cost increases the model becomes infeasible in hospital set-up especially in developing countries since those hospitals would require HPCs to operate the model.

### **2.3.7 Ensemble Framework of Deep CNNs**

Jinfeng et al [36] proposed a transfer learning model that aimed at addressing the challenges of imbalanced data in detecting diabetic retinopathy. The key motivation behind this research was that training deep neural network with imbalanced data results to biasness in classification. The researchers used the EyePacs Diabetic retinopathy dataset available in Kaggle repository[36]. Data pre-processing involved the following: resizing the images to 786\*512, randomly cropped patches of 512\*512 to avoid training overhead, mean normalization of each image, then the data imbalance is solved through up sampling[36].

Up sampling involved augmenting minority classes through 90 degrees rotation and flipping. The balanced dataset is then divided into training, validation, and testing sets in the ratios of 64%, 16% and 20% respectively. The balanced dataset has 129050 samples from which equal batches of size 27530 are generated from the training dataset[36]. The researchers then used two approaches: Model 1 is an ensemble DesnseNet121 trained on the three batches. Model 2 is a bagging ensemble of DesnseNet121, ResNet50, and Inception V3. All the models are pre-trained on ImageNet dataset and then fine-tuned using the three datasets[36].

The results of the study were as follows: the accuracy of the balanced data and imbalanced data in the first model were 60.80% and 78.13% respectively. However, class-wise results demonstrate that the balanced data has better performance compared to the imbalanced dataset[36]. The model achieved an average AUC of 0.895 in the imbalanced dataset and 0.87 in the balanced dataset. The overall results achieved by the two models are as follows: Model 1 recall 44.85%, specificity 85.48%. Model 2 accuracy imbalanced dataset 80.36%, Accuracy balanced dataset 60.89%, recall 47.70% and specificity 85.94%[36].

The model developed Jinfeng et al [36] is limited in the sense that it has achieved lower performance in the balanced data which goes against the hypothesis of the research, which was that data imbalance impairs performance. The model tends to perform poorly with large dataset. Also, the model is not optimal since its performance is lower than that of a previous model by Qummar et al [17] and also lower than the performance achieved by the improved model in this research as shown in Table 4.8.

### 2.3.8 Deep Learning Approach

This model was proposed by Tymchenko et al.,[30] it uses deep learning approaches that uses an ensemble of three convolutional neural network architectures (EfficeintNet-B4, EfficientNet-B5, and SE-ResNext50) and few shot-learning to detect deiabetic retinopathy. The model also proposes a transfer learning appraoch that makes use of similar datasets with different labeling. The following datasets were used in the model; EyePACs 2015, Indian diabetic retinopathy image dataset, MESSIDOR dataset, and APTOS 2019 blideness detection dataset.

Data preprocessing which involved image cropping and resiszing was done. High amount of agumanetation was used to prevent CNN from overfitting due to correlation between image conents and its features [30]. The reserchers divided the training phase into three phases; first pre-training where the model is trained with minibatch-Stochastic Gradient Descent, cross-entropy loss function and cosine-annealing learning rate. Second, the main training phase which used focal loss function, rectified adam optimizer and cosine-annealing learning rate schedule. Thrid, the post training phase which focused on fitting the regression part of the model. The model achieved a quadritic weighted Kappa score of 92.5% [30].

The main challenge with this model is that although it has tried to achieve few-shot learning and transfer learning which are part of meta-learning, it does not perform two-level optimization. Secondly, the model has not provided other evaluation measures that are necessary for evaluating model's perfomance. The resercher proposes as part of future work that meta-learning and more accurate hyperparameter optimization can be done.

### **2.3.9 Detection Using VGG-NIN a Deep Learning Architecture**

Khan et al [23] proposed a model that focuses on classifying diabetic retinopathy to different stages with the lowest learnable parameters to speed up training and enhance model convergence. To achieve this the researchers proposed stacking of the VGG16, spatial pyramid pooling layer and network-in-network (NiN) to create a highly non-linear scale-invariant model. The proposed model VGG-NIN is capable of processing DR image at any scale due to the SPP layer as well as achieve higher accuracy due to the NiN layer. The researchers used the labeled EyePacs dataset which consisted of 35126 images belonging to five classes[23].

Data pre-processing involved the following: resizing the images to size 1349\*1024 and then random cropping to size 512\*512 and data augmentation were done. The spatial pyramid pooling layer was embedded between the last convolutional layer and the first fully connected layer. The SPP layer performs information aggregation and thus avoids loss of features as a result of cropping and resolution adjustment. The Nin layer is then added on top of the SPP layer. Parametric Relu (PReLU) is used to compute the activation function. PReLU reduces overfitting by learning the rectifier parameter adaptively.

During training, the convolutional layers of the VGG16 model are frozen but the fully connected layers and the NiN layers are fine-tuned. The model achieved the following results: Micro AUC 95.0, Macro AUC 84.0 (weighted AUC 89.5), overall accuracy 85%, overall recall 55.6%, Overall precision 67%, overall specificity 91%, overall F1 score 59.6%.

The model is challenged in that it performed very poorly in recall, precision, and F1.Score. The low performance can be attributed to the model's complexity brought

about by the added network in Network layer. Ideally a classification model should aim to have a high recall. The model architecture and training process needs to be modified so that it can overcome the drawbacks of the Network in Network layer. Modification would result into a more robust model that would achieve better performance compared to what this model by Khan et al [23] has achieved.

### **2.3.10 Transfer Learning from Pre-trained CNN Models**

Chilukoti et al [18] proposed transfer learning models for diabetic retinopathy detection. The researchers used pre-trained ResNet50, VGG16, and EfficientNetB3 and fine-tuned them to fit the Eyepacs Kaggle dataset. The main goal of this study was to develop a model that is able to detect all stages of diabetic retinopathy. The researcher records that EfficientNetB3 achieved the highest quadratic weighted kappa metric of 0.85. The dataset was divided into three sets namely: training with 24590 retinal images, the validation set and test set had 5268 images. The images were resized to 150\*150 resolution[18].

All the pre-trained models were frozen during the training and validation phases and only the final classifier is trained. Initially all the models were trained for 30 epochs and the best model which was EfficientNetB3 was selected and retrained for another 30 epochs while saving the weights[18]. The researchers used metrics such as accuracy, precision, recall, F1-score, and quadratic weighted kappa metrics. Adam optimizer was used as the optimizer of choice, gradient clipping of value 0.1 to avoid exploding gradients. A weight decay value of  $10^{-4}$  was used therefore the learning rate for all the models was set at 0.001[18].

The top layer of the model was replaced as follows in a sequential manner: flattening layer, fully connected layer, dropout with a value of 0.5, full connected layer, dropout

of 0.25 layer, and finally the output layer. All the layers have Relu activation function. The results of the best model against the performance metrics were: Accuracy 0.87, F-1 score 0.84, precision 0.85, Quadratic weighted kappa metric 0.85, and Recall 0.87 [18].

Although, this model has achieved higher performance compared to previous models such as Qummar eta al[17] and Khan et al [23] , the model did not leverage on the power of self-optimization as well as weight sharing in the convolution block.

### **2.3.11 Transfer Learning Based Robust Automatic Detection**

Charu, Jain, and Sood [27] proposed a model for grading diabetic retinopathy using transfer learning and dynamic investigation. The model uses pre-trained neural network architectures for feature extraction, prominent feature transfer learning approach (PTFL), and then transfers the features to support vector machine for classification. The transfer learning architectures used in this case are AlexNet, GoogleNet, ResNet, VGGNet, and Inception. The datasets used in this study were the Messidor dataset and the IDRiD dataset [27].

The researchers conducted a series of experiments aimed at determining the best model for solving the task at hand. The results demonstrated that InceptionV3 performed best when combined with PTFL approach utilizing statistical feature reduction approach since it achieved an accuracy of 90.51%, sensitivity of 89.42%, specificity of 90.78%, precision of 91.95%, F1-score of 89.81%, and Area under the curve (AUC) of 0.91 [27].

Although this study was able to achieve high performance by leveraging on transfer learning it is limited. This is so because, the study uses Messidor dataset which classifies diabetic retinopathy in 4 classes namely: Healthy, Mild, Moderate, and



Severe thus it omits the proliferative diabetic retinopathy class[27]. Therefore, this model is not suitable for diabetic retinopathy classification since it will wrongly grade people with proliferative Diabetic retinopathy. This is especially critical where special care is required for PDR patients.

#### **2.4 Machine Learning Model Development Approaches**

Machine learning development is different from standard software development. Unlike standard software development which has explicit features, machine learning revolves around experimentation[37]. Machine learning aims at extracting patterns from data. Therefore, machine learning developers experiment on different datasets, algorithms, third party libraries, and parameter with an aim of optimizing business performance metric such as accuracy. The performance of a model is dependent on the training data and the training process, therefore, reproducibility of machine learning models is paramount[37].

Machine learning model development involves a common pipeline that is applied not only in standard machine learning algorithms but also in deep learning and its variants. The pipeline's aim is to use data as input and generate machine learning models that are capable of achieving high performance in unseen data[11]. The pipeline also known as machine learning workflow involves the following broad stages: Data management, Model learning, and Model verification[11]. Each stage in this workflow involves sub-activities and processes which differentiate the various variants of machine learning and deep learning. Researchers in the field of machine learning work on optimizing this workflow to achieve more optimal and powerful model[37].

### **2.4.1 Data Management**

Data management is a broad term in machine learning that involves several activities such as: acquisition of the data, pre-processing of the data, and feature extraction from the data. Data acquisition refers to getting the appropriate data that is needed to train the model. Data can be of two types namely primary data which is collected by the researcher or secondary data which is obtained from a repository[38]. Primary data is mainly used in situations where secondary data is not available. Primary data provides researchers the ability to collect data that is much oriented to the task the model is intended to solve. However primary data is expensive to acquire. Secondary data is the most commonly used in many models since its readily available in public repositories [39][38].

Data pre-processing is a crucial step in machine learning workflow that aims at converting data from its raw form to a form that is ready for training a machine learning model[40]. Data pre-processing also aims at reducing the dimensionality of data by eliminating unnecessary elements from the data. This makes it easy for a machine learning model to learn from the data with ease. [41] argues that the specific activities involved in data pre-processing depend heavily on the nature of the data, the algorithm used, the library used, and the nature of the task or expected output. In image processing data pre-processing involves aspects such as resizing, data augmentation, channel conversion, among others[42].

Feature extraction is mainly used in image processing, natural language processing, and video processing tasks. In image processing feature extraction refers to obtaining distinctive features from image that differentiate an image of one class from another. It is useful in machine learning since it makes it possible for developers to focus on necessary features from an image without losing critical aspects of the image[43].

Further to this, feature extraction also helps in automatic exploratory data analysis as well as making data visualization to be more convenient[44]. Feature extraction can be grouped into two main categories namely low-level feature extraction and high-level feature extraction.

#### **2.4.1.1 Low-level feature extraction**

This involves extracting features that fall into the following categories: First color features, which involve using color to differentiate different images. Some of the techniques under color feature extractors include; “color histogram, color moments, color coherence vectors, color correlogram, Average RGB, scalable color descriptor, color structure descriptor, and dominant color descriptor”[44]. Secondly, texture features which involve computing texture of an image from the pixels. The techniques under texture feature extractors include: Grey Level co-occurrence matrix, steerable pyramid, contourlet transform, Gabor wavelet transform[44].

Third, shape features which focuses on the shape of an image. Shape is considered to be a primitive feature for image content description[44]. Shape features can be grouped into two categories namely region-based methods and contour-based methods. Region based methods use the whole area of an image for shape description. Contour based methods are the ones that use contour information only. The techniques under shape features include: “Geometric moments, Algebraic moment invariants, Zernike moments, Fourier descriptor, Region-based Fourier descriptor, and Wavelet transform”[44].

#### **2.4.1.2 High-level feature extractors**

These ones focus on extracting features that are specific to a particular set of images. They are characterized with high levels of abstraction and are mostly used together

with low-level feature extractors[45]. High-level feature extraction is used in object detection since it helps a lot in differentiating one object from another. Convolutional neural networks have proved to be powerful feature extractors for both low-level features and high-level features[45].

#### **2.4.1.3 CNN feature extractors**

Unlike standard machine learning algorithms, CNNs do not require a separate feature extractor since the CNN layers have the ability to extract features from the data. A typical convolutional neural network architecture usually consists of the convolution layer, activation layer, pooling layer, and fully connected layer[46]. The convolutional layers have convolutional kernels with each kernel extracting certain features. The initial layers start with extracting basic features such as edges and shapes. The complex and specific features are extracted at the last layer[46].

Convolutional neural networks have proved to be more powerful than other machine learning algorithms since the inbuilt feature extraction capabilities eliminate the need for hand crafted features. Pre-trained CNN networks usually use their last output layer to extract high-level feature from an image. CNN uses 1D and 3D convolutional filters to extract spectral and spatial-spectral features respectively[46]. On the other hand 2D filters are used to analyze colored as well grey-scale images. Features extracted from CNN 1D, 2D, and 3D can be used to train other classifiers such as SVM, decision tree, and random forest[46].

#### **2.4.2 Model Training**

Model training is where actual machine learning takes place and it involves, selecting an algorithm, choosing hyperparameters, and training the algorithm using the training dataset[11]. This is a very critical stage since the algorithm chosen should be suitable

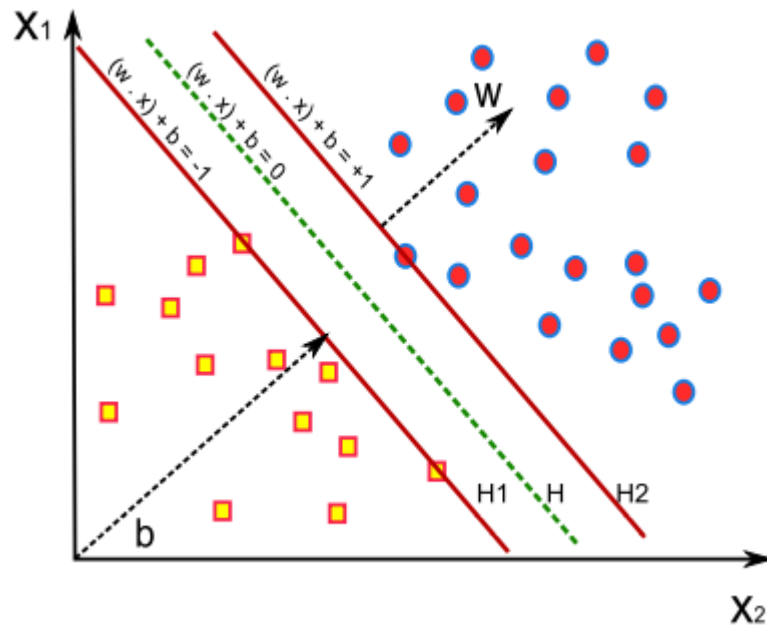
for the task to be solved. There are algorithms that are robust such that they can perform both classification and regression while there are others such as K-means clustering that are purely for clustering purposes. Some of the popular image classification algorithms include; Decision tree classifier, Support Vector Machine (SVM), Random Forest classifier, K-Nearest Neighbors (KNN), and Artificial Neural Networks, among others.

#### **2.4.2.1 Support Vector Machine**

Support Vector Machine (SVM) is both a regression and a classification algorithm which was introduced by Vapnik as a kernel based machine learning algorithm[47]. SVM has demonstrated prowess in solving binary classification tasks due to its theoretical foundation. SVM focuses on creating a decision boundary known as a separation hyperplane between two classes. The separation hyperplane is oriented in such a way that it is furthest from the closest data points of each class. These closest points of each class are known as support vectors. An ideal separation hyperplane should be in such a way that it avoids misclassification and at the same time maximizes the distance from the hyperplane to the support vectors[48]. Figure 2.2 show a linear SVM classifier.

H1 and H2 represents parallel hyperplanes, the goal is to maximize the distance between these two parallel hyperplanes. While maximizing this distance some data points known as support vectors may fall on H1 and H2, these will represent the maximum level at which the distance can be maximized[47]. An optimal hyperplane is now introduced between the two, this optimal hyperplane is H. From the figure 2.2 it is evident that H avoids marginal error by maximizing the distance between H1 and H2 it has also avoided misclassification error since no data point has been

misclassified.  $w$  represents a weight vector perpendicular to the optimal hyperplane,  $b$  represents the bias, while  $X_1$  and  $X_2$  represent data points [47].



**Figure 2.2: SVM Linear Classifier (Source; [47])**

#### 2.4.2.2 K-Nearest Neighbors

It is a machine learning algorithm that can be used for classification as well as regression and it falls under the category of supervised learning algorithms. K-Nearest Neighbor classifier classifies each element based on its  $K$  nearest neighbors. The algorithm works on the assumption that similar things exist in close proximity[49]. Therefore, it determines the correct class of a test sample by calculating the distance between the test sample and the training samples. Choosing the right  $K$  value is critical in the performance of KNN. Therefore, in most cases researchers test with different values of  $K$  until they find one which offers better performance in the data set being used[49]. KNN is simple to implement but it is unreliable when the number of classes as well as number of samples increase.

### **2.4.2.3 Decision Tree**

Decision tree is a supervised machine learning algorithm that can be used for both classification and regression. For classification tasks, a decision tree classifier is used. Decision tree algorithm consists of node and branches which make up a tree. Nodes represent features in a category to be classified[50]. The intuition behind decision trees is that you use dataset features to create binary question and continue splitting the dataset until all data points belonging to a particular class are isolated. The first node in a decision tree is known as the root node while the last node is known as the leaf node[51].

Two main variables namely entropy and information gain affect decision trees. Entropy refers to the measure of dataset randomness, it ranges from 0 to 1, the higher the entropy the more complex the decision tree gets and the worse it performs. Information gain refers to how much information a feature provides about a class. Therefore, unlike entropy the higher the information gain the better the performance[50]. Decision trees are powerful especially for binary classification. However single-tree model is possibly sensitive to specific training data and easy to overfit[52]

### **2.4.2.4 Random Forest**

Random forest is an ensemble of multiple decision trees such that each tree depends on a random independent dataset and all trees in the forest have the same random distribution[52]. The capacity of random forest is dependent on two main factors namely: the individual strength of each tree and the correlation between the trees. Better performance is achieved when the strength of a single tree is high and the correlation between the trees is low[52]. The trees may not necessarily have equal capacity which is as result of the randomness of the trees. Random forest addresses

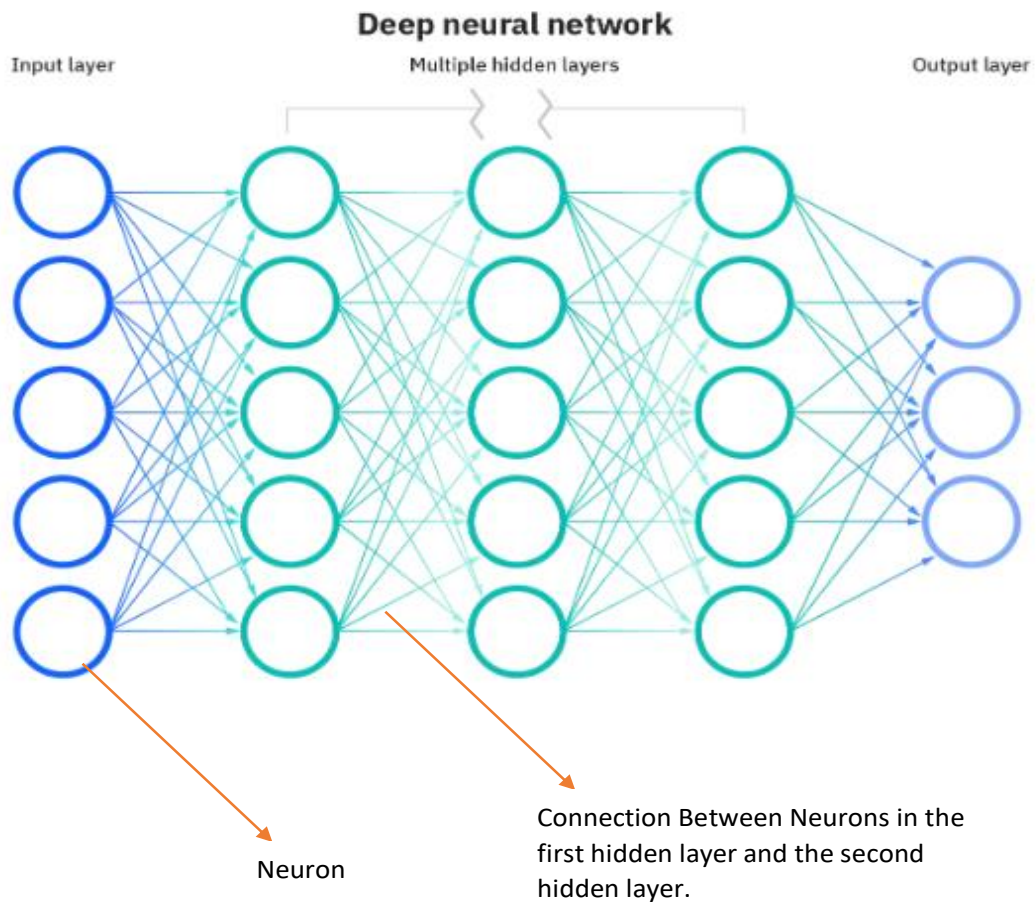
the challenges of decision trees, thus making it to be more robust than decision trees. This is because random forest reduces variance when averaged over trees and also decorates the trees[53]. However, random forest classifiers in Image classification are dependent on hand crafted features.

#### **2.4.2.5 Artificial Neural Networks**

Artificial Neural networks simulate the behavior of human brain allowing computer-based programs to solve complex problems. Artificial Neural networks are at the heart of deep learning which is a branch of machine learning[54]. Deep learning refers to a neural network with multiple complex layer[55]. The general structure of an Artificial Neural network involves the input layer, the output layer, one or more hidden layers, then each of these layers have neurons which are also referred to as nodes[55]. Each neuron connects to another neuron and has an associated weight, threshold, and an activation function. If the output of a certain neuron is above the threshold then the activation function activates the neuron, thus, allowing the neuron to pass data to the next layers. If the threshold is passed no data is passed to the next layer[54][55].

Figure 2.3 shows a deep neural network with three hidden layers with each having five neurons, an input layer with five neurons, and an output layer with three neurons[54]. The number of neurons in the output layer for classification task is dependent on the number of classes that the network classifies data into. Thus, a binary classification neural network has two neurons in the output layer while a neural network that is supposed to classify data into three classes has three neurons in the output layer as shown in Figure 2.3.





**Figure 2.3: Deep Neural Network (Source; [54])**

Neural networks have gained a lot of popularity in the recent past mainly due to their high-performance capabilities across different sets of tasks[56]. Neural networks can be grouped broadly as either feed forward neural networks or feed backwards neural networks. Feed forwards neural network propagate information from one layer to another until it gets to the output layer. Feed backward neural networks on the other hand use internal state memory to process sequence of data[55]. Deep learning is the main area of research by many researchers such as [31][57][8]in neural networks mainly due its wide scope of application and capabilities.

Deep learning, which involves training deep neural networks has evolved over time as researcher aim at improving its capabilities. This has resulted to more powerful neural networks that can perform tasks that could have otherwise been difficult to perform using ordinary machine learning algorithms[58]. Convolutional neural networks are the most used deep neural network in image processing tasks[59]. They are considered to be essentially powerful due to their capabilities to extract both low-level and high-level features from images [46].

However, Deep neural networks require a huge amount of data and high computational resources to train a model. Also, deep learning has a wide range of hyperparameters that need to be tuned[60]. Therefore, researchers in this field have focused on optimizing the training process through hyperparameter tuning as well as enhancing the capabilities of the deep neural networks. One major solution to these challenges is an advancement of deep learning known as Meta-learning[13][28].

