A Deep Learning Approach for the Detection and Grading of Neovascularization in Fundus Images Using Transfer Learning

A project report submitted in partial fulfilment of the requirement for the Award of the Degree of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

by

Sirnam Karthik Kumar (160719733016)

Amima Shifa (160719733012)

Bobbala Rohit Reddy (160719733022)

Under the Guidance of

Dr. P. Lavanya,

Professor &

Head of the Department

Department of Computer Science and Engineering

Methodist College of Engineering and Technology, King Koti, Abids, Hyderabad-500001.

2022-2023

A Deep Learning Approach for the Detection and Grading of Neovascularization in Fundus Images Using Transfer Learning

A project report submitted in partial fulfilment of the requirement for the Award of the Degree of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

by

Sirnam Karthik Kumar (160719733016)

Amima Shifa (160719733012)

Bobbala Rohit Reddy (160719733022)

Under the Guidance of

Dr. P. Lavanya,
Professor &,
Head of the Department

Department of Computer Science and Engineering

Methodist College of Engineering and Technology,
King Koti,Abids, Hyderabad-500001.

2022-2023

Methodist College of Engineering and Technology,
KingKoti, Abids, Hyderabad-500001,
Department of Computer Science and Engineering

DECLARATION BY THE CANDIDATES

We, **Sirnam Karthik Kumar (160719733016), Amima Shifa(160719733012)** and **Bobbala Rohit Reddy (160719733022)** students of Methodist College of Engineering and Technology, pursuing Bachelor's degree in Computer Science and Engineering, hereby declare that this Project report entitled "**A Deep Learning Approach for the Detection and Grading of Neovascularization in Fundus Images using Transfer Learning",** carried out under the guidance of **Dr.P.Lavanya** submitted in partial fulfilment of the requirements for the degree of Bachelor of Engineering in Computer Science. This is a record work carried out by us and the results embodied in this report have not been reproduced/copied from any source.

Sirnam Karthik Kumar (160719733016)

Amima Shifa (160719733012)

Bobbala Rohit Reddy (160719733022)

Methodist College of Engineering and Technology,
KingKoti, Abids, Hyderabad-500001,
Department of Computer Science and Engineering

CERTIFICATE BY THE SUPERVISOR

This is to certify that this Project report entitled "**A Deep Learning Approach for the Detection and Grading of Neovascularization in Fundus Images using Transfer Learning**" being submitted by Sirnam Karthik Kumar (160719733016), Amima Shifa (160719733012) and Bobbala Rohit Reddy (160719733022) submitted in partial fulfillment of the requirements for the degree of Bachelor of Engineering in Computer Science and Engineering, during the academic year 2022-2023, is a bonfide record of work carried out by them.

Dr. P. Lavanya
Professor &
Head of the Department

Date:

Methodist College of Engineering and Technology,
KingKoti, Abids, Hyderabad-500001,
Department of Computer Science and Engineering

CERTIFICATE BY HEAD OF THE DEPARTMENT

This is to certify that this Project report entitled "**A Deep Learning Approach for the Detection and Grading of Neovascularization in Fundus Images using Transfer Learning**" by Sirnam Karthik Kumar (160719733016), Amima Shifa(160719733012) and Bobbala Rohit Reddy (160719733022) submitted in partial fulfilment of the requirements for the degree of Bachelor of Engineering in Computer Science and Engineering of the Osmania University, Hyderabad, during the academic year 2022-2023, is a bonafide record of work carried out by them.

Dr. P. Lavanya
Professor &
Head of the Department

Date:

Methodist College of Engineering and Technology,
KingKoti, Abids, Hyderabad-500001,
Department of Computer Science and Engineering

PROJECT APPROVAL CERTIFICATE

This is to certify that this Project report entitled "**A Deep Learning Approach for the Detection and Grading of Neovascularization in Fundus Images using Transfer Learning**" by Sirnam Karthik Kumar (160719733016), Amima Shifa(160719733012) and Bobbala Rohit Reddy (160719733022) submitted in partial fulfilment of the requirements for the degree of Bachelor of Engineering in Computer Science and Engineering of the Osmania University, Hyderabad, during the academic year 2022-2023, is a bonafide record of work carried out by them.

INTERNAL **EXTERNAL** **HOD**

ACKNOWLEDGEMNTS

We would like to express our sincere gratitude our Project guide **Dr. P. Lavanya, Professor & Head of the Department,** for giving us the opportunity to work on this topic. It would never be possible for us to take this project to this level without her innovative ideas and his relentless support, encouragement and for her valuable guidance and encouragement which has played a major role in the completion of the project and for helping us by being an example of high vision and pushing towards greater limits of achievement.