## **2.5 Meta-Learning**

Meta-learning is also referred to as learning-to-learn. It is a field of Artificial intelligence that aims at improving the conventional machine learning and deep learning approaches where tasks are performed from scratch using a fixed learning algorithm. The aim of Meta-learning is to improve the algorithm. This provides an avenue of solving the key machine learning challenges which include data, performance, generalization, and computational bottlenecks [14]. According to [4] the goal of Meta-learning is to train a model with a diversity of tasks so that the model can easily solve new but related learning tasks with fewer training samples. This is achieved through transfer learning.

[14] records that even though deep learning techniques which were considered to be an improvement in machine learning have achieved great success, they have great drawbacks in the sense that their success is pegged on the availability of huge volumes of data and high computational power. This excludes some application areas where huge volumes of labeled data are either unavailable or expensive to obtain.

According to [4], learning quickly is the hallmark of human intelligence, this may involve learning new skills quickly or using previous knowledge to learn new things faster. Meta-learning imitates this human intelligence by creating a model that leverages on previous knowledge to learn a new task [4]. The meta-learning framework enables models that have been developed using meta-learning approaches to be less costly to train and more robust than those developed using deep learning through learning from few shots [61].

Finn, Abbeel, and Levine demonstrate that with a task-agnostic algorithm for meta-learning, a model can be trained in a way that few gradient steps can result to quick learning of new tasks [4]. Meta-learning can be framed as an optimization procedure in which the inner level represents an adaptation to a particular task while the outer level represents the meta-training objective. This formulation can result in understanding the optimal parameters specific to a model and also suitable for generalization [62]. According to [63] Meta-Learning can be viewed in two approaches: First, mechanistic view, this involves training a meta-learning model with meta-data. Secondly, the probabilistic view, this involves extracting prior information from a set of Meta training task that allows efficient learning of new tasks through transfer learning.

Equation 2.1 below is the Meta-learning equation for setting the meta-learning dataset to obtain meta-train task in a multi-task setup.

$$D_{meta-train} = \{(D^{tr}_1, D^{ts}_1), \dots, (D^{tr}_n, D^{ts}_n)\}.$$

Equation 2.1

Where:

$D_{meta-train}$  represents the meta-training data,

$D^{tr}$  represents training subset

$D^{ts}$  represents the testing subset.

This data is used to learn function  $\theta^*$ , representing a global meta-learning parameter that will contain the information needed to solve a new task. Equation 2.2 represents the Meta-learning Equation which is used to obtain  $\theta^*$  [63]:

$$\theta^* = \arg \max_{\theta} \log p(\theta | D_{meta-train})$$

Equation 2.2 (Source; [4] [61])

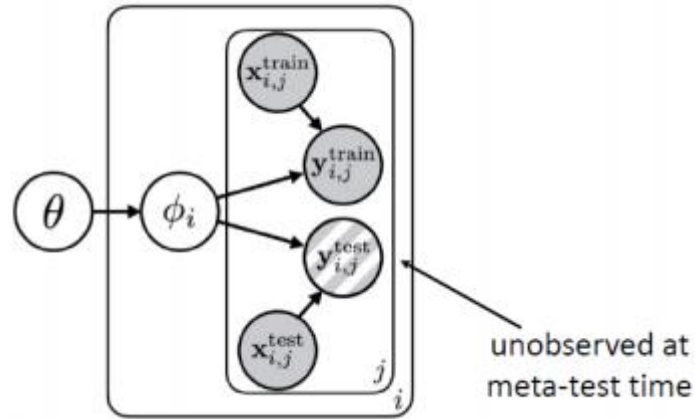
From this equation it is evident that the Meta-Learning problem is to obtain the right  $\theta^*$  starting from the initial model parameters  $\theta$  and the meta-training data.  $\theta^*$  contains everything that model needs to know in order to solve a new but related task. This is called outer loop optimization [61].

To achieve an efficient  $\theta^*$  the parameters of  $\theta$  needs to be fine-tuned on the training task of each meta-learning task. This results in the adaptation equation 2.3.

$$\varphi^* = \arg \max_{\varphi} \log p(\varphi | D^{tr}, \theta)$$

Equation 2.3 (Source; [13])

representing the set of the parameter of the specific task, reached through a meta-optimization problem from  $\theta$ , called inner loop optimization [61]. The key idea is to use  $\phi^*$  as model parameters to assess how the optimization of  $\theta$  is good for classifying new unseen data points, represented by the test part of each meta-task as shown in figure 2.4.



**Figure 2.4: Meta-Learning (Source; [61])**

Figure 2.4 demonstrates that for every meta-task,  $\phi^i$  with  $x_{\text{train}}$  determines  $y_{\text{train}}$  and  $x_{\text{test}}$  together with  $\phi^i$  determine  $y_{\text{test}}$ , which is observed during meta-training and not during meta-test. The meta-training phase can involve training with multiple collections of datasets, like  $D_{\text{meta-train}}$ . The meta-test phase comes into play once meta-training is complete and one wants to adapt the model to a training set of a new task [61] [63].

## 2.6 Meta-Learning Approaches

There are three approaches to Meta-learning: First, Hyperparameter optimization. Second, Multitask-Learning and Transfer learning. Third, Ensemble techniques include bagging, stacking, voting, boosting [16] [61] [63]. All of the above approaches to Meta-learning are aimed at ensuring that the model performs at optimal levels in all aspects.

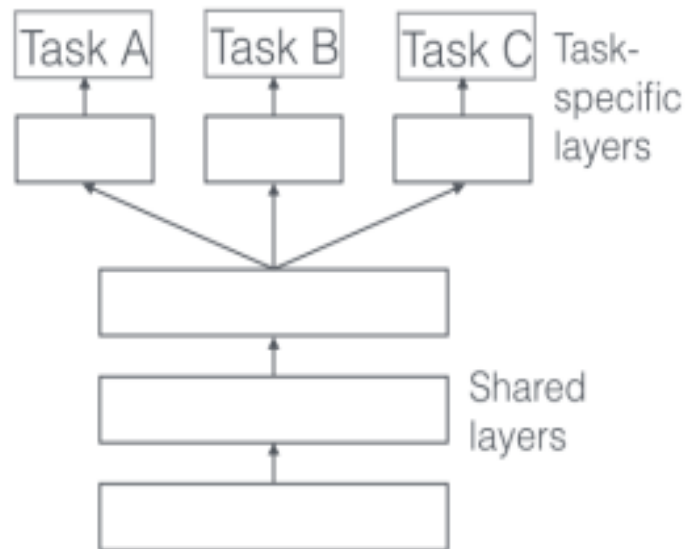
### 2.6.1 Multi-Task Learning

Multi-task learning is a learning approach that aims at jointly learning multiple tasks using shared knowledge among the tasks and then leveraging useful knowledge acquired from these several tasks to help improve the performance across all these related tasks through transfer learning.[64][65]. Zang and Yang [64] record that multi-task learning helps in eliminating overfitting by enabling a model to be more robust and learn a universal representation of multiple tasks. This also improves the model in the performance of each task. According to [66] multi-task learning is neural network that tries to simulate how human beings are able to apply previous knowledge in learning new things.

Multi-task learning can be implemented using two main approaches which are soft and hard parameter sharing of hidden layers[66]. Hard parameter sharing involves sharing the hidden layers across all the tasks while task specific output layers are sustained. It is the most commonly used method of multi-task learning since it reduces the possibility of model overfitting. This is so because the more the tasks the model learns the more the model tries to find a general representation of the tasks rather than a task specific representation. Zahng et al [67] argues that hard parameter sharing reduces storage cost and improves model's performance.

In hard parameter sharing, the bottom / shared layers act as feature extractors and thus they hold weights that can be transferred to top-layers/ task specific layers of individual tasks. This approach is known as top-specific parameter sharing. Hard parameter sharing and transfer learning go hand in hand since there is transfer of weights across tasks. Figure 2.5 shows hard multi-task learning [66]. The shared layer contains the general knowledge that is applicable to the three tasks A, B, and C. The task specific layers filter from the general knowledge specific knowledge that applies

to the specific task. The number of tasks is not limited to three, there can be as many tasks so long as the tasks are in line with the knowledge contained in the shared layers.

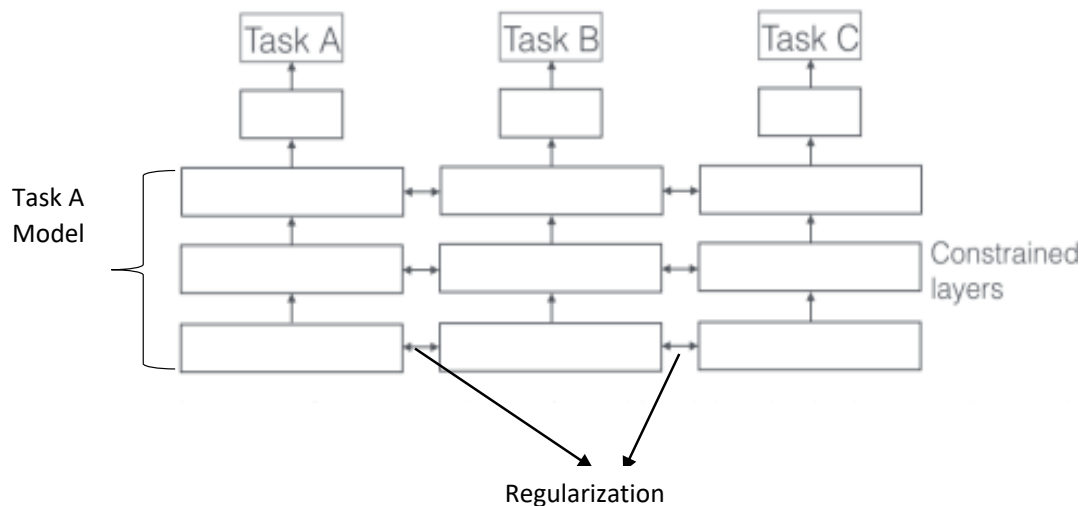


**Figure 2.5: Hard parameter Sharing in Multi-task Learning (Source; [66])**

Zahng et al [67] proposed an improvement to conventional hard parameter sharing by altering the shared layers. In the new model Zahng et al [67] used a bottom-specific approach in which, each task has its own task-specific bottom layers, then the top-layers are shared across the tasks. Experiments to validate this were conducted using MobileNetV2, ResNet50 and MNasNet neural network architectures. The neural network architectures were initially pre-trained using the ImageNet dataset then their multi-task versions are trained jointly through transfer learning to fit the task specific datasets. The task specific datasets used included: FGVC Aircraft, CUB200-2011, Stanford cars, Stanford Dogs and MIT indoor Scenes.

Soft parameter sharing involves each task having its own model with its specific parameters. The parameters are then encouraged to harmonize by regularizing the distance between them. There exist various regularization techniques that can be used in soft parameter sharing. They include; L1 regularization which estimates the

derivative of the loss function around the median of the data, L2 regularization in which the loss function tries to minimize the loss by subtracting it from the average of the data distribution, use of an autoencoder scheme which is an enhanced neural network with the same number of neurons in the input and output layers[68]. Figure 2.6 shows Soft parameter sharing [66]. Tasks A, B, and C represent the specific tasks while the constrained layers show the specific model for each task. The bidirectional horizontal lines interconnecting the constrained layers of each task represents the regularization that is used to harmonize parameters.



**Figure 2.6: Soft Parameter Sharing (Source; [66])**

### 2.6.2 Transfer Learning

Transfer learning is part of meta-learning that focuses on transferring knowledge that is learned from a task A to a new related task B [19]. This therefore, means that task B does not have to learn the features of the task from scratch since it leverages on what is already known from task A. Transfer learning is mainly used together with hard parameter sharing. Through transfer learning a model is able to easily achieve optimal performance without demanding very high training overheads [66].



Transfer learning mainly involves a series of steps which include the following: First, pre-training the model using a more general task. In image classification, researchers such as [17][18] have used the ImageNet dataset to pre-train the model. Second, the weights of the pre-trained model are transferred to the new but related task[69]. Therefore, a model that has been pre-trained for image classification, its weights can only be used for image classification task. In doing so the model is first frozen so that important features are not lost.

Third, task-specific trainable layers are added on top of the pre-trained model. These layers may vary depending on whether the researcher wants to modify the architecture or not. However, in most cases the top layer is replaced with a task-specific layer in terms of the number of neurons [70][71][60]. Also, the appropriate activation function for the output layer and the number of classes for the multi-class classification task are set. Fourth, training the model to extract features from the dataset of the new task B. This enables the model to use the knowledge gained from task A to extract features in a new task B as well as training the added layers[72].

Fifth fine-tuning the model to achieve optimal performance on the target dataset. This stage involves unfreezing some or part of the entire base model with the aim of now training it to fit the new task in the most optimal way[73]. Unfreezing the model increase the number of trainable weights thus making it possible for the model to fit the new dataset. This stage is very critical and involves a lot of hyperparameter tuning. The task is more challenging especially in a domain shift scenario whereby a model is pre-trained using a dataset in a certain domain then the weights are transferred to fit a task a new domain. For instance, a model pre-trained using ImageNet dataset and then the weights are transferred to medical imaging task will have a domain shift challenge.

Thus, much of the research in transfer learning focuses mainly on fine-tuning the model to fit the task at hand [73] [74]. The ability of a transfer learning to achieve optimal performance in task in different domain is known as domain shift generalizability. It differs from ordinary generalizability in the sense that domain shift generalizability focuses mainly on cross-domain transfer of knowledge while ordinary generalizability focuses on generalizing across related task. Domain shift problem is especially critical in scenarios where the model accuracy is not the only effective measure of performance success such as in medical imaging task.

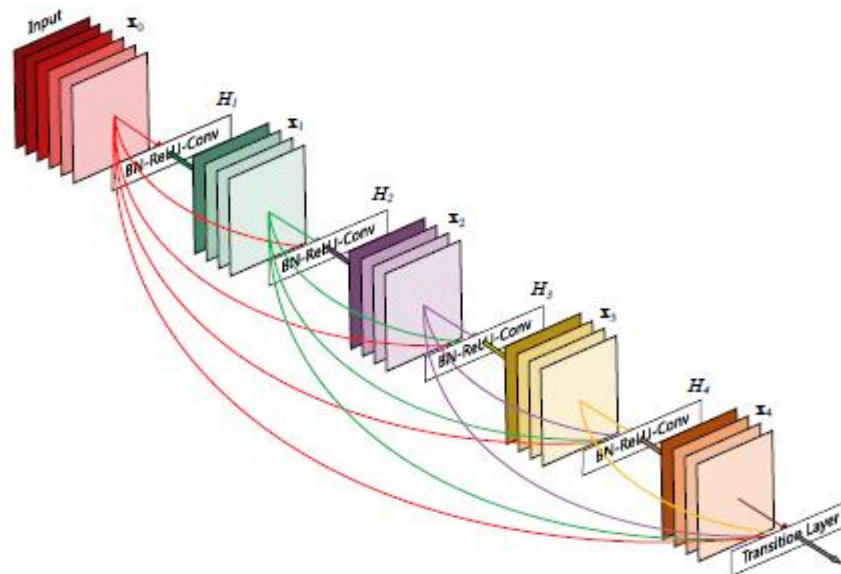
#### **2.6.2.1 Multi-task and Transfer Learning Architectures**

There exist several architectures that can be used for transfer learning and multi-task learning. The most popular open source architectures include the following: DenseNet, ResNet, EfficientNet, Inception, VGG, MobileNet, and NasNet.

##### **Densely Connected Convolutional Neural Networks**

DenseNet refers to a densely connected convolutional neural networks which connects each layer to every other layer in a feed forward format. Therefore for each layer the feature maps of all presiding layers are used as inputs and then its own feature maps are used as inputs in the subsequent layers [75]. Densely connected convolutional neural networks have several architectures which include: DenseNet169, DenseNet201, DenseNet121, DenseNet264. The depth of the neural network architecture is determined by the number that appears next to the name. DenseNets are advantageous in that apart from strengthening feature propagation, they also reduce vanishing gradient problem, reduces number of parameters and promotes feature reuse [75] [74]. Researchers such as Hassan et al. [74] used DenseNet in predicting covid-19 from CT-Scan images.

Figure 2.7 shows a 5-layers dense block in which each layer is connected to the other presiding layers [75].  $X_0, X_1, X_2, X_3, X_4$  represents the first, second, third, fourth, and fifth dense blocks respectively.  $H_1$  represents the first dense connection in the network which is between  $X_1$ , and  $X_2$ .  $H_2$  represents the dense connection between  $X_1$ , and  $X_2$ ,  $H_3$  represents the dense connection between  $X_2$ , and  $X_3$ ,  $H_4$  on the other hand shows the dense connection between  $X_3$ , and  $X_4$ .



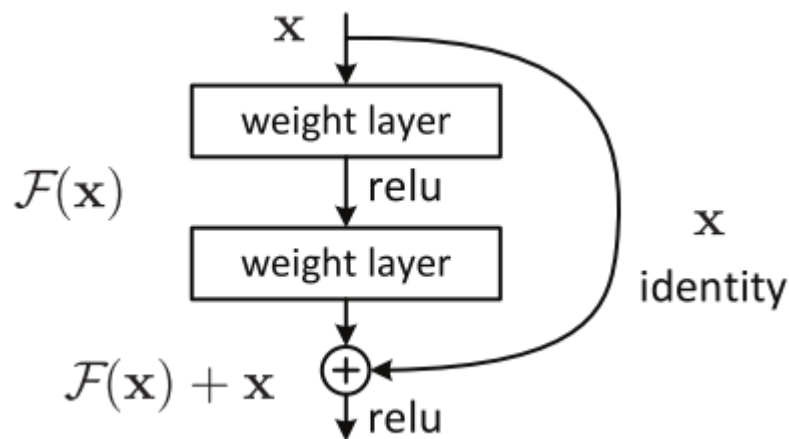
**Figure 2.7: DenseNet Architecture (Source; [75])**

### Residual Networks

ResNet refers to Residual Networks, it is an improvement of deep convolutional network that adds an identity short cut connection that skips some layers. The shortcut turns the network into its counterparts' residual versions[76]. This makes it possible for layers to be reformulated as learning residual functions with reference to the layers input. The main aim of residual networks is to make deep networks to be trained with less complexity while still achieving high performance. Residual networks have

several variants which are: ResNet18, ResNet50, ResNet101, ResNet152. The number after ResNet represents the number of layers in the residual network[76]. Mkuti and Biswas [60] used ResNet50 for plant disease detection. Algan et al.,[28] used ResNet50 and image dataset with noisy labels to classify diabetic retinopathy.

Figure 2.8 shows residual learning block with the identity shortcut [76].  $x$  represents the identity,  $f(x)$  represents the function of the model.  $f(x) + x$  show the output of combining the model's function with identify shortcut.



**Figure 2.8: Residual Learning Block( Source; [76])**

### Efficient Network

EfficientNet is network architecture that focuses on model scaling by uniformly scaling all the dimensions of height weight and resolution using compound coefficient[77]. The compound coefficient ensures that there is balancing in the scaling of all dimensions of a network, this balance is attained by using a constant scaling ratio. The scaling coefficients are determined by a simple grind search in the model. The EfficientNet family has seven architectures namely: EfficeintNetB0, EfficientNetB1, EfficeintNetB2, EfficientNetB3, EfficientNetB4, EfficientNetB5, EfficientB6, and EfficientB7[77]. The baseline model in the EfficientNet family is

the EfficientNetB0. All the other variants are obtained by scaling the EfficientNetB0 using the compound scaling method[77].

EfficientNetB7 demonstrated superior performance in the ImageNet dataset during experimental trials. Other researchers such as Tymchenko et al.,[30] used an ensemble of three convolutional neural network architectures namely (EfficientNet-B4, EfficientNet-B5, and SE-ResNext50) in diabetic retinopathy classification and achieved a Quadratic Weighted Kappa score 92.5%, [78] used EfficientNet for brain tumor classification, [42] on the other hand used EfficientNetB3 with an attention mechanism to classify remote sensing images. These studies have demonstrated that Efficient Nets are among some of the most used network architectures

### **Inception Network**

Inception Network was a game changer in convolutional neural networks. Prior to its introduction CNN models used to stack layers deeper and deeper hoping to achieve a high performance. The Inception network has evolved over time and today there exists the following versions of Inception network: InceptionV1 which has filters of multiple sizes operating at the same level thus making the network to be a little bit wider than deeper[79]. The InceptionV1 was the backbone of the GoogleNet architecture in 2014. InceptionV2 and Inception V3 were introduced together by [80].

Inception V2 made the following improvements to Inception V1: first it reduced representational bottlenecks by expanding the filter banks of the model thus making the model wider than deeper. Second, factorized a 5\*5 convolution to two 3\*3 convolutions to reduce computation cost[80]. The Inception V3 extended the Inception V2 by adding the following features to the architecture: Batch normalization in the auxiliary classifier, factorized 7\*7 convolutions, Batch normalization in the

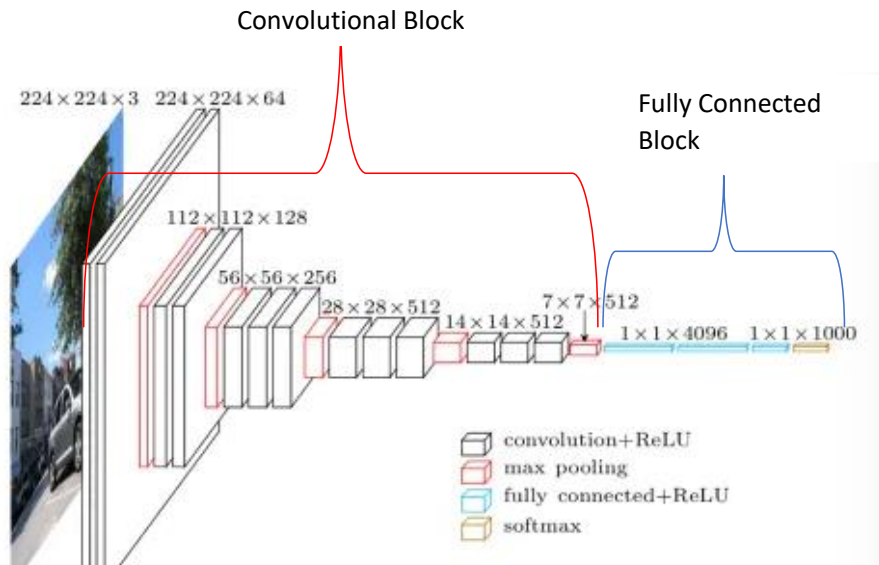
auxiliary classifier, RMSProp optimizer, and label smoothing. Label smoothing was a regularizing component that prevented overfitting[80].

Inception V4 modified the stem which consists of the initial set of operations before introducing the inception block[81]. This was done by adding specialized reduction blocks which are used to change the height and the width of the grid. The aims were to remove unnecessary computational baggage and to make the modules more uniform in order to improve on the performance of the models[81].

### **Visual Geometry Group Networks (VGGNets)**

Visual Geometry Group Networks (VGGNets) extended Alex Net. It did so by increasing the depth of the network using a small 3\*3 convolution filters[57]. Figure 2.9 shows the VGG architecture as proposed by [82] [57]. From the figure it is evident that the architecture of the VGG network consists of the following: First, an input layer which accepts colored images as inputs in the shape 224\*224\*3. This is followed by a convolution block which consists of the convolutional layers and 1\*1 convolution filter and Relu. The fully connected block follows thereafter and it has three fully connected layers with 4096, 4096, and 1000 neurons respectively. The fully connected layers are accompanied by Relu activation function [57].

Simonyan and Zisserman [57] record that there are four variants of VGG Networks which are: “First, VGG11 which supports 11 weight layers in the model’s (convolution layers). Second VGG13 which supports 13 weight layers. Third, VGG16 which supports 16 weight layers. Fourth VGG19 which supports 19 weight layers. VGG19 and VGG16 are the most commonly used architectures of the VGG model”. Researchers such as Khan et al. [23] used VGG16 to classify diabetic retinopathy and achieved a micro accuracy of 95% and an average accuracy of 83.8%.



**Figure 2.9: VGG Architecture (Source; [57] [82])**

Liu et al. [65] conducted a study that introduced the concept of hierarchical multi-task learning which can aid in discovering shared actions relatedness and action-specific feature subspace. The technique adopts an objective function regularized by the trace norm and group sparsity terms for joint multiple learning actions. Li et al [83] used multi-task self-supervised learning approach to improve the accuracy of deep learning models in small scopy datasets. The model is pretrained using a related task then the knowledge is transferred to mining useful information from limited training data, thus eliminating the need for huge training data sets which is the key goal of meta-learning. Jia et al. [84] used multi-agent and meta-learning approaches to develop a deep reinforcement learning model that was more scalable and flexible by building on previous knowledge.

### 2.6.3 Hyperparameter optimization

Hyperparameter optimization refers to fine-tuning the hyperparameters to achieve better optimization. Hyperparameter optimization goes hand-in-hand with the choice of algorithm. The key aim of hyperparameter optimization according to [85] is to reduce the loss value in the model. The most commonly used technique in hyperparameter optimization are Sequential model-based optimization (SMBO), Bayesian optimization, random search, and grid search [19] [86]. Keras tuner is also a formidable method for hyperparameter optimization especially where one is using the TensorFlow framework[87]. Although some manual tuning of parameters is still used, the above-mentioned techniques have proved to be more efficient than manual tuning [20]. Random and grid searches are considered to be uninformed of the previous evaluations since they don't pay attention to past results at all. Therefore, they may spend a lot of time evaluating bad hyperparameters.

The Bayesian optimization can tolerate stochastic noise in function evaluations. It uses a Gaussian process to quantify the uncertainty of surrogates it builds for the objective [20]. Unlike random and grid search, the Bayesian optimizer is informed of past evaluation results and they use them to form a probabilistic model mapping hyperparameters to a probability of score on the objective function [19].

Sequential model for hyperparameter optimization was originally proposed for black-box optimization and it is a formalization of the Bayesian optimizer. It tries to find a function  $f$  to represent the optimum function [18]. Finding the function  $f$  is an expensive exercise and thus this approach adopts a surrogate model  $\Psi$  and uses it to try and approximate  $f$ . The surrogate model consumes less time and is less expensive. The surrogate model is then combined with an acquisition function to tackle the exploitation-exploration dilemma [19].



Keras tuner is a general purpose hyperparameter tuning library that is maintained by Keras organization and works on top of TensorFlow. The library has very strong integration with Keras workflow however it can still be used in SCKIT learn. Keras tuner was developed with the key aim of allowing fast experimentation. Keras tuner helps in tuning hyperparameters such as number of layers, number of neurons per layer, learning rate, number of epochs, batch sizes, among others[88].

There are three main optimization methods in Keras tuner package namely Bayesian optimization, hyperband optimization, and random search. Keras tuner has been used in image classification by [88], it has also been used by [89] in Natural Language Processing. It has also been used in earthquake prediction in long-short term memory where the tuner enabled the researchers to get the best model among the candidate models [90].

#### **2.6.4 Ensemble Learning**

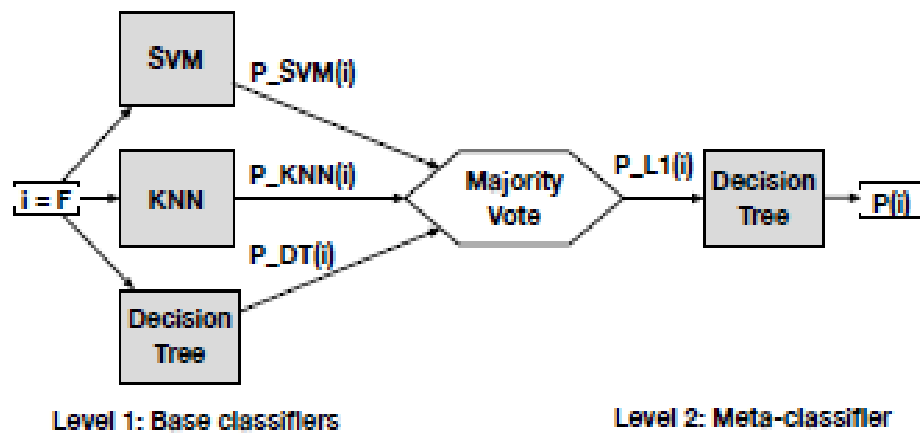
Ensemble learning is an umbrella term for methods that combine multiple inducers or base learners to make a decision[91]. Ensemble learning is motivated by the desire to combine the capabilities of different machine learning algorithms. The premise is that a combination of multiple inducers reduces the error rate of single inducer since it is likely to be compensated by the other inducers. Inducers can be of any machine learning algorithm. Ensemble learning borrows from the human nature of gathering different opinions, weighing them and using them to make decisions [91].

Ensemble methods perform better because of several reasons which include: first overfitting avoidance especially when one has small amounts of training data. This is so because ensemble learning averages the training hypothesis of all the models thus reducing the risk of choosing the wrong hypothesis that fits the training set and fails

to fit the test set. Second, computational advantage as result of minimizing the risk of being stuck in a local minimum. Third, better representation since by combining multiple models the search space is extended thus resulting to a better fit to the data space[91]. Ensemble approaches include voting, stacking, boosting, and bagging.

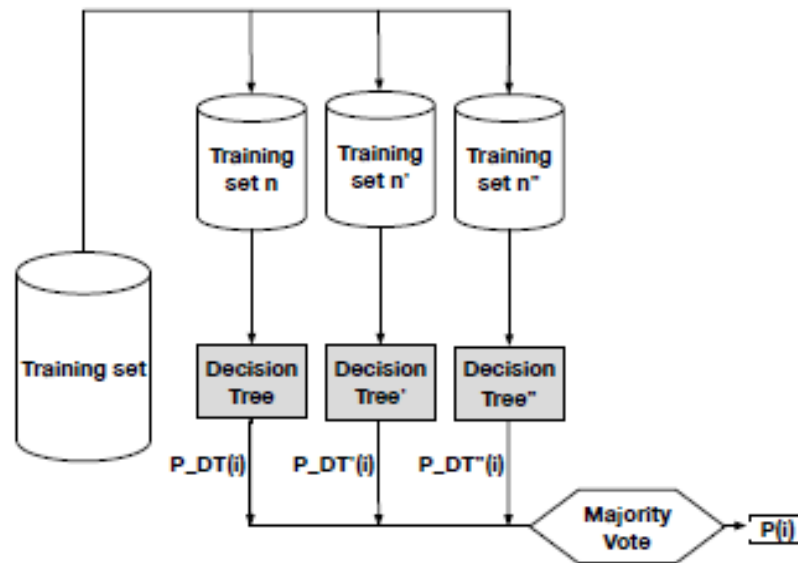
**Voting:** this approach involves using more than one machine learning algorithms to perform a task such as classification or regression. The base models then engage in a voting exercise. Each of the base models is entitled to only one vote. The model either classification or prediction that receives the highest votes, its value is considered to be the final value [16]. This approach provides room for the usage of an arbitrary number of classifiers.

**Stacking:** this involves a layered architecture in which each layer has one or more classifiers. “The prediction of a layer is used to extend the original feature vector of the corresponding instance. Each of these classifiers is trained with a subset of instances. Given a new instance to be classified, each classifier will produce a prediction. The predictions of these classifiers are then combined by majority voting” [16]. The resulting prediction is used to extend the feature vector of this instance, and this vector is the input for the next layer. Figure 2.10 below demonstrates stacking using SVM, KNN, and Decision tree. Where,  $i$  represents the input,  $P_{SVM}(i)$ ,  $P_{KNN}(i)$ , and  $P_{DT}(i)$  represents the predictions of the SVM, KNN, and Decision Tree respectively.  $P_L(i)$  represents the output of the first level after majority vote. This output is then passed through a decision tree to obtain the final prediction  $P(i)$ .



**Figure 2.10: Stacking (Source; [16])**

**Bagging:** unlike voting, bagging generates a combination of classifiers from the base classifier by manipulating the training dataset. Therefore, this approach involves selecting a single base classifier and then calling it multiple times using a subset of the training data set [16]. Each of these subsets of the training dataset are used to train the base classifier and this is considered to be a single model. The results of each of these models are then combined through a combination rule to get the final prediction [16]. It can be viewed more like divide and conquer approach. Figure 2.11 illustrates bagging approach in which the meta-classifier is using voting combination rule. In this illustration a decision tree has been used three times for bagging and then subjected to majority voting to get the final prediction.



**Figure 2.11: Bagging(Source; [16])**

**Boosting:** involves creating multiple base classifiers by sequentially assigning new weights to the instances of the dataset. In the first phase, all instances have the same weights. After the first phase, each iteration adopts a base classifier to the training instances with their respective weights [16]. Then “the error is computed and the weight of the correctly sorted instances is reduced while the weight of the wrongly sorted instances is increased. The final model obtained by the boosting technique is a linear combination of several base classifiers, weighted for the best performance”. This process can be looped until the desired performance is achieved or until when no further improvement can be done [16].

## 2.7 Model Evaluation

Model evaluation refers to assessing the performance of a model using various metrics. Performance evaluation metrics makes it possible for researchers to determine how the model has performed in with the training set, validation set, and testing set.

Model evaluation metrics vary in terms of what they assess in a classification model. The choice of the metric to use is highly influenced by the nature of task, the dataset used, the goal of the model, and also the environment setup.

### 2.7.1 Confusion Matrix

A confusion matrix is a performance measurement of machine learning models that is used in classification, regression, and reinforcement learning. It consists of table with four different values of predicted and actual values. A confusion matrix has two dimensions as shown in figure 2.12, the first dimension is the actual class of the object while the second dimension is the class that the classifier predicts[92]. There are several metrics that can be computed from the confusion matrix, they include: Accuracy, Recall, Precision, F1\_score, Area under the Curve (AUC), Receiver Operating Curve (ROC).

		Actual Values	
		Positive (1)	Negative (0)
Predicted Values	Positive (1)	TP	FP
	Negative (0)	FN	TN

**Figure 2.12: Confusion Matrix (Source; [92])**

**True Positive (TP):** refers to the cases in which the model predicted a particular class and in real sense the object belongs to that class. In the case of multi-class classification, the object is classified in the right class.

**True Negative (TN):** this is where the model predicts the class as negative and it is actually negative. In the case of multi-class classification, the object is classified in the right class.

**False Positive (FP):** this is where the model predicts an object as positive but in real sense it is negative. Or in case of multi-class classification, the object is classified in the wrong class.

**False Negative (FN):** The model predicts an object as negative and in real sense it is positive. Or in case of multi-class classification, the object is classified in the wrong class.

### **Accuracy**

Accuracy refers to the percentage of total accurate predictions which is based on the positive and negative classes classification. Accuracy is only reliable when you have a balanced dataset which is dataset with equal number of samples in each class[36]. This concept of unreliability of accuracy with unbalanced data is especially essential in medical imaging task where it is costlier to fail to detect a disease than it is to take a healthy person through a series of tests. Accuracy can be computed using Equation 2.4.

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

Equation 2.4

**Recall/ Sensitivity:** This is also known as the True Positive rate, it is the number of correct positive results divided by the number of *all* relevant samples. Recall measures the model's ability to detect positive samples. It is computed using Equation 2.5 [93]:

$$\text{Sensitivity} = \frac{TP}{TP+FN}$$

Equation 2.5

**Specificity:** It is also known as the True Negative Rate (TNR). It is computed using Equation 2.6 [93]

$$\text{Specificity} = \frac{TN}{TN+FP}$$

Equation 2.6

**Precision:** It is the ratio of truly classified number of samples and the given sum of True Positive and False Positive. Equation 2.7 shows how precision is computed [93]

$$\text{Precision} = \frac{TP}{TP+FP}$$

Equation 2.7

**F1-Score:** it is the harmonic mean of recall and precision, it has values between 0 and 1 where 0 is the lowest and 1 is the best score. High precision but lower recall, gives you an extremely accurate, but it then misses a large number of instances that are difficult to classify. The greater the F1 Score, the better is the performance of the model. Equation 2.8 shows how F1-Score is computed.

$$F1\_score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

Equation 2.8 (Source; [94])

**Area under the Curve (AUC):** refers to the degree of separability between classes. The higher the AUC score the better the model has learned. The AUC values also range from 0 to 1 where 0 is the lowest and 1 is the highest[95]. When the AUC is 1 it means that the model is able to perfectly distinguish between all classes. When the AUC is between 0.5 and 1 it means that there are higher chances that the model is able

to distinguish between classes. This is so because the model is able to detect more number of True positives and True Negatives. Where the AUC is 0.5 it means that the classifier is not able to distinguish between positive and negative class points. This shows that the classifier is either predicting a random class or a constant class for all data points[95].

**Receiver Operating Curve (ROC):** Refers to when the True positive rate (TPR) is plotted against the False Positive Rate (FPR). The ROC curve has two very critical measures of performance which are the Mirco-average and macro-average. Macro averaging reduces multiclass predictions to multiple sets of binary classifications[96]. It calculates the corresponding metric for each binary case and then averages the results together. Take an instance of multiclass classification with three classes A, B, and C.

“Macro-average reduces the problem to numerous all versus one comparison. The truth and estimate columns are recoded such that the only two levels are A and other. The recorded columns are then used to calculate precision, with A being the “relevant” column. This process is repeated for the other 3 levels to get a total of 4 precision values. The results are then averaged together”[97]. Equation 2.9 shows the formula representation of K classes in Macro-Averaging.

$$Pr_{macro} = \frac{Pr_1 + Pr_2 + \dots + Pr_k}{k} = Pr_1 \frac{1}{k} + Pr_2 \frac{1}{k} + \dots + Pr_k \frac{1}{k}$$

Equation 2.9 (Source; [97])

where PR1 is the precision calculated from recoding the multiclass predictions down to just class 1 and other. Macro averaging is suitable for a balanced dataset since all classes get equal weight when contributing their portion of the precision value to the



total (here 1/4). Thus huge amounts of class imbalance can highly affect the reliability of this metric[97]. To address this a weighted macro average can be used since it calculates weights from the frequency of the class in the truth column. Equation 2.10 shows the weighted Macro-Averaging.

$$Pr_{weighted-macro} = Pr_1 \frac{\#Obs_1}{N} + Pr_2 \frac{\#Obs_2}{N} + \dots + Pr_k \frac{\#Obs_k}{N} \quad [97]$$

Equation 2.10 (Source; [97])

Micro-averaging on the other hand “treats the entire set of data as an aggregate result, and calculates 1 metric rather than k metrics that get averaged together. For precision, this works by calculating all of the true positive results for each class and using that as the numerator, and then calculating all of the true positive and false positive results for each class, and using that as the denominator” as shown in Equation 2.11 [97].

$$Pr_{micro} = \frac{TP_1 + TP_2 + \dots + TP_k}{(TP_1 + TP_2 + \dots + TP_k) + (FP_1 + FP_2 + \dots + FP_k)}$$

Equation 2.11

In this case, rather than each *class* having equal weight, each *observation* gets equal weight. This gives the classes with the most observations more power.

### 2.7.2 Quadratic Weighted Kappa Metric (QK)

It is a metric that measures the level of disagreement between actual and predicted labels. “The Quadratic kappa metric can be computed using the three matrices namely Expected Matrix (E), Output Matrix (O), and Quadratic Weighted Metric (W)” [18].

The Quadratic Weighted Kappa Metric can be computed using the following process: The First step involves “calculating the Expected Matrix (E), by taking the outer

products between the actual labels vector and target labels vector. Second, the output matrix is constructed by building a confusion matrix of actual labels and predicted labels”. Third the weighted matrix is computed using the formula in the Equation 2.12 [18].

$$W_{ij} = \frac{(i - j)^2}{(k - 1)^2}$$

Equation 2.12

Where  $i$  is the actual label,  $j$  is the predicted label, and  $k$  is the number of classes

Fourth, normalize the expectation matrix (E) and output matrix (O) by dividing them by their sum. Fifth, calculate the weighted kappa using Equation 2.13 [18].

$$\text{Quadratic Weighted Kappa} = 1 - \frac{\text{num}}{\text{den}}$$

Equation 2.13

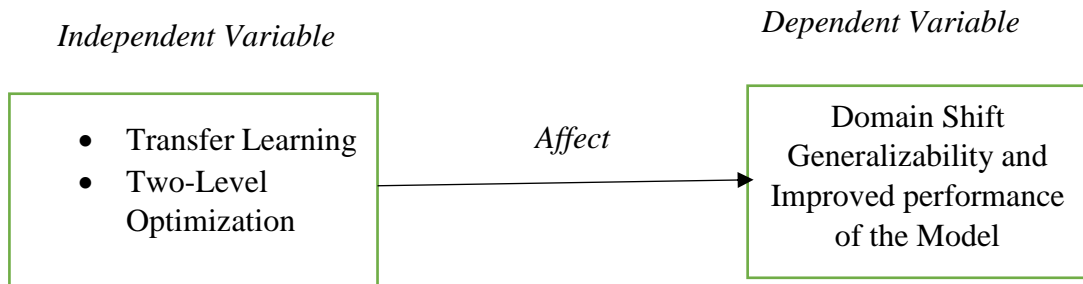
Where “ $num$  is the sum of elements obtained using element-wise multiplication between weight matrix (W) and output matrix (O), and  $den$  is the sum of elements obtained using element-wise multiplication between weight matrix (W) and expectation matrix (E)” [18]. QWK can range from -1 to 1 where 1 is perfect agreement while -1 is total disagreement between actual labels and predicted labels[18].

## 2.8 Conceptual Framework

A conceptual framework shows the relationship between various concepts in a research. This relationship is depicted using the independent and dependent variables. In this study the researcher experimented on how transfer learning and hyperparameter

optimization affects domain-shift generalizability and optimal performance of models.

Figure 2.13 shows the conceptual framework.



**Figure 2.13: Conceptual Framework**

## 2.9 Summary

The review of literature in this chapter aimed at first analyzing related works on how deep learning and meta-learning has been used in detecting and classifying diabetic retinopathy. Second, exploring meta-learning techniques used in classification tasks. Third, analyzing model testing and validation techniques that are currently being used in this field. The review of literature has revealed that indeed meta-learning techniques have been used by other researchers in detecting diabetic retinopathy. The most used meta-learning technique is transfer learning which involves leveraging on a previously acquired knowledge from pre-training models with the ImageNet dataset. There exist neural network architectures that can be used for transfer learning with most commonly used ones being; EfficientNets, DenseNets, ResNets, VGGNets, and Inception Nets. Transfer learning has proved to be more superior than standard deep learning. However, the existing models are still challenged in domain shift generalizability since they do not do proper model optimization to achieve high performance in classifying diabetic retinopathy to its five classes. The literature also revealed that apart from transfer learning other meta-learning techniques include multi-task learning, self-optimization, and ensemble learning. Further, the literature

revealed that there exists metrics such as Accuracy, Precision, Recall, F1-score, Area under the Curve, Quadratic Weighted Kappa Metric, and Receiver Operating Curve that can be used to evaluate performance of a model during validation and testing.

## **CHAPTER THREE**

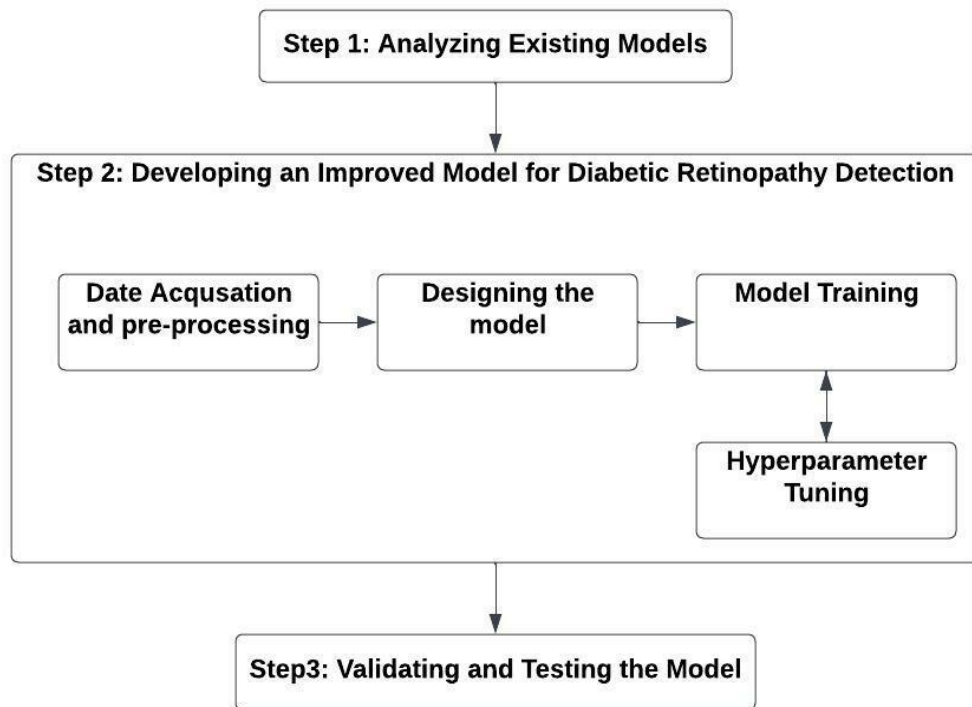
### **METHODOLOGY**

#### **3.1 Introduction**

This chapter discusses the methodology, the dataset, and the tools that the researcher used to achieve the research objectives. It explains the research process, research design, data acquisition, model development, model validation and model testing.

#### **3.2 Research Process**

Research process refers to the series of steps that a researcher undertakes in their research in order to achieve their objectives. In this research, the research process had three stages as per the research objectives. The first step involved analyzing the existing deep learning and transfer learning models for classification of diabetic retinopathy. The aim of this step was to first appreciate what has been done so far and then critique in terms of research gap these existing works. This also helped the researcher in choosing the neural network architecture to use for transfer learning. The second step which covered objective two involved redesigning the identified transfer learning model and tuning its training hyperparameters using two-level optimization. This second step has activities such as model development, model training, and hyperparameter tuning to achieve optimal performance. The final step involved validating and testing the developed model. Figure 3.1 shows the research process.



**Figure 3.1: Research Process**

### 3.3 Research Design

Research design is the overall strategy that the researcher uses to address the research problem. Akhtar [98] argues that research design is the glue that holds all the components of a research together. Research design not only envisages the choices of methods, approaches, and techniques but also the logical foundation for considering those methodologies. The choice of research design is influenced by the nature of evidence required to answer the research questions[98]. Therefore, a suitable research design is one which encapsulates all the processes to be involved in the research in order to answer the research questions. A suitable research design should also contain at least the following critical elements: the statement of the problem, procedures and techniques to be used for data collection, study population, data processing and analysis methods.

One of the most popular research designs in the field of computing and engineering is the experimental research design. [99] defines experimentation as “a recording of observations, quantitative or qualitative, made by defined and recorded operations and in defined conditions, followed by examination of the data, by appropriate statistical and mathematical rules, for the existence of significant relations”. Experimental methods build on technical insights and technologies and aims at bringing a contribution to the advancement of these technologies[99]. Experimental research design is further grouped into three categories namely Pre-experimental, true experimental and Quasi experimental research designs.

The researcher used true experimental research design in this research since the entire process of developing, testing and validating the model was an experiment. The research also involved the use of statistical analysis to determine cause and effect which is a major characteristic of true experimental design[100].The choice of this design was informed by the nature of the research and previous empirical studies that developed models for detection of diabetic retinopathy[18][17][23].