We would like to thank our project coordinator **Mr.T.Praveen Kumar, Assistant Professor, CSE,** who helped us by being an example of high vision and pushing towards greater limits of achievement.

We would like to express a deep sense of gratitude towards the **Dr. Prabhu G. Benakop, Principal, Methodist College of Engineering and Technology,** for always being an inspiration and for always encouraging us in every possible way.

We would like to express a deep sense of gratitude towards the **Dr. Lakshmipathi Rao, Director, Methodist College of Engineering and Technology,** for always being an inspiration and for always encouraging us in every possible way.

We are indebted to the Department of Computer Science & Engineering and Methodist College of Engineering and Technology for providing us with all the required facility to carry our work in a congenial environment. We extend our gratitude to the CSE Department staff for providing us to the needful time to time whenever requested.

We would like to thank our parents for allowing us to realize our potential, all the support they have provided us over the years was the greatest gift anyone has ever given us and also for teaching us the value of hard work and education. Our parents have offered us with tremendous support and encouragement, thanks to our parents for all the moral support and the amazing opportunities they have given us over the years.

METHODIST COLLEGE OF ENGINEERING & TECHNOLOGY

Approved by AICTE, Affiliated to Osmania University, - College Code – 1607

Department of Computer Science & Engineering

Vision & Mission

VISION

To become a leader in providing Computer Science & Engineering education with emphasis on knowledge and innovation.

MISSION

M1: To offer flexible programs of study with collaborations to suit industry needs
M2: To provide quality education and training through novel pedagogical practices
M3: To Expedite high performance of excellence in teaching, research and innovations.
M4: To impart moral, ethical valued education with social responsibility.

Program Educational Objectives

Graduates of Compute Science and Engineering at Methodist College of Engineering and Technology will be able to:

PEO1: Apply technical concepts, Analyze, Synthesize data to Design and create novel products and solutions for the real life problems.

PEO2: Apply the knowledge of Computer Science Engineering to pursue higher education with due consideration to environment and society.

PEO3: Promote collaborative learning and spirit of team work through multidisciplinary projects

PEO4: Engage in life-long learning and develop entrepreneurial skills.

Program Specific Outcomes

At the end of 4 years, Compute Science and Engineering graduates at MCET will be able to:

PSO1: Apply the knowledge of Computer Science and Engineering in various domains like networking and data mining to manage projects in multidisciplinary environments.

PSO2: Develop software applications with open-ended programming environments**.**

PSO3: Design and develop solutions by following standard software engineering principles and implement by using suitable programminglanguages and platforms

PROGRAM OUTCOMES

PO1: Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution ofcomplex engineering problems.

PO2: Problem analysis: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

PO3: Design/development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

PO4: Conduct investigations of complex problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation ofdata, and synthesis of the information to provide valid conclusions.

PO5: Modern tool usage: Create, select, and apply appropriate techniques, resources,and modern engineering and IT tools including prediction and modelling to complexengineering activities with an understanding of the limitations.

PO6: The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

PO7: Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

PO8: Ethics: Apply ethical principles and commit to professional ethics andresponsibilities and norms of the engineering practice.

PO9: Individual and team work: Function effectively as an individual, and as a memberor leader in diverse teams, and in multidisciplinary settings.

PO10: Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

PO11: Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

PO12: Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

ABSTRACT

Neovascularization, or the growth of new blood vessels, is a common complication of several ocular diseases, including age-related macular degeneration and diabetic retinopathy. Early detection and monitoring of neovascularization are crucial for the successful management of these conditions. In this project, we propose a method for detecting and grading neovascularization in fundus images using transfer learning. Transfer learning is the process of exploiting the knowledge gained from a previous task to improve generalization on another.

The pre-trained networks are trained on the ImageNet dataset, and their acquired knowledge is applied to a large dataset of fundus images containing both normal and neovascularized regions. The pre-trained networks are used as feature extractors to output features and effectively classify neovascularization in the target dataset with high accuracy. We then update model architectures by replacing fully connected layers with new initialized ones and further training the new fully connected layers to predict the neovascularization classes. In addition, the performance of this method is evaluated and compared to several baseline approaches on a held-out test set.

The method outperforms the baselines, demonstrating its effectiveness at detecting neovascularization in fundus images, with highest accuracy of 94.31% using ConvNext Large network. Overall, our study demonstrates the effectiveness of transfer learning for analyzing and detecting neovascularization in fundus images and has the potential to be applied in clinical settings for the early detection and monitoring of ocular diseases. However, further work can be done to use this approach on other medical image classification tasks.

Keywords: Neovascularization, Transfer Learning, CNN

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVATIONS

IDF- International Diabetes Federation

DR- Diabetic Retinopathy

VTDR- Vision-Threatening Diabetic Retinopathy

NDPR- Non-Proliferative Diabetic Retinopathy

PDR- Proliferative Diabetic Retinopathy

DL- Deep learning

CNN- Convolutional Neural Networks

APTOS- Asia Pacific Tele-Ophthalmology Society

NLMD- Non-Local Means Denoising

ICDRS- International Centre for Dispute Resolution

DFD- Data Flow Diagram

ILSVRC- ImageNet Large Scale Visual Recognition Challenge

1.INTRODUCTION

1.INTRODUCTION

1.1 Background

Diabetes is a chronic illness that builds blood sugar (glucose) levels to dangerous levels. Diabetes results from inadequate insulin synthesis or improper insulin utilization by the body's cells. The number of people diagnosed with diabetes has increased over the last few years. More than 415 million people worldwide have diabetes, according to the 7th edition of the International Diabetes Federation (IDF) Atlas from 2015 [1]. This serious public health problem affects 463 million people worldwide, and this number is projected to rise to 700 million by 2045 [2]. At least one-third of people with diabetes also suffer from an eye disease related to diabetes, of which diabetic retinopathy (DR) is the most common [3]. Any diabetic patient, regardless of how severe their condition is, can develop DR, which is characterized by progressive vascular disruptions in the retina brought on by chronic hyperglycemia [4]. There are estimated to be 93 million persons with DR worldwide, making it the main cause of blindness among working-age adults. These figures are anticipated to increase even more, mostly due to the increased prevalence of diabetes in emerging Asian nations like China and India.The rapid progress of internet-based networking technologies enables a remote access to engineering laboratory equipment and instruments. Remote access power system network is gradually emerging with the advance of technology. This topic and related issues have to be handled in the engineering curricula as platforms for inter-university collaborations aimed at establishing global distance education. Basically, the remote web based real time laboratory is based on the concept of using the web to monitor and control of the power system network. This concept needs to be addressed in the literature within unified and consistent framework.

1.2 Impact of Diabetic Retinopathy

Diabetic retinopathy (DR), a major microvascular complication of diabetes, significantly impacts the world's health systems. Globally, the number of people with DR will grow from 126.6 million in 2010 to 191.0 million by 2030. We estimate that the number of vision-threatening diabetic retinopathy (VTDR) will increase from 37.3 million to 56.3 million if prompt action is not taken. Despite mounting data supporting the value of routine DR screening and early intervention, DR typically impairs visual function and is the primary cause of blindness in working-age populations. In many low-income countries, DR has been

neglected in healthcare research and planning, where access to trained eye-care professionals and tertiary eye-care services may be inadequate. There can be a problem with both the supply and demand for services. Damage to the retina's blood vessels results in DR. It can be divided into two subtypes: non-proliferative diabetic retinopathy (NPDR) and proliferative diabetic retinopathy (PDR) [5]. Microvascular leakage of the retinal blood vessels, which causes microaneurysms, exudates, and hemorrhages, is a distinctive feature of NPDR [6]. Neovascularization occurs as part of the development from NPDR to PDR. Figure 1 shows a comparison between a healthy retina and an unhealthy retina.