### **3.4 Identification of Best Architecture for Transfer learning Model**

The researcher used literature review to address objective one. Literature review involved identifying, selecting, and critically appraising research in order to answer a research question [101]. This review involved two approaches: the first one was a systematic literature review of how meta-learning has been used in classification tasks. The second approach was a literature review in which the researcher reviewed research articles on existing deep learning and meta-learning models with an aim of identifying the techniques used in these existing models. The articles reviewed were sourced from high-end journals and were not more than ten years old. The researcher also reviewed

related works that specifically apply the meta-learning or deep learning techniques in detecting diabetic retinopathy. This review is presented in chapter two.

There exist multiple architectures that can be used for transfer learning. The most commonly used architectures in literature are; InceptionV3, Residual Networks (ResNet), Densely Connected Networks (DenseNet), EfficientNet family, and Visual Geometry Group Networks (VGGNet). To determine the best architecture to use in this research the researcher conducted a comparative analysis of ResNet50, DenseNet169, EfficientNetB0, and VGG16 architectures. The aim was to determine which architecture is superior in classifying fundus image under the same environmental set up.

The comparative analysis involved setting up an experiment in which the architectures were trained using the Indian diabetic retinopathy and the Aptos 2015 datasets. The datasets were converted into 4-class classification problem using the Messidor dataset classification as the benchmark. The choice of the dataset in this section was informed by the fact that the aim of the comparative analysis was to determine which of the architectures is superior than the others. Other researchers such as [31] [27] have used the same dataset benchmarks. The overall dataset was small in size consisting of 4125 images, this provided an easier way to compare the architectures without too much training overhead.

The images were then pre-processed using TensorFlow library which involved resizing them to shape  $224*224*3$  to fit the default input shape of all architectures under consideration. Data augmentation that was done on this model involved: horizontal random flip, random rotation, and random translation. The aim of data



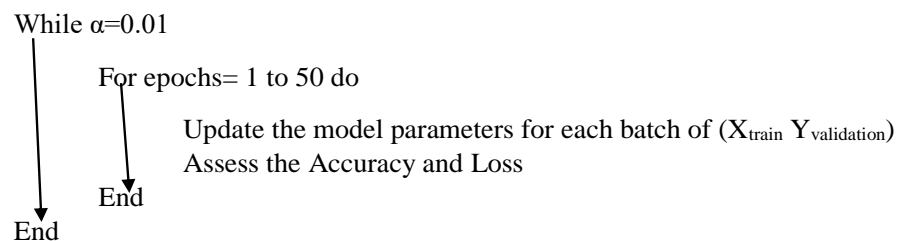
augmentation in these experiments was to several representations of the same image so that the model can be able to analyze an image from multiple viewpoints[102].

The architectures were then trained in Google Collaboratory pro with Nvidia GPU and TensorFlow library which has Keras in its back end. The training involved two approaches which were, first, deep learning approach, this consisted of training the architecture for 50 epochs without the transferred ImageNet weights. The second approach was transfer learning approach which involved training the architecture with transferred weights.

Input →Fundus Images belonging to 4 classes (DR0, DR1, DR3, DR4)

Output →A model that classifies fundus images into the four classes

- 1) Load the dataset
- 2) Data pre-processing
  - Split 80% training and 20% Validation ( $X_{\text{train}}$   $Y_{\text{validation}}$ )
  - Resizing images
  - Data Augmentation
    - ❖ Horizontal random flip
    - ❖ Random rotation
    - ❖ Random Contrast
    - ❖ Random translation (h-factor=0.1, W\_factor=0.1)
- 3) Import the architectures without the weights= (EfficientNet B0, DenseNet169, ResNet50, VGG 16)
- 4) Set the model parameters (Optimizer, Loss function, Metrics)



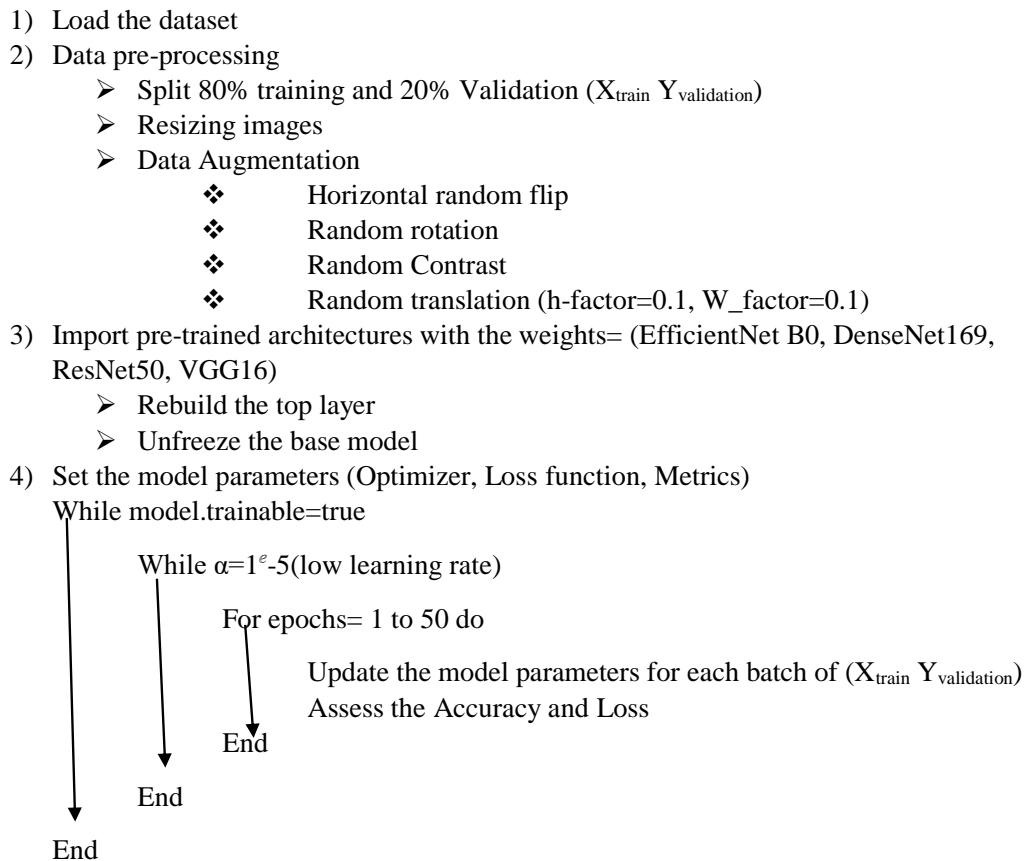
### *Algorithm 1: Deep Learning Approach*

Algorithm 1 shows the deep learning algorithm that was used to train the architectures without the transferred weights. Therefore, the algorithm shows how the architectures were trained from scratch. The following model hyperparameters were set, Adam optimizer with the default learning rate of  $\alpha = 0.001$ , categorical cross-entropy loss

function, and Accuracy metrics. The models were trained for 50 epochs and their performance was recorded using model checkpoints.

Input → Fundus Images belonging to 4 classes (DR0, DR1, DR3, DR4)

Output → A model that classifies fundus images into the four classes



### *Algorithm 2: Transfer Learning Approach*

Algorithm 2 shows the transfer learning approach. The imported neural network architectures had been initially pre-trained using the ImageNet dataset. The architectures were then restructured to make them suitable to the task of detecting and classifying diabetic retinopathy. Restructuring of the architecture involved rebuilding the top layer as follows: A Global Average pooling 2D layer, a Batch Normalization layer, dropout layer with a dropout rate of 0.2, a dense layer as the output with four neurons and SoftMax activation function, were added. The base models were unfrozen

by setting the trainable function of the models as true. The following model parameters were used for all the architectures: Adam optimizer with a learning rate of ( $\alpha=1^e-5$ ), categorical cross-entropy loss function, and a batch size of 32. The architectures were trained for 50 epochs and the performance was recorded. The performance results of these architectures were then compared to determine the best architecture of the best architecture.

The detailed results of this comparative study have been discussed in chapter 4 section 4.2 and a paper containing the results of this comparative study has been published (see appendix 4).

### **3.5 Data Sourcing**

The researcher used a secondary datasets of eye fundus images which was obtained from Kaggle repository. The dataset is known as the EyePACs Diabetic retinopathy dataset which has 35126 colored fundus images. Eyepacs is an organization that specializes in screening of diabetic retinopathy, selling screening cameras, and providing programs that enable patients to manage diabetic retinopathy. The organization also conducts research on diabetic retinopathy management in partnership with University of California Berkeley[103]. Therefore, the dataset is labeled by experts in the area of diabetic retinopathy. Thus it has been used as benchmark by multiple studies such as [18][36][23]

The dataset contains fundus images belonging to the five classes of diabetic retinopathy. The distribution of the dataset was as shown in Table 3.1.

**Table 3.1: Dataset Distribution**

<b>Level</b>	<b>Number of Images</b>	<b>Label used in the model</b>
No DR	25810	Level_0
Mild DR	2443	Level_1
Moderate DR	5292	Level_2
Severe DR	873	Level_3
Proliferative DR	708	Level_4

The choice of secondary dataset over primary data was informed by previous studies which also used secondary data [10] [11] [12] [13]. Also, primary data for this field is quite time intensive and expensive to acquire. After acquiring the dataset the researcher did data pre-processing.

Data pre-processing refers to the process of preparing raw data and making it suitable for training in machine learning. The motivation behind data pre-processing is the reality that much of the data collected from the real world is not always in a form ready to be used for training. Data pre-processing can help a machine learning model to optimize its performance[104]. This is so because pre-processing eliminates noise in the data and only provides the machine learning model with what is relevant for it to learn. The specific activities conducted under data pre-processing vary from one model to another and they are influenced by several factors such as: nature of the dataset, nature of the task, capability of the algorithm used, training pipeline, among others[104].

This research is an image processing task; therefore, data pre-processing activities followed the conventions of image processing. The first step involved resizing the images into 512\*512 pixels. Second, the dataset in this study was imbalanced in the sense that the No diabetic retinopathy class consisted of more than 50% of the entire dataset. This is a true reflection of real-world data where out of 100 random people tested there is likely to be more people without diabetic retinopathy than those with diabetic retinopathy.

To address this challenge selective data augmentation was done. This involved augmenting the minority classes only (Level 1 to Level 5) with the aim of increasing their instances. The specific augmentations done to these classes were: random flip, random rotation, random zoom (height\_factor = (-0.2, 0.3), width\_factor (-0.2, -0.3), interpolation= 'bilinear', random contrast, and random translation. Third, the dataset was split into training set (70%), validation set (10%), and testing set (20%). Fourth, one-hot encoding of the three sets of the data was done.

### 3.6 Model Development

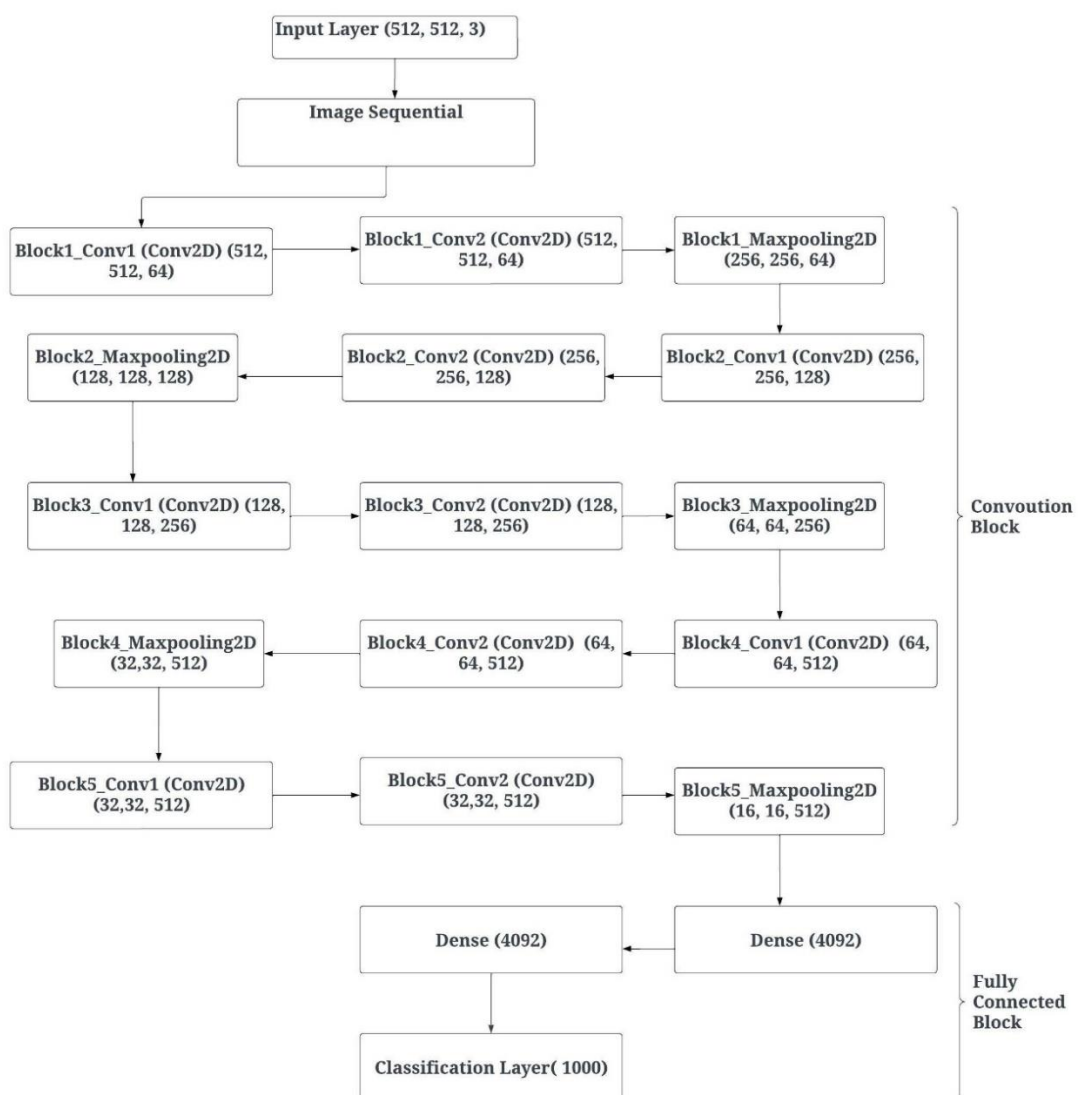
The researcher chose the VGG16 architecture based on the results obtained from the comparative study. Figure 3.2 shows a code snippet that the researcher used to import the VGG16 architecture. The code specifies that the weights to be used are the ImageNet weights.

```
from tensorflow.keras.applications import VGG16
img1_shape = (512,512,3)
model_224 = VGG16(include_top=False, weights='imagenet', input_shape=img1_shape)
print(model_224.summary())
```

**Figure 3.2: Code for importing VGG16 Architecture**

Figure 2.9 shows visual representation of the VGG16 architecture, while Figure 3.3 shows a detailed layer-by-layer VGG16 architecture. The architecture has two main

sections namely the convolution block and the fully connected layer block. The convolution block consists of the input layer, Convolutional layers, Rectified Linear Unit (ReLU) activation function, and max pooling layers. The fully connected layer section consists of dense layers, output layer, Rectified Linear Unit (ReLU) activation function, and Softmax activation function. Figure 3.3 shows a layer-by-layer architecture of the VGG16 network. This architecture is the default architecture before any modification has been done to it.



**Figure 3.3: Layer-by-layer VGG16 Architecture before layers' Modification**

This default architecture accepts input of size  $224*224*3$  although researchers can adjust the input shape. In this case the input shape was adjusted from  $224*224*3$  to  $512*512*3$ . The default architecture also classifies images into 1000 classes since it has been pretrained using the ImageNet dataset. Therefore, the default architecture was not appropriate for classifying diabetic retinopathy since DR classification is a 5-class classification problem. To address this, the researcher rebuilt the fully connected part of the network and also added an attention model to the network.

The network was rebuilt as follows: First an attention model was added as part of the Image sequential layer, the role of the attention model was to help the model in focusing on the critical features of the fundus images that differentiate the image of one class from another. The attention layer was customized to use self-attention where the query comes from the input. Self-attention was considered to be suitable since it will provide the model with capabilities to automatically define what in the input should be given attention [105]. This not only increases the classification capability of the model but also makes the model to be more robust. Thus, making it more useful in real world applications where input images can vary in size, contrast, shape, background noise etc.

An attention mechanism has been used by researchers such as [106] have used attention mechanism to increase accuracy in text classification, however, the researchers placed the attention mechanism in the classification part of the network, thus the attention mechanism did not help in features extraction. Others such as [107] used a dual attention mechanism involving channel attention and spatial attention to classify hyperspectral images. The model achieved a good performance. However, in both cases the attention mechanism used is not data-driven, thus there is need to use an attention mechanism that is data-driven. This forms the basis under which the

researcher in this study settled on using a self-attention mechanism and placed it as part of the image sequential model in the modified VGG16 architecture.

Second, a Global average pooling layer was added at the beginning of the fully connected block. The role of this global average pooling layer was to enforce correspondence between feature maps and class categories. The layer does this by generating one feature map for each corresponding category of the classification task in the last convolution layer[108]. The feature maps are generated by calculating the average output of each feature map in the previous layers. It prepares the model for classification and reduces the complexity of the model by reducing the number of trainable weights since it has no trainable parameters[109]. Also, The Global average pooling layer avoids overfitting since it does not have any parameters that need optimization, this makes it more suitable compared to fully connected layers[108].

Third, a Batch Normalization layer was added after the Global Average pooling layer. Batch Normalization enables faster and more stable training of the network. It achieves this by making the optimization landscape significantly smoother thus inducing more predictive and stable behavior of gradients[110]. This impact of batch normalization on the training process informed the researcher's decision to include it as part of the network.

Fourth, a dropout layer with a dropout rate of 0.2 was added below the batch normalization layer. The role of the dropout layer was to regulate the network training process. It does so by dropping some neurons in the layers during training. The number of neurons dropped are determined by the dropout rate which represents the percentage of neurons to be dropped. The dropout rate is a hyperparameter that is usually tuned by researchers to optimize the network. A dropout rate of 0.2 means 20% of neurons



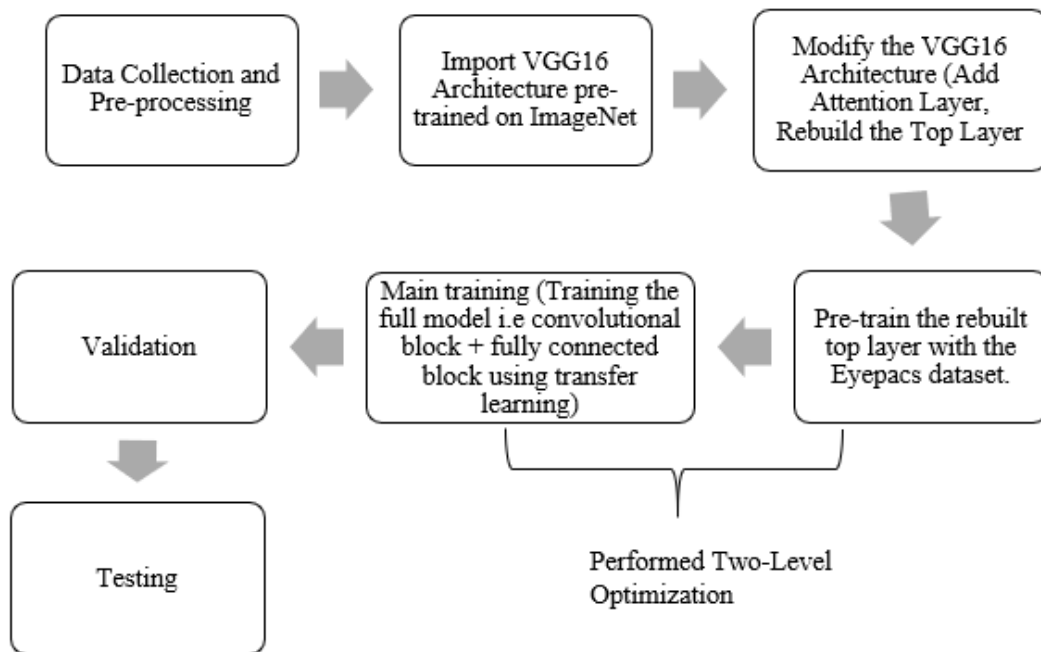
will be dropped. This plays a very critical role in preventing overfitting as a result of the network learning the noise in the images.

Fifth, the final classification layer with a SoftMax activation function was added. The classification layer was assigned 5 neurons in order to classify diabetic retinopathy into the five classes. A SoftMax activation function was used since it normalizes the output converting them into probabilities, therefore, each output of SoftMax is interpreted as the probability of membership for each class[111]. This, therefore makes SoftMax suitable for multi-class classification problem. Related works in literature have also used a SoftMax function [18], [23], [17]. The classification layer is the final output layer of the network and thus it forms the end of the network.

### **Model Training**

Model training involved several phases as depicted in figure 3.4.

Model training started with using a pretrained model for feature extraction. The pretrained VGG16 model was imported from Keras library. The model had been pretrained using the ImageNet dataset which is dataset consisting of over 1.2 million images of objects grouped into 1000 classes. The ImageNet dataset has been used previously to train other transfer learning architectures studied in section 4.4.3. This is due to the fact that the dataset is a general-purpose dataset whose features and weights can be used in transfer learning to train other domain specific models. The Eyepacs dataset was split into 70% training, 20% testing and 10% validation. This split was informed by previous researchers such as [17][18]who have used almost similar split.



**Figure 3.4: Model Development and Training**

### **Two-Level optimization**

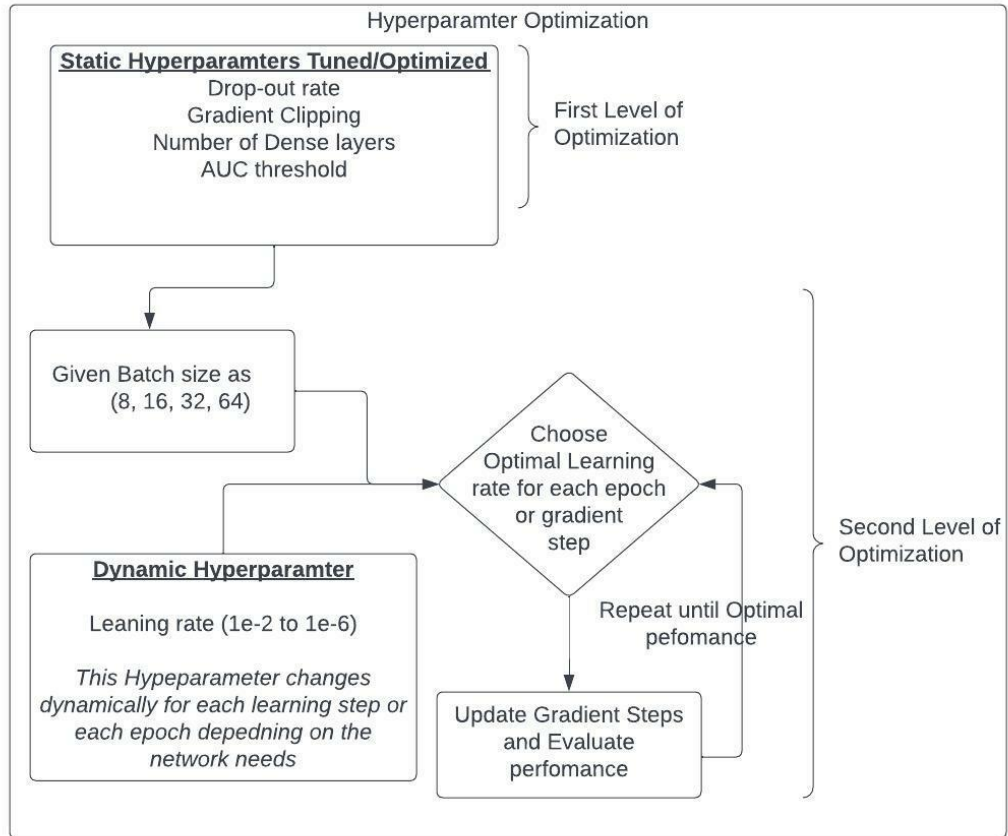
This involved optimizing the network in two key levels. The first level involved tuning some hyperparameters using Keras tuner and manual tuning. The hyperparameters that were tuned using this approach included: learning rate range, drop-out rate, gradient clipping, number of dense layers, number of neurons in the dense layers, AUC threshold, data augmentation height and width factors. The second level of optimization involved providing the model with a progressive batch size starting with small batch size of 8 in the top layer training, followed by batch size of 16, 32, and 64 in the full model training 1<sup>st</sup>, 2<sup>nd</sup>, and 3<sup>rd</sup> sessions respectively. Once provided with the batch size the model would automatically tune the learning rate by choosing the most optimal learning rate from the range 1e-2 to 1e-6 for a given batch size. Therefore, the model did self-evaluation through checkpoints during the gradient steps and adjusted

the learning rate to minimize loss and increase convergence which then results to high performance. This approach was informed by the fact that the two hyperparameters (learning rate and batch size) were discovered to be highly affecting the performance of the network.

Kandel and Castalli [112] studied the relationship between batch size and learning rate and they discovered that during the training process the network may need different batch sizes at different instances. [113][112] further discovered that the right batch size with the wrong learning rate will lower the performance of the model. However, the researchers did not hit a balance between the right batch size and the optimal learning rate. To address this unresolved challenge the researcher used dynamic learning rate and progressive batch size.

This involved starting with a small batch size of size 8 during feature extraction, then progressively increasing the batch size to 16, 32, and finally 64 in different training sessions. During these training sessions, the learning rate was set to range of  $10^{-2}$  -  $10^{-6}$ . Therefore, during training the model would automatically choose the most optimal rate for the specific batch size, as well as adjust the learning rate from one epoch to another through observing the improvement in the gradient steps towards the local minima while still considering the global convergence.

Figure 3.5 shows two-level optimization architecture.



**Figure 3.5: Two-Level Optimization**

The second phase involved extracting features from the EyePacs dataset. This was done by freezing the convolutional block of the VGG16 architecture and only training the rebuilt fully connected layer block with Eyepacs dataset. The model was trained for 40 epochs with a batch size of 8. This made it possible for the new model architecture to extract important features from the dataset. The extracted features were useful in the next phase of training. In doing so the model used Adam optimizer, categorical cross-entropy loss function, and gradient clipping. The model used two-level optimization as illustrated in Figure 3.5.

The aim of feature extraction was to get a learning function  $\theta^*$ , which contains all the information the model needs to know to solve the task. The algorithmic representation of this is as follows:

$$\theta^* = \arg \max_{\theta} \log p(\theta | D_{\text{meta-train}})$$

Equation 2.2 (Source; [4] [61])

The third phase of training involved fine-tuning the model by unfreezing the convolutional block. The weights obtained by the fully connected block from the feature extraction phase are used together with the ImageNet weights stored in the convolutional block. The model is trained for three phases of 40, 20, and 20 epochs respectively with two-level optimization. The progressive batch-size was set at sizes 16, 32, and 64 for the three phases. Fine-tuning the parameters on the training task of each meta-learning helped to achieve meta-optimization and adaptation as per the following equation 3;

$$\varphi^* = \arg \max_{\varphi} \log p(\varphi | D_{\text{tr}}, \theta)$$

Equation 2.3 (Source; [13])

Training using cycles was essential because according to Kandel and Castalli [112] a machine learning model should start training with a small batch size so that the model can extract relevant features from the data in the first phases of training. However, using a small batch size all through the model training will increase training time and reduce convergence. Therefore, [112] recommends that the batch size be increased as the model approaches convergence so that it can converge.

The cycles were settled at based on two main reasons: first the training of top-layer only is standard practice in transfer learning to help extract features from the dataset

it has been previously used by researchers such as [19][18]. Second, the performance of training and validation curves informed whether a new cycle is needed or not. Where both the training and validation curves achieve almost a flat curve as is the case in figures 4.6 and 4.8, then a new training cycle is necessary to overcome the curve. Also, where both training and validation curves demonstrate potential progress by not converging as is the case in figure 4.7, then a new cycle is needed to achieve optimal performance. Else if only the validation curve achieves a flat curve as is the case in figure 4.9 then this means that the model has achieved optimal performance and thus a new cycle is not necessary. Further training in this case would result to overfitting.

### **3.7 Model Validation**

Model validation followed the same approach used in previous works by [17][18][23][36][35]. In this approach, the model is evaluated using the validation split. The aim is to find the best model parameters that give optimal performance and generalize well with the validation split. Then the best model is tested using the testing set and the performance is recorded. This approach was useful since it helped in determining optimal model hyperparameters for both manual tuning and automatic model hyperparameter tuning. The approach is also reliable since the model is not subjected to the testing data, therefore, the model's generalization ability can easily be evaluated based on how well the model performs on the test data.

### **3.8 Model Testing and Evaluation**

Model testing involved subjecting the model to the unseen testing dataset which was created during the training: validation: Test split of the dataset. Testing evaluates how well the model generalizes with the testing dataset. It is the testing results that formed the final model results. The testing approach used here was informed by previous researchers such as [17][18][23][36][35][28] who used the same approach.

Model evaluation is a critical part of the overall research since it helps in knowing how well the model has performed. The researcher used the following metrics to evaluate the performance of the model: Accuracy, Precision, Recall, F1-Score, Quadratic Weighted Kappa Metric, and Area Under the Curve (AUC). The researcher used more than one evaluation metric since researchers such as Qummar et al [17], [18], and [23] have recorded that the imbalanced nature of the Eyepacs dataset makes accuracy an unreliable performance evaluation metric. Although the researcher addressed the problem of data imbalance through augmentation, it was important to guarantee the validity of the model, thus the researcher considered the other evaluation metrics together with accuracy.

Recall also known as sensitivity was used to measure the model's ability to detect positive samples. Precision metric shows the ratio of correctly classified positive samples (True Positive) to a total number of classified positive samples. F1-score shows the harmonic mean between Precision and Recall[93]. Area under the curve depicts the degree of separability between classes, the higher the AUC the better the model has learned. Quadratic weighted Kappa metric illustrates the level of agreement between the predicted label and the ground truth[18]. Chilukoti et al [18] further argues that Quadratic weighted kappa metric is an essential metric in medical classifications since it shows how well two rater agree on the classification.

### **3.9 Tools and Techniques used**

The researcher used the following tools, platforms, and techniques: Google Collaboratory pro with Nvidia GPU, Python programming language, TensorFlow Library, Keras, Keras Tuner, Transfer Learning, VGG16 architecture, HP EliteBook Core i5 laptop, and Google Drive storage. Google Collaboratory pro was chosen

because it offers cloud-based platform notebooks for coding in python. This platform also has a backend Nvidia GPU and high RAM.

### **3.10 Ethical Considerations**

This researcher used secondary dataset and did not use any human respondents all through the research process. The researcher acquired a permit from NACOSTI since the model is expected to be used by people, the researcher also acquired a research approval letter from the Board of Postgraduate Studies (BPS), Murang'a University of Technology. A copy of the NACOSTI Permit is attached as Appendix 5. Also, a copy of the research approval letter from BPS is attached as Appendix 6.

### **3.11 Summary**

This chapter outlines the methodology used to achieve the intended research objectives. The chapter further discusses the research process used starting from choice of best architecture, data acquisition clearly explaining the dimensions of the data, then data pre-processing which involved series of steps taken to prepare the data for training purposes. The chapter further outlines model development which elucidates all the improvements that were done to the model's architecture. This is followed by model training which discusses how the model was trained and how two-level optimization was used. Then model validation which discusses how the model was validated. Finally, model testing and evaluation which shows how the model was tested and the various evaluation metrics that were used in testing the model.



## **CHAPTER FOUR**

### **RESULTS AND DISCUSSION**

#### **4.1 Introduction**

This chapter presents findings of the research conducted. The evaluation metrics discussed in chapter three have been used to analyze performance of the model. The chapter also presents the choice of suitable transfer learning architecture, the algorithm for the improved model, the architecture of the model as well as the hyperparameters that gave the optimal output. The chapter finally discusses the findings and their impact

#### **4.2 Identification of Best Architecture for Transfer learning Model**

The comparative analysis of existing transfer learning architectures was conducted using the algorithms in chapter 3, Algorithm1 shows the deep learning approach in which no pre-trained weights were transferred while Algorithm 2 shows the transfer learning approach which uses weights pre-trained on the ImageNet dataset. The aim of this analysis was to determine which architecture is superior compared to the others in the same environmental setup. The results were as follows:

##### **4.2.1 Model Parameters**

The models exhibited the parameters shown in Table 4.1. These model parameters are obtained from the architecture and can be viewed by printing the model summary. Trainable parameters refer to parameters which can be used to generate model weights. Trainable parameters influence the performance of a model while non-trainable parameters are parameters that cannot change even when training on a new dataset happens.

**Table 4.1: Comparative Analysis of Models' parameters**

Model Architecture	Deep Learning (DL) Approach		Transfer Learning (TL) Approach		Difference between TL and DL trainable parameters
	Total Parameters	Trainable Parameters	Total Parameters	Trainable Parameters	
EfficientNetB0	4, 054, 695	4, 012, 672	4, 059, 815	4, 015, 232	2560
ResNet50	23, 595, 908	23, 542, 788	23, 604, 100	23, 546, 884	4096
DenseNet169	12, 649, 540	12, 491, 140	12, 656, 196	12, 494, 468	3328
VGG16	14, 719, 301	3589	14, 719, 301	14, 718, 277	14,714,688

The parameters show that different architectures have different number of total parameters and number of trainable parameters. This difference is directly related to the number of convolution layers, number of dense layers, and the number of neurons in the dense layers and the convolution layers. Layers such as Max pooling layers do not have any parameters. The more the number of neurons a layer has the more the parameters the layer is going to have. It is also evident that the transfer learning approach increases the number of trainable parameters. This is as a result of the unfreezing of the convolution layers. Therefore, in transfer learning the model is able to easily and quickly learn patterns in the dataset unlike the deep learning architecture. VGG16 demonstrates that much of its trainable parameters are found within the convolution block of the architecture and that's why the deep learning approach has generated very few trainable parameters for VGG16.

#### **4.2.2 Comparison of the performance achieved by the Models**

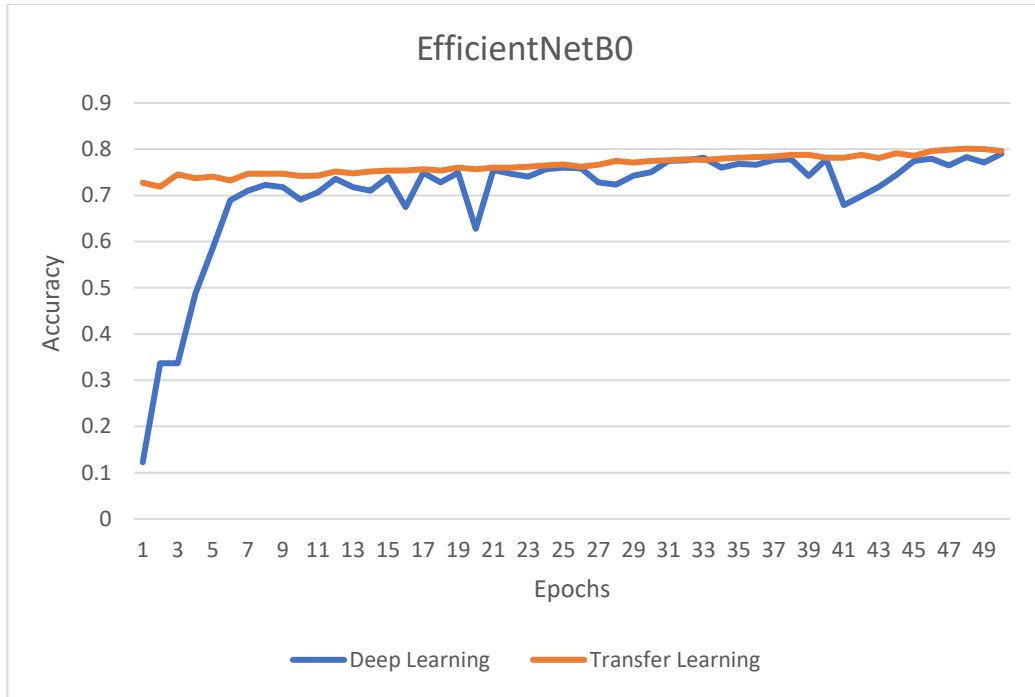
To assess the performance of the architectures, the researchers used accuracy metric. This was chosen as the only metric since the aim was to determine the best architecture

that can be modified to make the final model. The performance of the models was as per the summary presented in Table 4.2.

**Table 4.2: Comparison of the four models in both deep learning and Transfer learning accuracies.**

<b>Model</b>	<b>Deep Learning Accuracy %</b>	<b>Transfer Learning Accuracy %</b>
EfficientNetB0	79.03	80.12
DenseNet169	75.39	83.15
ResNet50	77.70	81.33
VGG16	73.78	84.12

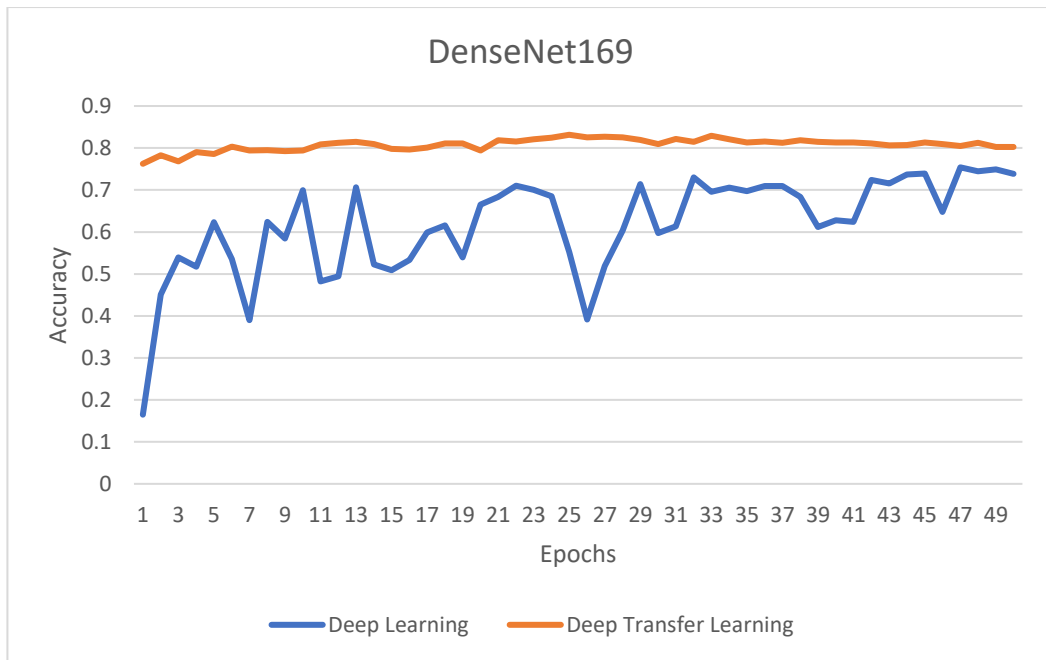
The results of the experiments show that in all models, Transfer learning achieved better performance compared to deep learning. This is mainly due to the fact that, in transfer learning the models were able to leverage on knowledge acquired from the ImageNet dataset. This performance difference has been depicted in the accuracy curves for both approaches. Figure 4.1 represents the accuracy curves for EfficientB0.



**Figure 4.1: EfficientNetB0 Deep Learning and Transfer Learning Accuracies**

From these curves it is evident that transfer learning is able to achieve a high performance within a very few epochs unlike deep learning. EfficientNetB0 managed to achieve an accuracy of 75.15% for the first 15 epochs in transfer learning, while deep learning achieved 73.58% for the same 15 epochs.

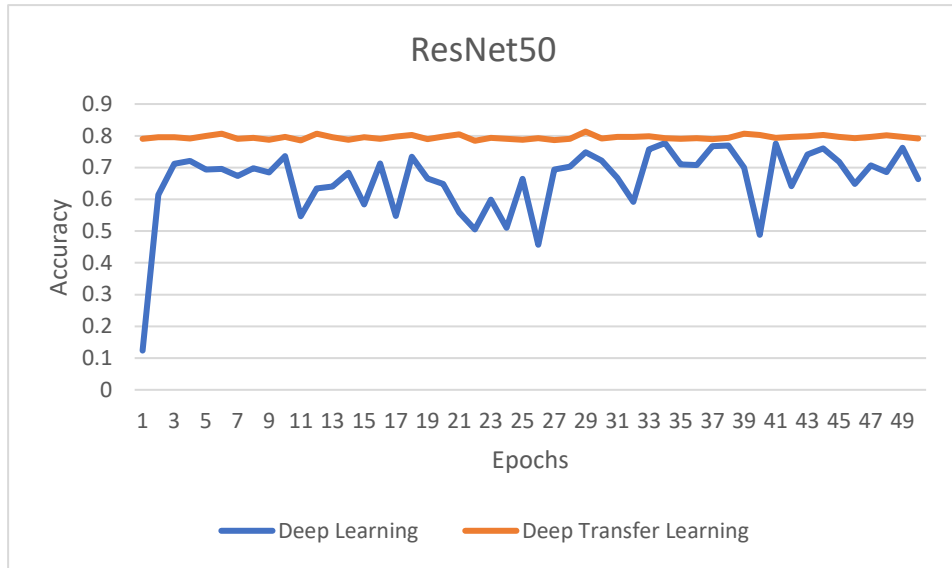
Figure 4.2 shows that DenseNet169 also had a high performance in transfer learning all through the training cycle.



**Figure 4.2: DenseNet169 Deep Learning and Transfer Learning Accuracies**

Figure 4.2 shows that within the first 15 epochs the model was able to achieve a validation accuracy of 81.45% for transfer learning while deep learning achieved a validation accuracy of 70.6%. The power of transfer learning in this model has also been depicted by the high validation accuracy of 76.24% achieved within the first epoch of transfer learning as compared to deep learning which achieved 16.48% validation accuracy.

Figure 4.3 shows ResNet50 Deep Learning and Transfer Learning Accuracy curves.

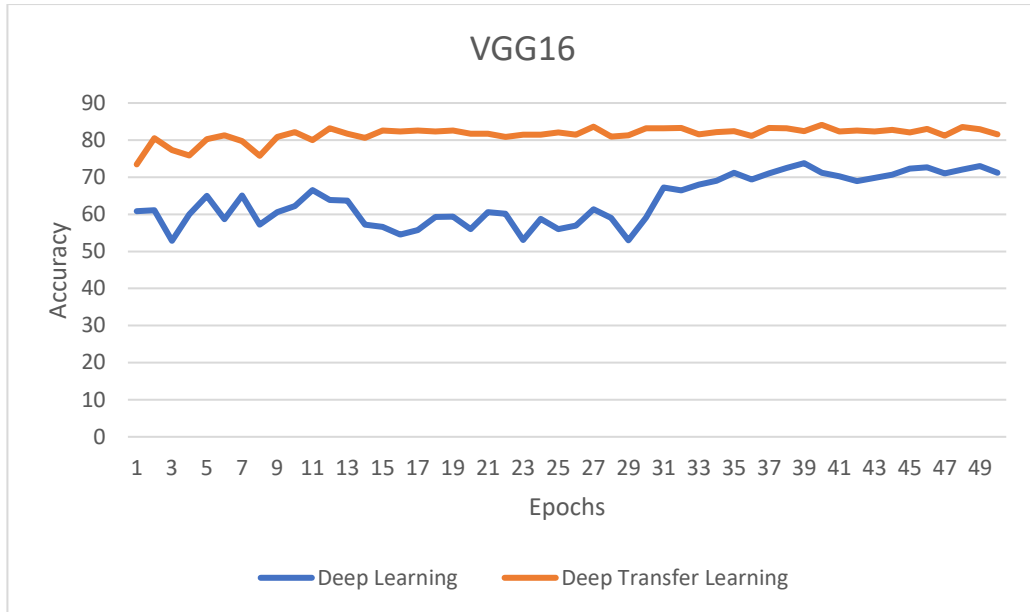


**Figure 4.3: ResNet50 Deep Learning and Transfer Learning Accuracy Curves**

Figure 4.3 shows that ResNet50 achieved an accuracy of 79.03%, and 12.3% for transfer learning and deep learning respectively in the first epoch. The accuracies registered with the first 15 epochs were 80.61% and 73.58% for deep learning and transfer learning respectively.

VGG16 represented in Figure 4.4 also showed a similar pattern to ReseNet50, DenseNet169, and EfficientNetB0. The model achieved an accuracy of 60.85% for deep learning and 73.5% for transfer learning in the first epoch. In the first 15 epochs deep learning managed to achieve an accuracy of 66.5% while transfer learning achieved an accuracy of 83.15%.

Figure 4.4 shows Deep learning and Transfer Learning accuracy curves



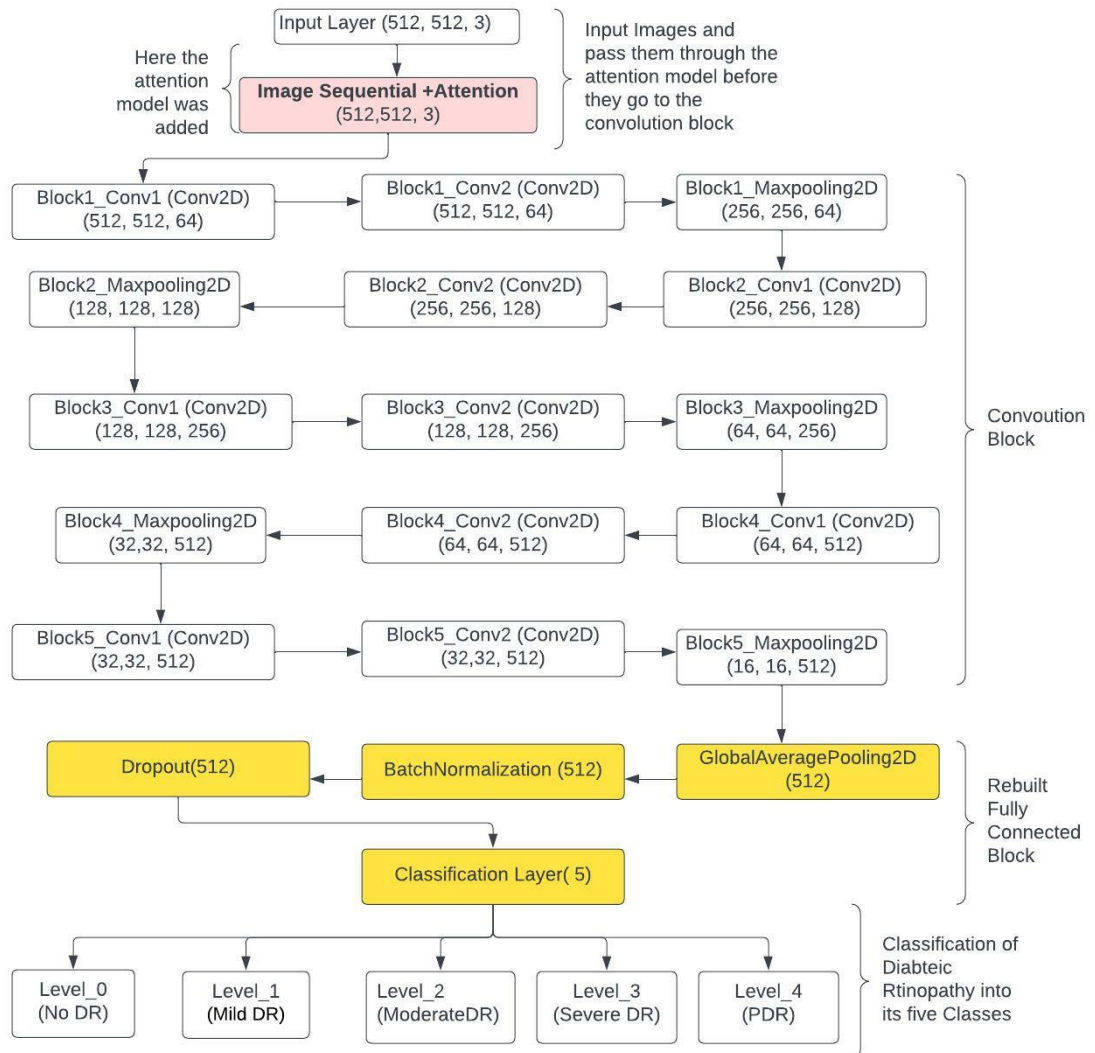
**Figure 4.4: VGG16 Deep Learning and Transfer Learning Accuracy Curves**

The results also demonstrate that the Visual Geometry 16 network (VGG16) model achieved the highest accuracy of 84.12%. This high performance of the VGG16 model over the other models can be attributed to its simple yet powerful architecture which is able to properly leverage on the convolution block and the fully connected block. Therefore, if VGG16 is modified and optimized it can achieve great performance in classifying diabetic retinopathy into its 5 classes. This has been demonstrated by [23] who added a NiN layer to VGG16 and the model was able to achieve 87% accuracy, Quadratic weighted Kappa metric of 0.85, F1-score of 0.84, precision of 0.85, and recall of 0.87. This performance of VGG16 justified the researcher’s choice to use it.