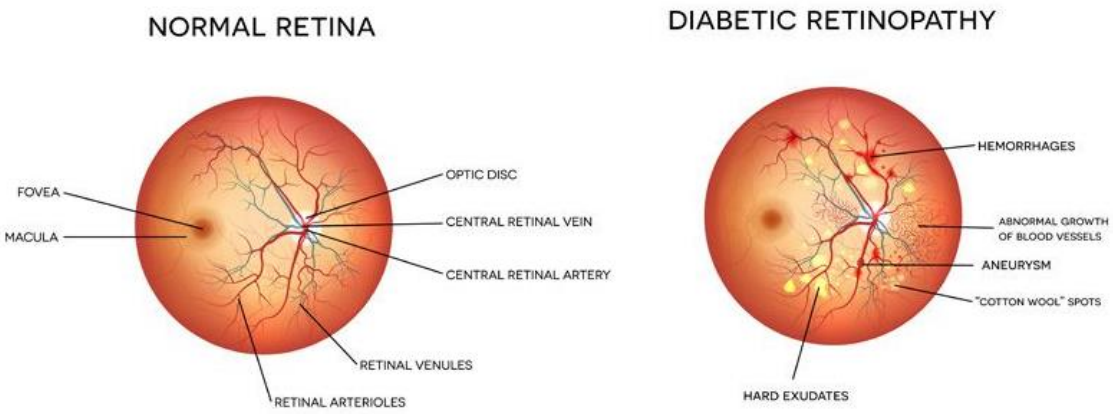

Figure 1.2: A healthy retina vs unhealthy retina (https://neoretina.com/blog/diabetic-retinopathy-can-it-be-reversed/)

NPDR and PDR both risk significant vision loss [6]. However, PDR is more severe because it can potentially develop microvascular occlusion of retinal vessels. In response, the retina develops new, and Vitreous bleeding can occur if these fragile new blood vessels rupture. This vitreous bleeding is dangerous, as the blood in the vitreous will organize and form fibrous tissue. The contraction of fibrous tissue will cause traction on the retinal layer and damage the retinal cells.

A serious visual impairment may result as a consequence. The retina is a special location for diagnosing microvascular diseases and fundus imaging. The development of computer-aided techniques for automatic retinal disease identification is now possible because of recent developments in retinal imaging. Due to its low cost and scalability, this strategy has recently attracted a lot of researchers to build retinal screening devices employing imaging techniques [7,8]. Due to PDR's small size and haphazard growth pattern, it is still challenging to detect neovascularization. As a result, it is challenging to develop an

automated diagnosis system for PDR detection due to the fact that automated illness diagnosis is ineffective in complex health conditions.

Recent methods for PDR detection frequently rely on studying retinal fundus photographs. It typically starts with picture enhancement and removal of the optic disc, then uses image processing or machine learning to extract the clinical aspects of the condition.

1.3 Deep Learning Methods

Deep learning (DL) is a class of artificial intelligence (AI) methods inspired by the structure of the human brain and based on artificial neural networks. Essentially, DL refers to methods for learning the mathematical representation of the latent and intrinsic relations of the data in an automatic manner. Contrasting to typical machine learning techniques, deep learning ones learn the proper features directly from the data instead of relying on the development of hand-crafted features, a procedure that may be very time-consuming and labor-intensive. Additionally, when the volume of data grows, DL methods scale significantly better than conventional ML methods. Some of the most important DL concepts are briefly summarized in this section.

1.3.1 Convolutional Neural Network (CNN) in Fundus Analysis

Convolutional Neural Networks (CNN) were inspired by human vision and are based on a fundamental mathematical operation called "convolution." Unlike shallow neural networks, CNNs may accept 2D arrays as input. The main difference between a CNN and a DNN is that for the latter, the output of every neuron at the subsequent layer is computed by all the neurons at that layer, which is not the case for a CNN. On the other hand, a CNN uses filters or kernels to compute convolutions by swiping over a portion of the original image to create a feature map. Thus, if the filter's size is x x, only a window of x2 pixels will be used to calculate each unit's value in the feature map of the subsequent layer. This affects the receptive field, which is the area in the input space that a certain CNN feature is impacted by. Finally, the convolutional part is often referred to as "the feature extraction part" of the network, while the rest is called "the classification part". The latter, which is effectively a deep neural network, receives the one-dimensional array of imaging features from the former and uses it to classify the input image using the created features.

1.3.2 Transfer Learning

Training a deep neural network is very demanding in terms of the computational resources and data required. The world's largest object detection database, ImageNET [9], consists of over 14 million real-life images, such as animals, devices, food, people, vehicles, etc. Due to the crucial curation, annotation, and regulatory considerations, medical photographs are much more difficult to collect than images of commonplace objects. As a result, it might be challenging to train reliable and accurate models for medical issues. Even though the application field differs, it is still possible to use models that are trained on huge datasets, like ImageNet, by transferring the learned information to another model. Transfer learning does exactly that, i.e., it improves the learning in one task by transferring knowledge from another task that is already learned. The network can more easily recognize low-level elements of the image (such as edges, contours, etc.) by transferring knowledge from ImageNet to a medical aging domain. The model must be fine-tuned (i.e., retrained) on the new task (i.e., a new dataset) in order to actually detect DR. This process will, however, be significantly faster and more accurate than training the model from scratch.