### 4.3 Modified Model Architecture

This section presents the architecture of the modified model which is part of the contributions that this research made. The modifications made have been highlighted in Figure 4.5. The model architecture has an attention model as part of the image

sequential pipeline. It also has a rebuilt top layer which has eliminated the original two dense layers of the VGG16 and replaced the output layer with a custom output layer.



**Figure 4.5: Modified Model Architecture**

#### 4.4 Modified Model Algorithm

Algorithm 3 represents the overall model’s algorithm that was used in the model. This algorithm is also part of the researcher’s contribution. The algorithm depicts where two-level optimization is happening during model training as well as the attention layer/mechanism added to the model’s architecture



Input → Fundus Images belonging to 5 classes

Output → A model that classifies fundus images into the five classes

- 1) Load the EyePacs dataset
- 2) Data Pre-processing
  - Reshape image to (512, 512, 3), Split 70% training, 10% validation, 20% testing
  - Argument Minority classes only (To address data Imbalance)
    - ❖ Random Flip
    - ❖ Random Rotation
    - ❖ Random Zoom (Height\_factor(-0.2, 0.3), Width\_factor(-0.2, -0.3))
    - ❖ Random Contrast (factor 0.1)
    - ❖ Random Translation (height\_factor(0.1), width\_factor (0.1))
- 3) Import pre-trained VGG16 architecture+ ImageNet weights
- 4) Modify the Architecture (Add Attention layer, rebuild top-layer, Freeze the model)
- 5) Set Initial model hyperparameters
- 6) Do feature extraction

While initial batchsize=8 & Model.Trainable=False (The model is frozen)

For epochs 1 to 40 do

- Automatically choose optimal learning rate from the range (1e-2 to 1e-6)
- Update model parameters ( $X_{train}$ ,  $Y_{validation}$ ),
- Evaluate model performance

End

End

- 7) Model training after feature extraction

While model.trainable=True (Unfreezing the model)

Set batch\_size=16, 32, and 64 for sessions 1, 2, and 3 respectively

For session 1 to 3 & epoch 1-40 for session 1, and 1-20 sessions 2-3 do

- Automatically choose optimal learning rate ( $\alpha$ ) from the range (1e-2 to 1e-6) for each gradient step.

While  $\alpha$ = optimal

- Update model parameters ( $X_{train}$ ,  $Y_{validation}$ ),
- Evaluate model's ( $X_{train}$ ,  $Y_{validation}$ ) performance (Acc, Auc, QWK, F1, Recall, Precision)
- Choose best model performance

If Model ( $X_{train}$ ,  $Y_{validation}$ ) performance = Best

- ✓ Apply the model to the test dataset
- ✓ Record Performance

End

End

End

End

*Two-Level  
Optimization*

### Algorithm 3: Algorithm for the Improved Model

#### 4.5 Summary of hyperparameters after Tuning

Table 4.3 shows the results of the hyperparameters that were obtained after hyperparameter tuning was done using a combination of hand-crafted hyperparameters and Keras tuner. Keras tuner was used since it fastens the process of hyperparameter search in TensorFlow platform. Handcrafting was used for

hyperparameters that could not be tuned using Keras tuner such as the progressive batch size, and the learning rate range.

**Table 4.3: Final Hyperparameters Obtained After Tuning**

<b>Hyperparameter</b>	<b>Value</b>
Drop out	0.2
Input size	512*512
Input channels	3
Gradient Clipping	1.0
Number of Dense layers	1
Number of Neurons in the Dense layer	5
Area Under the curve (AUC) Threshold	200
Dataset set split random seed	23
Steps per epoch	792
Mini_delta	0.01
Batch size	8, 16, 32, 64
Learning rate	Range (1e-2 to 1e-6)

During the hyperparameter optimization process, the researcher discovered that there is a direct relationship between hyperparameters chosen and the performance of the model. Therefore, the hyperparameters presented in table 4.3 are the ones that gave the best performance of the model.

#### **4.6 Results of the improved model's performance**

This section shows how the improved model performed on the Eyepacs dataset. The results have been divided into four main sections to show results for training top layer only, training session 1, training session 2, and training session 3.

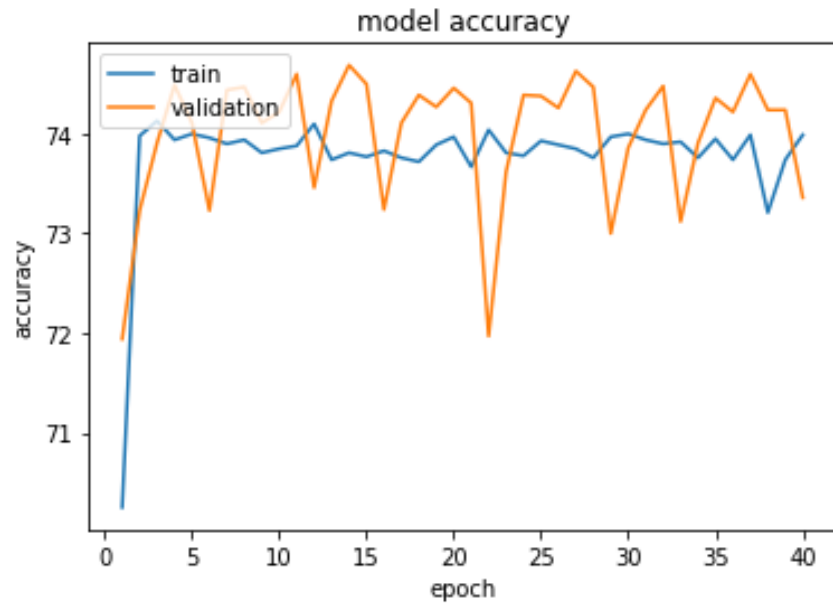
#### 4.6.1 Top layer Training

Table 4.4 shows the results obtained after training the top layer only while the other parts of the model were frozen. These results form the foundational results which show the model's capability to leverage on previous knowledge.

**Table 4.4: Results of Training the Top Layer Only**

<b>Performance Evaluation Metric</b>	<b>Score Achieved</b>
Accuracy	74.32%
Precision	74.92%
Recall	73.95%
F1-Score	26.93%
Quadratic Weighted Kappa Metric	0.3

Figure 4.6 shows the top layer training and validation curves for the 40 epochs. The curve depict that the model was able to extract relevant high-level features from the dataset using the top layer. The curve also depicts that the training almost achieved a flat curve with minimal changes and thus further training without unfreezing the convolutional block would not have improved performance. The aim of freezing the other parts of the model was to enable the model to extract high-level features from the EyePACS dataset.



**Figure 4.6 Top Layer Training and Validation Curve**

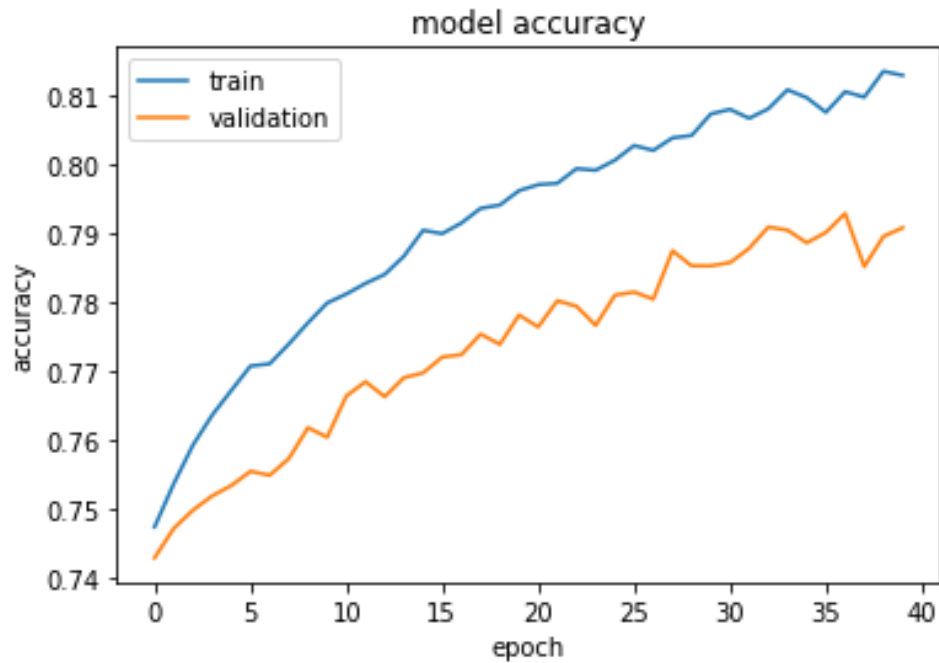
#### 4.6.2 Full Model Session one

Table 4.5 shows the results obtained from the first cycle of full model training, the results have a very significant improvement compared to the results obtained by training the top-layer of the model only. The F1-score registered that highest improvement from 26.93% to 41.30%, accuracy improved by 4.91%, precision improved by 5.24%, recall improved by 4.47%, and Quadratic weighted kappa metric improved by 0.23.

**Table 4.5: Results of the First Session of Training the Full Model**

<b>Performance Evaluation Metric</b>	<b>Score Achieved</b>
Accuracy	79.23%
Precision	80.16%
Recall	78.42%
F1-Score	41.30%
Quadratic Weighted Kappa Metric	0.53
Area Under the ROC Curve (AUC)	84.26%

Figure 4.7 shows the training and validation curves obtained after training the full model for the first session. The curve demonstrate that unfreezing of the convolutional block helped in transferring convolutional weights gained from ImageNet into DR detection task. The curves also show that the by training the entire architecture the model is able to achieve higher performance than when training the top layer only.



**Figure 4.7: Full Model Session 1 Training and Validation Curves**

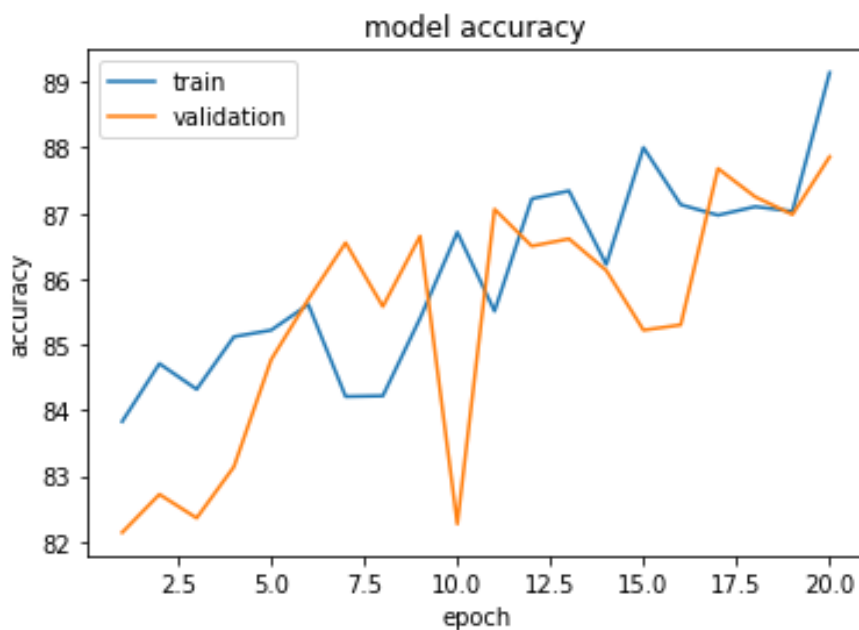
### 4.6.3 Full model Second Session

Table 4.6 shows the results that the model achieved from the second training cycle. The results though not yet optimal, demonstrate that the model is able to improve on its performance. Accuracy improved by 6.11%, precision improved by 4.3%, recall improved by 7.96%, F1-Score improved by 24.58%, Quadratic weighted Kappa metric improved by 0.26, Area Under the Curve (AUC) improved by 8.24%.

**Table 4.6: Results of Training the Full Model for the Second Session**

Performance Evaluation Metric	Score Achieved
Accuracy	85.34%
Precision	84.46%
Recall	86.38%
F1-Score	65.88%
Quadratic Weighted Kappa Metric	0.79
Area Under the ROC Curve (AUC)	92.5%

Figure 4.8 shows the training and validation curves obtained after training the full model for the second session. The curve demonstrates considerable improvement from the previous training sessions.



**Figure 4.8: Full Model Session Two training and validation curves**

#### 4.6.4 Full model Third Session

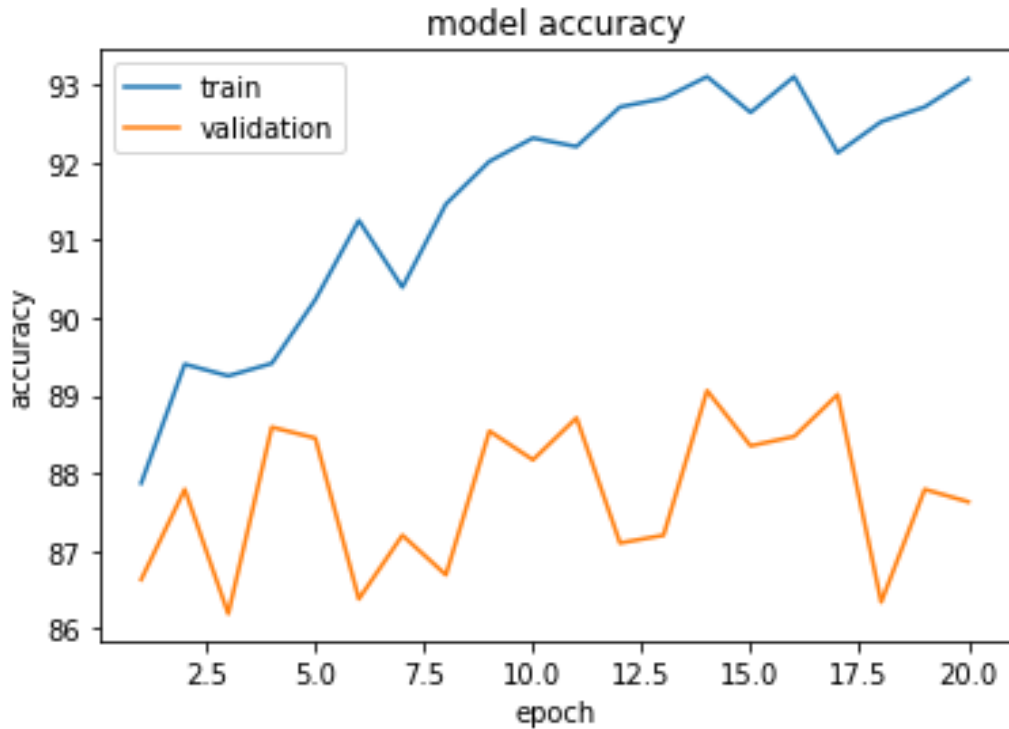
Table 4.7 shows the performance results that the model achieved during the third training session. The results show evidence of improvement from the previous training cycle. Accuracy improved by 2.89%, precision improved by 3.64%, 2.22%, F1-Score improved by 7.58%, Quadratic weighted kappa metric improved by 0.05, and AUC improved by 0.8%.

**Table 4.7: Results of Training the Full Model for the Third Session**

<b>Performance Evaluation Metric</b>	<b>Score Achieved</b>
Accuracy	89.06%
Precision	88.9%
Recall	89.2%
F1-Score	75%
Quadratic Weighted Kappa Metric	0.84
Area Under the ROC Curve (AUC)	93.3%

Figure 4.9 shows the training curve obtained after training the full model for the third session. The third curves show considerable improvement in performance from the previous training sessions. The curve also shows that the model has achieved optimal performance since the validation curve has stopped improving and the highest points almost flattened.





**Figure 4.9: Full Model Session Three training and validation curves**

#### **4.7 Comparison of the Results with the existing models**

Table 4.8 shows a comparison of the results obtained by the Improved model against the other existing models in literature. In this comparison we only considered models that classified diabetic retinopathy into five classes. Therefore, models such as the ones developed by Charu, Jain, and Sood [23], Shenavarmasouleh et al., [32], Hagos and Kant [19], were omitted since they classify diabetic retinopathy into four, three, and two classes respectively. From the table it is evident that the new model has higher performance compared to the existing models. The blank cells in the table means that the researcher did not use the evaluation metric in that column as part of their evaluation metrics. For instance, Kahn et al [23] did not use QWK while Jinfeg et al [36] did not use precision.

**Table 4.8: Comparison of Existing Models to the Improved Model**

Model	Evaluation Metrics					
	Accuracy (%)	QWK	F1 Score (%)	Precision (%)	Recall (%)	AUC (%)
Qummar et al [17]	80.8		53.74	63.85	51.5	91.0
Pratt et al [35]	75		41.6		30	
Jinfeng et al [36]	80.36				47.70	89.5
Khan et al [23]	85		59.6	67	55.6	89.5
Chilukoti et al [18]	87	0.85	84	85	87	
Bodapti et al. [114]	81.7	0.70	80	80	81	
Kassani et al [115]	83.09				88.24	91.8
Dondeti et al [116]	77.90		75	76	77	
Bodapti et al.[117]	82.54		82	82	83	79
Proposed Model	89.06	0.84	75	88.9	89.2	93.3

#### 4.8 Discussion

The main objective of this study was to develop an improved model for diabetic retinopathy classification using transfer learning approach and two-level optimization of hyperparameters. The findings of the study demonstrate that transfer learning which is one of the meta-learning techniques is a powerful approach that enables models to leverage on the previously acquired knowledge in solving a new task. This powerful capability of transfer learning has been demonstrated not only in this study but also in the related works. Table 4.8 shows that Pratt et al [35] who trained a CNN model has achieved the lowest performance compared to all the others who have used transfer learning.

##### 4.8.1 Identification of Best Architecture for Transfer learning Model

Based on the results, transfer learning has demonstrated capability to deliver models that are scalable, robust, less costly in terms of training time, and able to learn from limited dataset and achieve high performance[118][69]. These capabilities directly

relate to the fact that in transfer learning a model learns general knowledge then optimizes to adopt the knowledge acquired to a specific task [13][119]. For instance, in Image classification, a meta-learning model learns key aspects of image classification such as feature extraction and weighting then transfers this knowledge and applies it to a specific task like Diabetic retinopathy classification.

The findings also show that the various existing neural network architectures that can be used for transfer learning have different capabilities. This difference in capabilities is influenced by the internal structures of different architectures, for instance DenseNets use densely connected networks, EfficientNets use balancing of weight, height, and contrast, while VGG Nets use propagation through a combination of convolutional blocks and fully connected block. VGG16 proved to be more superior than the others in diabetic retinopathy classification. This superiority was as a result of the powerful sequence of convolutional blocks coupled by the fully connected block that has capabilities to map the output of convolutional block and prepare them for final classification.

The findings further demonstrate that the default architecture of VGG16 may not give optimal performance in some task due to dataset differences. Therefore, there is need to modify the architecture in order to achieve high performance. This fact has been substantiated by other researchers such as [23] who added a spatial pooling layer to VGG16 to boost its performance. In this study, modifying the architecture by adding an attention layer and rebuilding the top layer played a critical role in attaining the superior performance[57].

The findings also depict that model optimization plays a very critical role in determining model performance. A good optimization approach in Meta-learning

should be one in which a model has some aspect of self-optimization influenced by the dataset used [118][69]. Two-level optimization used in this study gave the model capabilities to have learning and data-driven optimization through automatically selecting a suitable learning rate. This made it possible for the model to achieve high performance and continuously improve its performance from one session to another.

The batch size used has significant influence on the performance of a model. Kandel and Castelli [112] argue that it is ideal to start with a small batch size during initial stages of the model training so that the model can extract important features from the data. However, this batch size must be mapped with a suitable learning rate for good performance to be achieved. Large batch sizes affect generalizability while small batch size take longer to train[113]. Therefore, there is need to balance between batch size and learning rate to achieve optimal performance. This study used a progressive batch size and autotuning of the learning rate under the two-level optimization approach, thus, for each batch size the model was able to choose its optimal learning rate, therefore, resulting to the high performance recorded by the model.

The findings also depict that training data distribution affects performance of the model. Imbalanced data does not give reliable accuracy while balanced dataset can provide more reliable accuracy[36]. This research did some data balancing by augmenting the minority classes. The impact of this balancing has been depicted in the performance since the model achieved a higher accuracy, precision, recall and AUC as compared with existing works such as [36].

#### **4.8.2 Top layer only**

The results of training the top layer demonstrate that in transfer learning, models are able to extract critical features from the dataset through the top layer. The results also

show that network architecture is powerful since the model was able to achieve a good performance in this training session. Also, by training the rebuilt top-layer only while the other parts were frozen, the model was able to prepare the top-layer to now utilize weights obtained from ImageNet in the subsequent training sessions. The flat training and validation curves depict that the model managed to achieve the climax of top-layer training and thus there was need to engage the other layers in the model.

#### **4.8.3 Full model Session one**

The results obtained after training the full model for the first session depict that; by using a small batch size of 8 the model was able to leverage on knowledge acquired from ImageNet and utilized it in solving diabetic retinopathy detection task. Also, the training and validation curves depict that by unfreezing the convolutional block the model was able to overcome the flat curve in Top-layer training only. Further, the results show the power of transfer learning since in this first session of transfer learning, the model has already surpassed models such as Pratt et al [35].

#### **4.8.4 Full Model Session Two**

The results of training the full model for the second session demonstrate a considerable improvement from the previous training session. This can be attributed to the concepts progressive batch size and Two-level optimization that the researcher employed in this research. Two-level optimization has enabled the model to choose the best learning rate for batch size 32. The training and validation curves have some rigorous ups and downs. This depicts that the model was adjusting the learning rate to fit the batch size almost after every 1.5 epochs. This also shows that at batch size 32 the model had obtained sufficient features from the dataset and now the model was now seeking for convergence. According to Kandel and Castelli's [112] at this point the model desires a larger batch size so that it converge.

#### **4.8.5 Full Model Session Three**

The findings show that in this final session optimal performance was achieved. This is depicted by the training and validation curves which show that the validation curve stopped improving. The increased gap between the training and validation curves show that further training was not necessary since it would not improve the model. The findings also show that the new improved model is more superior than the existing model.

This is so since the model has been able to achieve domain-shift generalizability by optimizing the weights obtained from ImageNet to fit the diabetic retinopathy classification task. This was affirmed by validating the model using the validation data set and finally testing the model using the testing dataset. The model demonstrated capability to differentiate between the five classes of diabetic retinopathy as shown by the high AUC achieved. The model also achieved a good quadratic weighted Kappa metric of 0.84 which shows that there is almost perfect agreement between the classified labels and ground truth.

Although Chilukoti et al [18] achieved a higher QWK of 0.85 which is 0.01 higher compared to the proposed model, the difference is insignificant since in both cases the QWK's are above the 0.81 threshold of almost perfect agreement. Chilukoti et al [18] has also achieved a higher F1-score compared to the proposed model. However, the proposed model has surpassed Chilukoti et al [18] in model's accuracy by 2.06%, model's precision by 3.9% and model's recall by 2.2%. Therefore, the overall performance of the proposed model is better than Chilukoti et al [18]. Also, Chilukoti et al [18] has not considered the degree of separability between classes.

The high recall of 89.2% (True Positive rate) achieved by the model shows that the model has very low chances of failing to detect a patient with diabetic retinopathy. In terms of recall the model has surpassed the second best which is Kassani et al [115] by 0.96% which is significant difference. The model has also achieved a high Area Under the Curve (AUC) of 93.3%. This high AUC shows that the model is able to clearly separate between the five classes of diabetic retinopathy. The model has defeated other models in literature in terms of AUC. The difference between the proposed model's AUC and the second-best AUC in literature is 1.5%.

The model achieved an accuracy of 89.06% which is 2.06% higher than the first runners up in literature. The high accuracy achieved by the model shows that the model is better in classifying unseen data into the five classes of DR compared to other existing models in literature. The model also achieved a high precision of 88.9% which is 2.9% higher than the second-best model. The high precision depicts that the model was able to achieve a good ratio of truly classified samples against the true positives and false positives.

The combination of high AUC, high recall, high QWK, high accuracy, and high precision shows that the model can correctly classify a patient's fundus image into the rightful class of diabetic retinopathy. This surpasses models such as the ones developed by [28] and [19] which can only classify a patient's fundus image as either healthy or unhealthy, thus they can only be used for detection of diabetic retinopathy and cannot be used for categorization of diabetic retinopathy. It also surpasses other models that can do both tasks in terms of how perfectly they do the task as shown in Table 4.8.

## **4.9 Summary**

This chapter presents the results of the research conducted. The chapter starts by identifying the best architecture for transfer learning through a comparative analysis. VGG16 emerged as the best architecture for transfer learning and thus it was modified to suit the DR detection task. The results of the modified VGG16 which depicts the new architecture after modification as well as the modified model algorithm have also been presented. The chapter has provided a summary of hyperparameters that were obtained during hyperparameter tuning.

The model was trained for four sessions and the results for each training session have been provided in this chapter. This is followed by a comparison of the proposed model with existing models. The results depict that the proposed model has achieved superior performance compared to existing models in literature. The chapter ends with a discussion section. This section discusses the results obtained in this research and how they compare to what is already existing.



## CHAPTER FIVE

### CONCLUSION, RECOMMENDATIONS AND FUTURE WORK

#### 5.1 Conclusion

This study aimed at developing a Transfer learning and two-level hyperparameter optimization-based model for improved detection of diabetic retinopathy. To achieve this the researcher formulated three specific objectives and three research questions.

The first objective was to analyze existing deep learning and transfer learning models for classification of diabetic retinopathy. To achieve this, the researcher conducted a literature review of existing studies that have used transfer learning approaches or deep learning to classify diabetic retinopathy. The obtained research works were further grouped into two, group A consisted of those that classified diabetic retinopathy into its five distinct classes namely No. DR, Mild DR, Moderate DR, Severe DR, and Proliferative DR. Group B on the other hand consisted of the works that did not classify DR into the five classes. The researcher focused on group A of the related works. EfficientNet, VGGNets, DenseNets, ResNets were identified as the most used network architectures for transfer learning.

The researcher further did a comparative analysis of these architectures. The analysis involved training the architectures using the same dataset, same hyperparameters, and same environment setup. The aim was to determine which architecture is superior than the rest. VGG16 emerged as the best performer and thus it was selected as the architecture of choice in this study.

The second objective involved redesigning the transfer learning model identified in objective one and tuning its training hyperparameters using two-level optimization. Here the VGG16 architecture was modified to include an attention module, Batch normalization layer, Global average pooling layer, and dropout layer. The

classification layer was also modified to classify DR into five classes. The model was then trained using the EyePACS dataset and applying two-level optimization.

The third objective was to validate and test the model. Model validation was done using the validation dataset while model testing was done using the testing dataset. The testing dataset was unseen by the model since when the dataset was split to 70% training, 10% validation and 20% testing, the model was only subjected to the training and validation splits. Then the best performing model scenario based on model checkpoints was subjected to the unseen testing data. Therefore, the final test results obtained by this model are results of unseen testing data. This is the industry standard and has been practiced by other researchers such as [17][23][18][116][114]

The final results obtained show that the improved model is superior than the existing ones. The model managed to achieve the following in accuracy, quadratic weighted kappa metric, F1 score precision, recall, and area under the curve respectively; 89.06%, 0.84, 75%, 88.9%, 89.2%, 93.3%. The high performance of the model is attributed to the modified structure of the neural network, data pre-processing, and two-level optimization.

This study has also demonstrated that meta-learning techniques which include transfer learning are powerful compared to other approaches since they are able to deliver a model that is robust. This has been demonstrated by the ability of the improved model to achieve domain shift generalizability. i.e the model was pretrained using ImageNet dataset and the knowledge in form of weights was transferred to solving the diabetic retinopathy classification task and achieved superior performance.

## **5.2 Recommendations**

The scope of this research is in machine learning, more specifically neural networks. Achieving the results obtained in this research requires high performance computers with powerful GPUs that can-do parallel processing. Such resources were not available and thus the researcher turned to using cloud-based resources in Google Collaboratory. The challenges encountered in this research influenced the researcher to make the following recommendations.

### **5.2.1 The Ministry of health to collect and maintain a database of medical datasets**

The ministry of health should collect medical data from various hospitals in Kenya, store the data, and maintain it for purposes of usage in machine learning. This will help researchers in machine learning to apply machine learning methods in addressing major medical conditions affecting Kenyans. This will result to solution that are custom-made to address health challenges in Kenya and also suitable for adoption in Kenyan hospitals. The homegrown dataset will also reduce reliance on secondary datasets obtained from other countries. This research used the EyePacs dataset which is collected by EyePacs organization based in USA. Therefore, having local publicly available dataset will be a good path towards localizing solutions.

### **5.2.2 Adoption of developed models**

There is a lot of research in machine learning that has been done across the learning institutions in the country. However, the findings of these research remain unused despite the value they would deliver. The government as well as the private sector should consider adopting machine learning models that deliver desirable solutions in their domain. This will help the country in leveraging on the power of machine

learning to address some of the issues it faces not only in healthcare but also in other fields such as agriculture.

### **5.2.3 Facilitation of postgraduate students**

Publishing open access in high impact factor journals requires a lot of money. Postgraduate researchers should be facilitated to publish in high impact factor journals so that their research work can reach out to the global research community. Also, the government in partnership with the institution should provide postgraduate researchers with stipends to cater for other research costs such as printing, software subscription, conference fees, among others.

## **5.3 Future work**

This research focused on what was specified in the study objectives and the scope of the study. Also, other factors such as time and resource constraints inhibited further extension of this model. Therefore, there are some issues that are still open for further research as a way of improving this study. Some of the future works that the researcher identified include the following.

### **5.3.1 Use of another type of machine learning**

This study used supervised learning which uses labeled data. Supervised learning is the most used type of machine learning and has been used by other researchers in literature such as [115][17]. Therefore, it has been considered to be the best compared to others in terms of ease of training. The literature also documents that there are other types of machine learning such as semi-supervised learning which uses partly labeled data, unsupervised learning which uses unlabeled data, and reinforcement learning which uses a reward system. The researcher proposes that in future other researchers can use other types of machine learning in classifying diabetic retinopathy.

### **5.3.2 Increase Multi-task capabilities of the model**

This model focused on classification of diabetic retinopathy. The researcher proposes that in future researchers can use multi-task learning to extend the model to detect other eye diseases. This will increase the robustness of the model since a single model will be useful in detecting and classifying multiple eye disorders. This will also be a path towards Artificial General Intelligence (AGI).

### **5.3.2 Creating a G.U.I to facilitate deployment of this model**

The researcher proposes that in future other researchers can use deployment frameworks to create a suitable graphical user interface and deploy the model for usage. This would make the model to be patentable and ready for use in the hospitals. It would as well enhance the portability of the model and compatibility of the model across different platforms.

## REFERENCES

- [1] R. Bhardwaj, A. R. Nambiar, and D. Dutta, “A Study of Machine Learning in Healthcare,” *Proc. - Int. Comput. Softw. Appl. Conf.*, vol. 2, pp. 236–241, 2017, doi: 10.1109/COMPSAC.2017.164.
- [2] “What is Machine Learning? | IBM.”  
<https://www.ibm.com/cloud/learn/machine-learning> (accessed Jul. 06, 2021).
- [3] S. S. Dash, S. K. Nayak, and D. Mishra, “A review on machine learning algorithms,” in *Smart Innovation, Systems and Technologies*, 2021, vol. 153, no. 10, pp. 495–507, doi: 10.1007/978-981-15-6202-0\_51.
- [4] A. E. Ezugwu *et al.*, “A comprehensive survey of clustering algorithms: State-of-the-art machine learning applications, taxonomy, challenges, and future research prospects,” *Engineering Applications of Artificial Intelligence*, vol. 110, p. 104743, Apr. 01, 2022, doi: 10.1016/j.engappai.2022.104743.
- [5] K. M. Mendez, S. N. Reinke, and D. I. Broadhurst, “A comparative evaluation of the generalised predictive ability of eight machine learning algorithms across ten clinical metabolomics data sets for binary classification,” *Metabolomics*, vol. 15, no. 12, pp. 1–15, 2019, doi: 10.1007/s11306-019-1612-4.
- [6] M. R. Hassan, S. Huda, M. M. Hassan, J. Abawajy, A. Alsanad, and G. Fortino, “Early detection of cardiovascular autonomic neuropathy: A multi-class classification model based on feature selection and deep learning feature fusion,” *Inf. Fusion*, vol. 77, pp. 70–80, Jan. 2022, doi:

10.1016/j.inffus.2021.07.010.

- [7] H. H. Sultan, N. M. Salem, and W. Al-Atabany, “Multi-Classification of Brain Tumor Images Using Deep Neural Network,” *IEEE Access*, vol. 7, pp. 69215–69225, 2019, doi: 10.1109/ACCESS.2019.2919122.
- [8] Z. Hu, J. Tang, Z. Wang, K. Zhang, L. Zhang, and Q. Sun, “Deep learning for image-based cancer detection and diagnosis – A survey,” *Pattern Recognit.*, vol. 83, pp. 134–149, Nov. 2018, doi: 10.1016/J.PATCOG.2018.05.014.
- [9] H. Pratt, F. Coenen, D. M. Broadbent, S. P. Harding, and Y. Zheng, “Convolutional Neural Networks for Diabetic Retinopathy,” *Procedia Comput. Sci.*, vol. 90, no. July, pp. 200–205, 2016, doi: 10.1016/j.procs.2016.07.014.
- [10] H. Polat and H. D. Mehr, “Classification of pulmonary CT images by using hybrid 3D-deep convolutional neural network architecture,” *Appl. Sci.*, vol. 9, no. 5, 2019, doi: 10.3390/app9050940.
- [11] R. Ashmore, R. Calinescu, and C. Paterson, “Assuring the Machine Learning Lifecycle: Desiderata, Methods, and Challenges,” *ACM Computing Surveys*, vol. 54, no. 5. 2021, doi: 10.1145/3453444.
- [12] S. Kaya, “Denetimli Makine Öğrenme Algoritmaları Önce ve Sonra Performans Karşılaştırılması: Bir Örnek : Göğüs Kanseri Tespiti An Example of Performance Comparison of Supervised Machine Learning Algorithms Before and After PCA an,” pp. 0–5, 2020.
- [13] C. Finn, P. Abbeel, and S. Levine, “Model-agnostic meta-learning for fast

- adaptation of deep networks,” *34th Int. Conf. Mach. Learn. ICML 2017*, vol. 3, pp. 1856–1868, 2017.
- [14] T. M. Hospedales, A. Antoniou, P. Micaelli, and A. J. Storkey, “Meta-Learning in Neural Networks: A Survey,” *IEEE Trans. Pattern Anal. Mach. Intell.*, pp. 1–20, 2021, doi: 10.1109/TPAMI.2021.3079209.
- [15] T. Gressling, *84 Automated machine learning*. 2020.
- [16] I. P. Possebon, A. S. Silva, L. Z. Granville, A. Schaeffer-Filho, and A. Marnerides, “Improved Network Traffic Classification Using Ensemble Learning,” *Proc. - IEEE Symp. Comput. Commun.*, vol. 2019-June, 2019, doi: 10.1109/ISCC47284.2019.8969637.
- [17] S. Qummar *et al.*, “A Deep Learning Ensemble Approach for Diabetic Retinopathy Detection,” *IEEE Access*, vol. 7, pp. 150530–150539, 2019, doi: 10.1109/ACCESS.2019.2947484.
- [18] and D. X. H. Sai Venkatesh Chilukoti, Dr. Anthony S Maida, “Diabetic Retinopathy Detection using Transfer Learning from Pre-trained Convolutional Neural Network Models,” *IEEE J. Biomed. Heal. INFORMATICS*, vol. 20, pp. 0–10, 2022, doi: 10.36227/techrxiv.18515357.v1.
- [19] M. T. Hagos and S. Kant, “Transfer Learning based Detection of Diabetic Retinopathy from Small Dataset,” 2019.
- [20] S. Kusuhara, Y. Fukushima, S. Ogura, N. Inoue, and A. Uemura, “Pathophysiology of diabetic retinopathy: The old and the new,” *Diabetes Metab. J.*, vol. 42, no. 5, pp. 364–376, 2018, doi: 10.4093/dmj.2018.0182.



- [21] W. Wang and A. C. Y. Lo, “Diabetic retinopathy: Pathophysiology and treatments,” *Int. J. Mol. Sci.*, vol. 19, no. 6, 2018, doi: 10.3390/ijms19061816.
- [22] “What Is Diabetic Retinopathy? - American Academy of Ophthalmology.” <https://www.aao.org/eye-health/diseases/what-is-diabetic-retinopathy> (accessed Jul. 06, 2021).
- [23] Z. Khan *et al.*, “Diabetic Retinopathy Detection Using VGG-NIN a Deep Learning Architecture,” *IEEE Access*, vol. 9, pp. 61408–61416, 2021, doi: 10.1109/ACCESS.2021.3074422.
- [24] A. D. Lewis *et al.*, “Prevalence of diabetic retinopathy and visual impairment in patients with diabetes mellitus in Zambia through the implementation of a mobile diabetic retinopathy screening project in the Copperbelt province: a cross-sectional study,” *R. Coll. Ophthalmol.*, vol. 32, pp. 1201–1208, 2018, doi: 10.1038/s41433-018-0055-x.
- [25] E. S. Elwali, A. O. Almobarak, M. A. Hassan, A. A. Mahmood, H. Awadalla, and M. H. Ahmed, “Frequency of diabetic retinopathy and associated risk factors in Khartoum, Sudan: Population based study,” *Int. J. Ophthalmol.*, vol. 10, no. 6, pp. 948–954, 2017, doi: 10.18240/ijo.2017.06.18.
- [26] A. Bastawrous *et al.*, “The incidence of diabetes mellitus and diabetic retinopathy in a population-based cohort study of people age 50 years and over in Nakuru , Kenya,” *BMC Endocr.*, pp. 1–14, 2017, doi: 10.1186/s12902-017-0170-x.
- [27] C. Bhardwaj, S. Jain, and M. Sood, “Transfer learning based robust automatic detection system for diabetic retinopathy grading,” *Neural Comput. Appl.*, vol. 2, 2021, doi: 10.1007/s00521-021-06042-2.

- [28] G. Algan, I. Ulusoy, Ş. Gönül, B. Turgut, and B. Bakbak, “Deep Learning from Small Amount of Medical Data with Noisy Labels: A Meta-Learning Approach,” 2020.
- [29] M. Mateen, J. Wen, N. Nasrullah, S. Sun, and S. Hayat, “Exudate Detection for Diabetic Retinopathy Using Pretrained Convolutional Neural Networks,” *Complexity*, vol. 2020, 2020, doi: 10.1155/2020/5801870.
- [30] B. Tymchenko, P. Marchenko, and D. Spodarets, “Deep learning approach to diabetic retinopathy detection,” *ICPRAM 2020 - Proc. 9th Int. Conf. Pattern Recognit. Appl. Methods*, pp. 501–509, 2020, doi: 10.5220/0008970805010509.
- [31] A. S. Thiagarajan, J. Adikesavan, S. Balachandran, and B. G. Ramamoorthy, “Diabetic retinopathy detection using deep learning techniques,” *J. Comput. Sci.*, vol. 16, no. 3, pp. 305–313, 2020, doi: 10.3844/JCSSP.2020.305.313.
- [32] F. Shenavarmasouleh, F. G. Mohammadi, M. H. Amini, and H. R. Arabnia, “DRDr II: Detecting the Severity Level of Diabetic Retinopathy Using Mask RCNN and Transfer Learning,” pp. 788–792, 2020, doi: 10.1109/csci51800.2020.00148.
- [33] A. Grzybowski *et al.*, “Artificial intelligence for diabetic retinopathy screening: a review,” *Eye*, vol. 34, no. 3, pp. 451–460, 2020, doi: 10.1038/s41433-019-0566-0.
- [34] H. E. Nkumbe, K. H. M. Kollmann, and H. C. Gaeckle, “Assessment of diabetic retinopathy in newly diagnosed black Kenyan type 2 diabetics,” *East Afr. Med. J.*, vol. 87, no. 3, pp. 109–114, 2010, doi: 10.4314/eamj.v87i3.62196.

- [35] H. Pratt, F. Coenen, D. M. Broadbent, S. P. Harding, and Y. Zheng, “Convolutional Neural Networks for Diabetic Retinopathy,” *Procedia Computer Science*, vol. 90. pp. 200–205, 2016, doi: 10.1016/j.procs.2016.07.014.
- [36] G. Jinfeng, S. Qummar, Z. Junming, Y. Ruxian, and F. G. Khan, “Ensemble framework of deep CNNs for diabetic retinopathy detection,” *Comput. Intell. Neurosci.*, vol. 2020, 2020, doi: 10.1155/2020/8864698.
- [37] M. Zaharia *et al.*, “Accelerating the Machine Learning Lifecycle with MLflow,” *Bull. IEEE Comput. Soc. Tech. Comm. Data Eng.*, pp. 39–45, 2018.
- [38] P. E. Miller *et al.*, “Predictive Abilities of Machine Learning Techniques May Be Limited by Dataset Characteristics: Insights From the UNOS Database,” *J. Card. Fail.*, vol. 25, no. 6, pp. 479–483, Jun. 2019, doi: 10.1016/j.cardfail.2019.01.018.
- [39] Y. Roh, G. Heo, and S. E. Whang, “A Survey on Data Collection for Machine Learning: A Big Data-AI Integration Perspective,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 33, no. 4. pp. 1328–1347, 2021, doi: 10.1109/TKDE.2019.2946162.
- [40] C. Vladimiro González Zelaya, “Towards Explaining the Effects of Data Preprocessing on Machine Learning; Towards Explaining the Effects of Data Preprocessing on Machine Learning,” *2019 IEEE 35th Int. Conf. Data Eng.*, 2019, doi: 10.1109/ICDE.2019.00245.
- [41] L. Moreira, C. Dantas, L. Oliveira, J. Soares, and E. Ogasawara, “On Evaluating Data Preprocessing Methods for Machine Learning Models for Flight Delays,” in *Proceedings of the International Joint Conference on*

*Neural Networks*, 2018, vol. 2018-July, doi: 10.1109/IJCNN.2018.8489294.

- [42] H. Alhichri, A. S. Alswayed, Y. Bazi, N. Ammour, and N. A. Alajlan, “Classification of Remote Sensing Images Using EfficientNet-B3 CNN Model with Attention,” *IEEE Access*, vol. 9, pp. 14078–14094, 2021, doi: 10.1109/ACCESS.2021.3051085.
- [43] M. Nixon, *Feature extraction & image processing*. 2008.
- [44] R. Sharma and E. N. Singh, “Comparative Study of Different Low Level Feature Extraction Techniques,” *Int. J. Eng. Res. Technol.*, vol. 3, no. 4, pp. 1454–1460, 2014.
- [45] W. Liu, S. Liao, W. Ren, W. Hu, and Y. Yu, “High-level Semantic Feature Detection: A New Perspective for Pedestrian Detection.”
- [46] Y. Liu, H. Pu, and D. W. Sun, “Efficient extraction of deep image features using convolutional neural network (CNN) for applications in detecting and analysing complex food matrices,” *Trends Food Sci. Technol.*, vol. 113, pp. 193–204, Jul. 2021, doi: 10.1016/J.TIFS.2021.04.042.
- [47] J. Cervantes, F. Garcia-Lamont, L. Rodríguez-Mazahua, and A. Lopez, “A comprehensive survey on support vector machine classification: Applications, challenges and trends,” *Neurocomputing*, vol. 408, pp. 189–215, Sep. 2020, doi: 10.1016/j.neucom.2019.10.118.
- [48] A. Zendeboudi, M. A. Baseer, and R. Saidur, “Application of support vector machine models for forecasting solar and wind energy resources: A review,” *Journal of Cleaner Production*, vol. 199, Elsevier, pp. 272–285, Oct. 20, 2018, doi: 10.1016/j.jclepro.2018.07.164.

- [49] H. Saadatfar, S. Khosravi, J. H. Joloudari, A. Mosavi, and S. Shamshirband, “A new k-nearest neighbors classifier for big data based on efficient data pruning,” *Mathematics*, vol. 8, no. 2, 2020, doi: 10.3390/math8020286.
- [50] B. Charbuty and A. Abdulazeez, “Classification Based on Decision Tree Algorithm for Machine Learning,” *J. Appl. Sci. Technol. Trends*, vol. 2, no. 01, pp. 20–28, 2021, doi: 10.38094/jastt20165.
- [51] J. Liang, Z. Qin, S. Xiao, L. Ou, and X. Lin, “Efficient and Secure Decision Tree Classification for Cloud-Assisted Online Diagnosis Services,” *IEEE Trans. Dependable Secur. Comput.*, vol. 18, no. 4, pp. 1632–1644, 2021, doi: 10.1109/TDSC.2019.2922958.
- [52] S. Xuan, G. Liu, Z. Li, L. Zheng, S. Wang, and C. Jiang, “Random forest for credit card fraud detection,” in *ICNSC 2018 - 15th IEEE International Conference on Networking, Sensing and Control*, 2018, pp. 1–6, doi: 10.1109/ICNSC.2018.8361343.
- [53] L. Cheng, X. Chen, J. De Vos, X. Lai, and F. Witlox, “Applying a random forest method approach to model travel mode choice behavior,” *Travel Behav. Soc.*, vol. 14, pp. 1–10, Jan. 2019, doi: 10.1016/J.TBS.2018.09.002.
- [54] IBM Cloud Education, “What are Neural Networks? | IBM,” *Ibm*. 2020.
- [55] O. I. Abiodun, A. Jantan, A. E. Omolara, K. V. Dada, N. A. E. Mohamed, and H. Arshad, “State-of-the-art in artificial neural network applications: A survey,” *Heliyon*, vol. 4, no. 11, p. e00938, Nov. 2018, doi: 10.1016/J.HELIYON.2018.E00938.
- [56] A. H. Elsheikh, S. W. Sharshir, M. Abd Elaziz, A. E. Kabeel, W. Guilan, and

- Z. Haiou, "Modeling of solar energy systems using artificial neural network: A comprehensive review," *Solar Energy*, vol. 180. Pergamon, pp. 622–639, Mar. 01, 2019, doi: 10.1016/j.solener.2019.01.037.
- [57] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *3rd Int. Conf. Learn. Represent. ICLR 2015 - Conf. Track Proc.*, pp. 1–14, 2015.
- [58] P. P. Shinde, *A Review of Machine Learning and Deep Learning Applications; A Review of Machine Learning and Deep Learning Applications*. 2018.
- [59] J. Qin, W. Pan, X. Xiang, Y. Tan, and G. Hou, "A biological image classification method based on improved CNN," *Ecol. Inform.*, vol. 58, p. 101093, Jul. 2020, doi: 10.1016/j.ecoinf.2020.101093.
- [60] I. Z. Mukti and D. Biswas, "Transfer Learning Based Plant Diseases Detection Using ResNet50," *2019 4th Int. Conf. Electr. Inf. Commun. Technol. EICT 2019*, no. December, pp. 20–22, 2019, doi: 10.1109/EICT48899.2019.9068805.
- [61] P. D. I. Torino and D. D. I. A. E, "Meta-Learning for Cross-Domain One-Shot Object Detection," 2021.
- [62] A. Rajeswaran, S. M. Kakade, C. Finn, and S. Levine, "Meta-learning with implicit gradients," *Adv. Neural Inf. Process. Syst.*, vol. 32, no. NeurIPS, pp. 1–12, 2019.
- [63] C. Finn and S. Levine, "Meta-Learning: from Few-Shot Learning to Rapid Reinforcement Learning," *ICML-Tutorial*, 2019.

- [64] Y. Zhang and Q. Yang, "A Survey on Multi-Task Learning," *IEEE Trans. Knowl. Data Eng.*, vol. 4347, no. c, pp. 1–20, 2021, doi: 10.1109/TKDE.2021.3070203.
- [65] A. A. Liu, Y. T. Su, W. Z. Nie, and M. Kankanhalli, "Hierarchical Clustering Multi-Task Learning for Joint Human Action Grouping and Recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 1, pp. 102–114, 2017, doi: 10.1109/TPAMI.2016.2537337.
- [66] S. Ruder, "An Overview of Multi-Task Learning in Deep Neural Networks," no. May, 2017.
- [67] L. Zhang, Q. Yang, X. Liu, and H. Guan, "Rethinking Hard-Parameter Sharing in Multi-Domain Learning," 2021.
- [68] I. Nusrat and S. B. Jang, "A comparison of regularization techniques in deep neural networks," *Symmetry (Basel)*, vol. 10, no. 11, 2018, doi: 10.3390/sym10110648.
- [69] Y. Pathak, P. K. Shukla, A. Tiwari, S. Stalin, and S. Singh, "Deep Transfer Learning Based Classification Model for COVID-19 Disease," *IRBM*, vol. 43, no. 2, pp. 87–92, Apr. 2022, doi: 10.1016/J.IRBM.2020.05.003.
- [70] T. Kaur and T. K. Gandhi, "Automated brain image classification based on VGG-16 and transfer learning," in *Proceedings - 2019 International Conference on Information Technology, ICIT 2019*, 2019, pp. 94–98, doi: 10.1109/ICIT48102.2019.00023.
- [71] A. Nandy, "A densenet based robust face detection framework," *Proc. - 2019 Int. Conf. Comput. Vis. Work. ICCVW 2019*, pp. 1840–1847, 2019, doi:

10.1109/ICCVW.2019.00229.

- [72] D. Impedovo, V. Dentamaro, G. Abbattista, V. Gattulli, and G. Pirlo, “A comparative study of shallow learning and deep transfer learning techniques for accurate fingerprints vitality detection,” *Pattern Recognit. Lett.*, vol. 151, pp. 11–18, 2021, doi: 10.1016/j.patrec.2021.07.025.
- [73] L. T. Duong, P. T. Nguyen, C. Di Sipio, and D. Di Ruscio, “Automated fruit recognition using EfficientNet and MixNet,” *Comput. Electron. Agric.*, vol. 171, no. January, p. 105326, 2020, doi: 10.1016/j.compag.2020.105326.
- [74] N. Hasan, Y. Bao, A. Shawon, and Y. Huang, “DenseNet Convolutional Neural Networks Application for Predicting COVID-19 Using CT Image,” *SN Comput. Sci.*, vol. 2, no. 5, pp. 1–11, 2021, doi: 10.1007/s42979-021-00782-7.
- [75] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, “Densely connected convolutional networks,” *Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017*, vol. 2017-Janua, pp. 2261–2269, 2017, doi: 10.1109/CVPR.2017.243.
- [76] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2016-Decem, pp. 770–778, 2016, doi: 10.1109/CVPR.2016.90.
- [77] M. Tan and Q. V. Le, “EfficientNet: Rethinking model scaling for convolutional neural networks,” *36th Int. Conf. Mach. Learn. ICML 2019*, vol. 2019-June, pp. 10691–10700, 2019.
- [78] D. R. Nayak, N. Padhy, P. K. Mallick, M. Zymbler, and S. Kumar, “Brain Tumor Classification Using Dense Efficient-Net,” *Axioms*, vol. 11, no. 1.



2022, doi: 10.3390/axioms11010034.

- [79] C. Szegedy *et al.*, “Going deeper with convolutions,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 07-12-June, pp. 1–9, 2015, doi: 10.1109/CVPR.2015.7298594.
- [80] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “Rethinking the Inception Architecture for Computer Vision,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2016-Decem, pp. 2818–2826, 2016, doi: 10.1109/CVPR.2016.308.
- [81] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, “Inception-v4, inception-ResNet and the impact of residual connections on learning,” *31st AAAI Conf. Artif. Intell. AAAI 2017*, pp. 4278–4284, 2017.
- [82] “VGG Very Deep Convolutional Networks (VGGNet) - What you need to know - viso.ai.” <https://viso.ai/deep-learning/vgg-very-deep-convolutional-networks/> (accessed Mar. 12, 2022).
- [83] Y. Li, J. Chen, and Y. Zheng, “A Multi-Task Self-Supervised Learning Framework for Scopy Images,” *Proc. - Int. Symp. Biomed. Imaging*, vol. 2020-April, pp. 2005–2009, 2020, doi: 10.1109/ISBI45749.2020.9098527.
- [84] H. Jia, B. DIng, H. Wang, X. Gong, and X. Zhou, “Fast adaptation via meta learning in multi-agent cooperative tasks,” *Proc. - 2019 IEEE SmartWorld, Ubiquitous Intell. Comput. Adv. Trust. Comput. Scalable Comput. Commun. Internet People Smart City Innov. SmartWorld/UIC/ATC/SCALCOM/IOP/SCI 2019*, pp. 707–714, 2019, doi: 10.1109/SmartWorld-UIC-ATC-SCALCOM-IOP-SCI.2019.00156.

- [85] M. Wistuba, N. Schilling, and L. Schmidt-Thieme, “Hyperparameter optimization machines,” *Proc. - 3rd IEEE Int. Conf. Data Sci. Adv. Anal. DSAA 2016*, pp. 41–50, 2016, doi: 10.1109/DSAA.2016.12.
- [86] K. H. N. Bui and H. Yi, “Optimal hyperparameter tuning using meta-learning for big traffic datasets,” *Proc. - 2020 IEEE Int. Conf. Big Data Smart Comput. BigComp 2020*, pp. 48–54, 2020, doi: 10.1109/BigComp48618.2020.0-100.
- [87] M. Z. A. Pon and K. P. KK, “Hyperparameter Tuning of Deep learning Models in Keras,” *Sparklinglight Trans. Artif. Intell. Quantum Comput.*, vol. 1, no. 1, pp. 36–40, 2021.
- [88] S. Joshi, J. A. Owens, S. Shah, and T. Munasinghe, “Analysis of Preprocessing Techniques, Keras Tuner, and Transfer Learning on Cloud Street image data,” pp. 4165–4168, 2022, doi: 10.1109/bigdata52589.2021.9671878.
- [89] A. F. Rogachev and E. V. Melikhova, “Automation of the process of selecting hyperparameters for artificial neural networks for processing retrospective text information,” *IOP Conf. Ser. Earth Environ. Sci.*, vol. 577, no. 1, 2020, doi: 10.1088/1755-1315/577/1/012012.
- [90] M. H. Al Banna *et al.*, “Attention-Based Bi-Directional Long-Short Term Memory Network for Earthquake Prediction,” *IEEE Access*, vol. 9, pp. 56589–56603, 2021, doi: 10.1109/ACCESS.2021.3071400.
- [91] O. Sagi and L. Rokach, “Ensemble learning: A survey,” *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.*, vol. 8, no. 4, pp. 1–18, 2018, doi: 10.1002/widm.1249.

- [92] X. Deng, Q. Liu, Y. Deng, and S. Mahadevan, “An improved method to construct basic probability assignment based on the confusion matrix for classification problem,” *Inf. Sci. (Ny)*, 2016, doi: 10.1016/j.ins.2016.01.033.
- [93] M. Heydarian, T. E. Doyle, and R. Samavi, “MLCM: Multi-Label Confusion Matrix,” *IEEE Access*, vol. 10, pp. 19083–19095, 2022, doi: 10.1109/ACCESS.2022.3151048.
- [94] J. Li *et al.*, “Establishment of noninvasive diabetes risk prediction model based on tongue features and machine learning techniques,” *Int. J. Med. Inform.*, vol. 149, no. February, p. 104429, 2021, doi: 10.1016/j.ijmedinf.2021.104429.
- [95] Z. H. Hoo, J. Candlish, and D. Teare, “What is an ROC curve?,” *Emerg. Med. J.*, vol. 34, no. 6, pp. 357–359, 2017, doi: 10.1136/emered-2017-206735.
- [96] H. Uğuz *et al.*, “ce pte d M us pt,” *J. Phys. Energy*, vol. 2, no. 1, pp. 0–31, 2020.
- [97] D. Vaughan, “Multiclass averaging,” *cran*, 2021. <https://cran.r-project.org/web/packages/yardstick/vignettes/multiclass.html> (accessed May 01, 2022).
- [98] N. Narayan, “Research design Research design,” *Res. Soc. Sci. Interdiscip. Perspect.*, no. September, pp. 68–84, 2017.
- [99] S. Bell, “Experimental Design,” *Int. Encycl. Hum. Geogr.*, pp. 672–675, 2009, doi: 10.1016/B978-008044910-4.00431-4.
- [100] Ridwan, “The Effect of Cooperative Learning in Blended Learning Environment on Students’ Learning Achievement: A True-Experimental

- Study,” *Proc. 2nd Int. Conf. Innov. Educ. Pedagog. (ICIEP 2020)*, vol. 619, no. Iciep 2020, pp. 157–164, 2022, doi: 10.2991/assehr.k.211219.029.
- [101] Y. Xiao and M. Watson, “Guidance on Conducting a Systematic Literature Review,” *J. Plan. Educ. Res.*, vol. 39, no. 1, pp. 93–112, 2019, doi: 10.1177/0739456X17723971.
- [102] A. Mikołajczyk and M. Grochowski, “Data augmentation for improving deep learning in image classification problem,” in *2018 International Interdisciplinary PhD Workshop, IIPhDW 2018*, 2018, pp. 117–122, doi: 10.1109/IIPHDW.2018.8388338.
- [103] “EyePACS.” <https://www.eyepacs.com/> (accessed Apr. 20, 2022).
- [104] R. Vimal Kurup, V. Sowmya, and K. P. Soman, “Effect of Data Pre-processing on Brain Tumor Classification Using Capsulenet,” *ICICCT 2019 – Syst. Reliab. Qual. Control. Safety, Maint. Manag.*, pp. 110–119, 2020, doi: 10.1007/978-981-13-8461-5\_13.
- [105] D. Zoran, M. Chrzanowski, P. Sen Huang, S. Goyal, A. Mott, and P. Kohli, “Towards robust image classification using sequential attention models,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 9480–9489, 2020, doi: 10.1109/CVPR42600.2020.00950.
- [106] B. Jang, M. Kim, G. Harerimana, S. U. Kang, and J. W. Kim, “Bi-LSTM model to increase accuracy in text classification: Combining word2vec CNN and attention mechanism,” *Appl. Sci.*, vol. 10, no. 17, 2020, doi: 10.3390/app10175841.
- [107] R. Li, S. Zheng, C. Duan, Y. Yang, and X. Wang, “Classification of

- hyperspectral image based on double-branch dual-attention mechanism network,” *Remote Sens.*, vol. 12, no. 3, 2020, doi: 10.3390/rs12030582.
- [108] M. Lin, Q. Chen, and S. Yan, “Network in network,” *2nd Int. Conf. Learn. Represent. ICLR 2014 - Conf. Track Proc.*, pp. 1–10, 2014.
- [109] T. Y. Hsiao, Y. C. Chang, H. H. Chou, and C. Te Chiu, “Filter-based deep-compression with global average pooling for convolutional networks,” *J. Syst. Archit.*, vol. 95, no. February, pp. 9–18, 2019, doi: 10.1016/j.sysarc.2019.02.008.
- [110] S. Santurkar, D. Tsipras, A. Ilyas, and A. Madry, “How does batch normalization help optimization?,” *Adv. Neural Inf. Process. Syst.*, vol. 2018-Decem, no. NeurIPS, pp. 2483–2493, 2018.
- [111] B. Gao and L. Pavel, “On the Properties of the Softmax Function with Application in Game Theory and Reinforcement Learning,” 2017.
- [112] I. Kandel and M. Castelli, “The effect of batch size on the generalizability of the convolutional neural networks on a histopathology dataset,” *ICT Express*, vol. 6, no. 4, pp. 312–315, 2020, doi: 10.1016/j.ict.2020.04.010.
- [113] A. Devarakonda, M. Naumov, and M. Garland, “AdaBatch: Adaptive Batch Sizes for Training Deep Neural Networks,” 2017.
- [114] J. D. Bodapati *et al.*, “Blended multi-modal deep convnet features for diabetic retinopathy severity prediction,” *Electron.*, vol. 9, no. 6, pp. 1–16, 2020, doi: 10.3390/electronics9060914.
- [115] S. H. Kassani, P. H. Kassani, R. Khazaeinezhad, M. J. Wesolowski, K. A. Schneider, and R. Deters, “Diabetic Retinopathy Classification Using a

Modified Xception Architecture,” 2019, doi:

10.1109/ISSPIT47144.2019.9001846.

- [116] V. Dondeti, J. D. Bodapati, S. N. Shareef, and V. Naralasetti, “Deep convolution features in non-linear embedding space for fundus image classification,” *Rev. d’Intelligence Artif.*, vol. 34, no. 3, pp. 307–313, 2020, doi: 10.18280/ria.340308.
- [117] N. S. S. & V. N. Jyostna Devi Bodapati, “Composite deep neural network with gated-attention mechanism for diabetic retinopathy severity classification.” *Journal of Ambient Intelligence and Humanized Computing* volume, 2021.
- [118] J. Vanschoren, “Meta-Learning : A Survey,” *arxiv.org Accept. Pre-print*, pp. 1–29, 2018.
- [119] T. M. Hospedales, A. Antoniou, P. Micaelli, and A. J. Storkey, “Meta-Learning in Neural Networks: A Survey,” *IEEE Trans. Pattern Anal. Mach. Intell.*, 2021, doi: 10.1109/TPAMI.2021.3079209.

## APPENDICES

## APPENDIX 1:

### SAMPLE CODE FOR ATTENTION MODULE

```
query_input = tf.keras.Input(shape=(None,), dtype='int32')
value_input = tf.keras.Input(shape=(None,), dtype='int32')
token_embedding = tf.keras.layers.Embedding(input_dim=512, output_dim=64)
query_embeddings = token_embedding(query_input)
value_embeddings = token_embedding(value_input)
cnn_layer = tf.keras.layers.Conv1D(
    filters=256,
    kernel_size=3,
    padding='same')
query_seq_encoding = cnn_layer(query_embeddings)
value_seq_encoding = cnn_layer(value_embeddings)
query_value_attention_seq = tf.keras.layers.Attention()(
    [query_seq_encoding, value_seq_encoding])
query_encoding = tf.keras.layers.GlobalAveragePooling1D()(
    query_seq_encoding)
query_value_attention = tf.keras.layers.GlobalAveragePooling1D()(
    query_value_attention_seq)
input_layer = tf.keras.layers.Concatenate()(
    [query_encoding, query_value_attention])
```



## APPENDIX 2:

### SAMPLE CODE FOR REBUILDING THE TOP LAYER AND TRAINING

#### THE TOP LAYER ONLY

```
def build_model(num_classes):
    inputs = layers.Input(shape=(512,512, 3))
    x = image(inputs)
    model = VGG16(include_top=False, input_tensor=x, weights="imagenet")

    # Freeze the pretrained weights for teh convolution block
    model.trainable = False
    x= input_layer
    # Rebuild top
    x = layers.GlobalAveragePooling2D(name="avg_pool")(model.output)
    x = layers.BatchNormalization()(x)

    x = layers.Dropout(0.2, name="top_dropout")(x)
    outputs = layers.Dense(NUM_CLASSES, activation="softmax", name="pred")(x)
```

```
# compile
model = tf.keras.Model(inputs, outputs, name="VGG16")
optimizer = tf.keras.optimizers.Adam(learning_rate=1e-2, clipnorm=1.0)
model.compile(
    optimizer=optimizer, loss="categorical_crossentropy", metrics=["accuracy", tf.keras.metrics.Recall(name='recall'),
    tf.keras.metrics.Precision(name='precision'), tfa.metrics.F1Score(name= 'f1_score', num_classes=5),
    tfa.metrics.CohenKappa(num_classes=5, name = 'cohen_kappa', weightage = "quadratic", sparse_labels = False, regressio
)
return model
```

## APPENDIX 3:

### SAMPLE CODE FOR FULL MODEL COMPILATION


```
Modelv1.trainable = True
Modelv1.summary()

Modelv1.compile(
    optimizer=tf.keras.optimizers.Adam(learning_rate=lr, clipnorm=1.0), # Low learning rate
    loss="categorical_crossentropy",
    metrics=["accuracy", tf.keras.metrics.Recall(name='recall'),tf.keras.metrics.Precision(name='precis
    tfa.metrics.F1Score(name= 'f1_score', num_classes=5),
    tfa.metrics.CohenKappa(num_classes=5, name = 'cohen_kappa', weightage = "quadratic",sparse_labels
    tf.keras.metrics.AUC(num_thresholds=200, curve='ROC',summation_method='interpolation', name=None,
    multi_label=True,
    num_labels=5,
    label_weights=None,
    from_logits=False
)
    ],
)

epochs = 20
hist = Modelv1.fit(ds_train, epochs=epochs, validation_data=validation , callbacks=[model_cp, Auto_lr]
```

## APPENDIX4:

### RESEARCH AUTHORIZATION (MUT)

  
**MURANG'A UNIVERSITY OF TECHNOLOGY**  
**DIRECTORATE OF POSTGRADUATE STUDIES**

P.O. BOX 75 - 10200, MURANG'A Email: [bps@mut.ac.ke](mailto:bps@mut.ac.ke)

---

**Ref: MUT/ARP/PGS/20/2020/VOL.I** **Date: 3<sup>rd</sup> February 2022**

Dear Jackson Kamiri Wambugu (SC401/5412/2019),

**RE: APPROVAL OF RESEARCH PROPOSAL AND SUPERVISORS**

I am pleased to inform you that the Directorate of Postgraduate Studies on 17<sup>th</sup> January 2022 considered and approved your Masters research proposal entitled "*A Meta-Learning Model for Early Detection of Diabetic Retinopathy*" and appointed the following as supervisors:

1. **Dr. Geoffrey Mariga - Murang'a University of Technology**
2. **Dr. Aaron Oirere - Murang'a University of Technology**

You may now proceed with your data collection subject to obtaining research permit from NACOSTI, if required. You should also begin consulting your supervisors and submit through them quarterly progress reports to the Director Postgraduate Studies through your CoD and School Dean. Progress Reports can be accessed in the University Website.

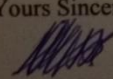
It is the policy and regulations of the University that you observe deadlines. The guidelines on Postgraduate supervision can be accessed in the post graduate Handbook.

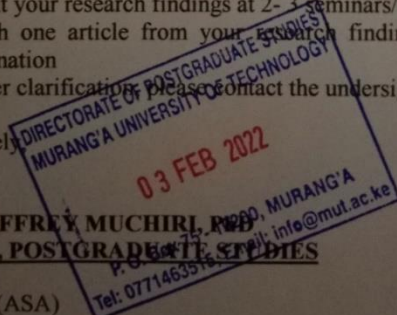
Your responsibilities as a student will include, among others;

- i. Maintain regular consultation with your supervisor(s), at least once a month
- ii. Submit quarterly reports on time, through your supervisors, CoD, Dean and to the Director of Postgraduate Studies;
- iii. Ensure quality work all through;
- iv. Present your research findings at 2- 3 seminars/conferences prior to thesis examination.
- v. Publish one article from your research findings in a refereed journal prior to thesis examination


For any further clarification, please contact the undersigned.

Yours Sincerely,

  
**PROF. GEOFFREY MUCHIRI, PhD**  
**DIRECTOR, POSTGRADUATE STUDIES**  
P.O. BOX 75, 10200, MURANG'A  
Tel: 0771463518, Email: [info@mut.ac.ke](mailto:info@mut.ac.ke)


  
DIRECTORATE OF POSTGRADUATE STUDIES  
MURANG'A UNIVERSITY OF TECHNOLOGY  
03 FEB 2022


Cc Registrar (ASA)  
Dean (SCIT)

 **MUT IS ISO 9001:2015 CERTIFIED**

**APPENDIX5:**


**RESEARCH AUTHORIZATION (NACOSTI)**

  
**REPUBLIC OF KENYA**

  
**NATIONAL COMMISSION FOR  
SCIENCE, TECHNOLOGY & INNOVATION**

Ref No: **655257** Date of Issue: **28/March/2022**


**RESEARCH LICENSE**




**This is to Certify that Mr. Jackson Kamiri Wambugu of Murang'a University of Technology, has been licensed to conduct research in Muranga on the topic: A Meta-Learning Model for Early Detection of Diabetic Retinopathy for the period ending : 28/March/2023.**

License No: **NACOSTI/P/22/16528**

**655257**  
Applicant Identification Number

  
Director General  
**NATIONAL COMMISSION FOR  
SCIENCE, TECHNOLOGY &  
INNOVATION**

Verification QR Code



**NOTE: This is a computer generated License. To verify the authenticity of this document,  
Scan the QR Code using QR scanner application.**

**APPENDIX 6:**

**LINK TO THE DATASET**

[Diabetic Retinopathy Detection | Kaggle](#)

## APPENDIX 7:

### PUBLICATIONS

The following papers have been published from this thesis results:

- [1] J. Kamiri, G. Mariga, and A. Oirere, “A Systematic Literature Review of Meta-Learning Models for Classification Tasks,” *Int. J. Comput. Appl. Technol. Res.*, vol. 11, no. 03, pp. 56–65, 2022, doi: 10.7753/ijcatr1103.1002.

(<https://ijcat.com/archieve/volume11/issue3/ijcatr11031002.pdf> )

- [2] J. Kamiri, G. M. Wambugu, and A. M. Oirere, “A Comparative Study of Deep Learning and Transfer Learning in Detection of Diabetic Retinopathy,” *Int. J. Comput. Appl. Technol. Res.*, vol. 11, no. 07, pp. 247–254, 2022, doi: 10.7753/ijcatr1107.1001.

(<https://ijcat.com/archieve/volume11/volume11issue7.pdf> )