Pre-Trained Model Approach:

Source Model: A pre-trained model is chosen from an available pool of candidate models.

Customize Model: Next, new fully linked layers can be constructed using the pre-trained model as a baseline. One can use all or only a portion of the model, depending on the needs and the methods employed.

Tuning Mode: To improve performance, the model is further altered for better performance.

Transfer learning benefits include:

1. **Higher start:** When compared to a model without the use of transfer learning, the skill level on the source model is higher at the beginning.
2. **Higher slope:** The skill develops more quickly during model training, resulting in superior performance.
3. **A higher asymptote**: The skill convergence is superior to the one that does not apply transfer learning.

1.3.3 Ensemble Learning

The use of several models (also known as base models) to provide predictions that are more accurate than those produced by individual models is known as ensemble learning, and it is a very important area of research in AI. The idea behind ensemble learning is that by combining the predictions of multiple models, the overall performance and accuracy of the ensemble can be improved compared to individual models.

Ensemble learning can be used with various types of machine learning algorithms, such as decision trees, neural networks, support vector machines, and more. There are several common approaches to ensemble learning, including:

- **Voting Ensemble:** In this approach, predictions from multiple models are combined by taking a majority vote. For example, in a binary classification problem, if the majority of the models predict class A, the ensemble predicts class A.

- **Bagging (Bootstrap Aggregating):** In this approach, multiple instances of the same model are trained on different subsets of the training data obtained by bootstrapping (random sampling with replacement). The predictions of these models are then combined, often by averaging or taking a majority vote, to obtain the ensemble prediction.

- **Boosting:** In this approach, multiple instances of the same model are trained sequentially, with each instance giving more importance to the misclassified instances from the previous instances. The idea is to correct the errors of the previous models and improve the overall performance of the ensemble.

- **Stacking:** In this approach, multiple models are trained to make predictions, and their predictions are used as input features for training another model, often called a meta-model or blender, to make the final prediction. This allows the ensemble to capture higher-level interactions among the base models and can lead to improved performance.

Ensemble learning can be a powerful technique to improve the accuracy and robustness of machine learning models, especially when dealing with complex tasks or noisy datasets.

1.4 Dataset Description

Kaggle's APTOS 2019 dataset was collected by Aravind Eye Hospital in India's rural areas to build powerful tools to automatically diagnose diabetic retinopathy and improve the hospital's ability to identify potential patients. This dataset contains 3662 fundus photographs that were taken under various imaging circumstances.

Each image has received a 0–4 severity rating from a medical professional for the presence of diabetic retinopathy.

0 - No DR

1 - Mild

2- Moderate

3- Severe

4 - Proliferative DR

Among four severity levels: Normal, Mild, Moderate, Severe, and Proliferative Diabetic Retinopathy. The most dangerous stage of Diabetic Retinopathy is the proliferative DR, in which the likelihood of blood leaking is at a peak, causing permanent vision loss. The current state of Diabetic Retinopathy screening in the real world is based on the assessment of color fundus photography (see Figure 2), which is induced by an Ophthalmologist.

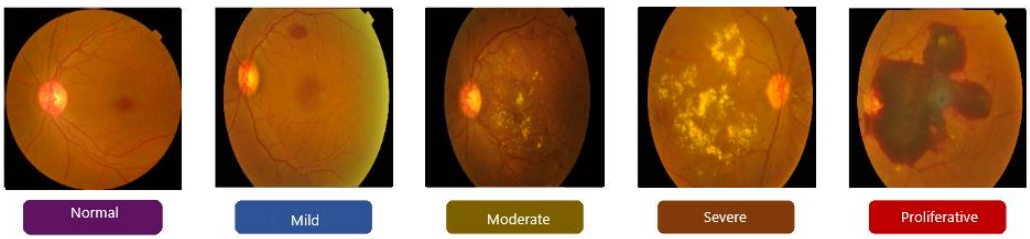

Figure 1.4.1: The fundus photography images from APTOS2019 Kaggle's dataset for each class.

Like any real-world data set, there is noise in both the images and labels. Although they are blurry,underexposed, or overexposed, images always contain artifacts (see Figure 3). Further variation will be introduced because the photos were collected over an extended

period of time from numerous clinics using different cameras. The significant class imbalance is one of its drawbacks, particularly for the severe NPDR class, which includes only 193 photos.

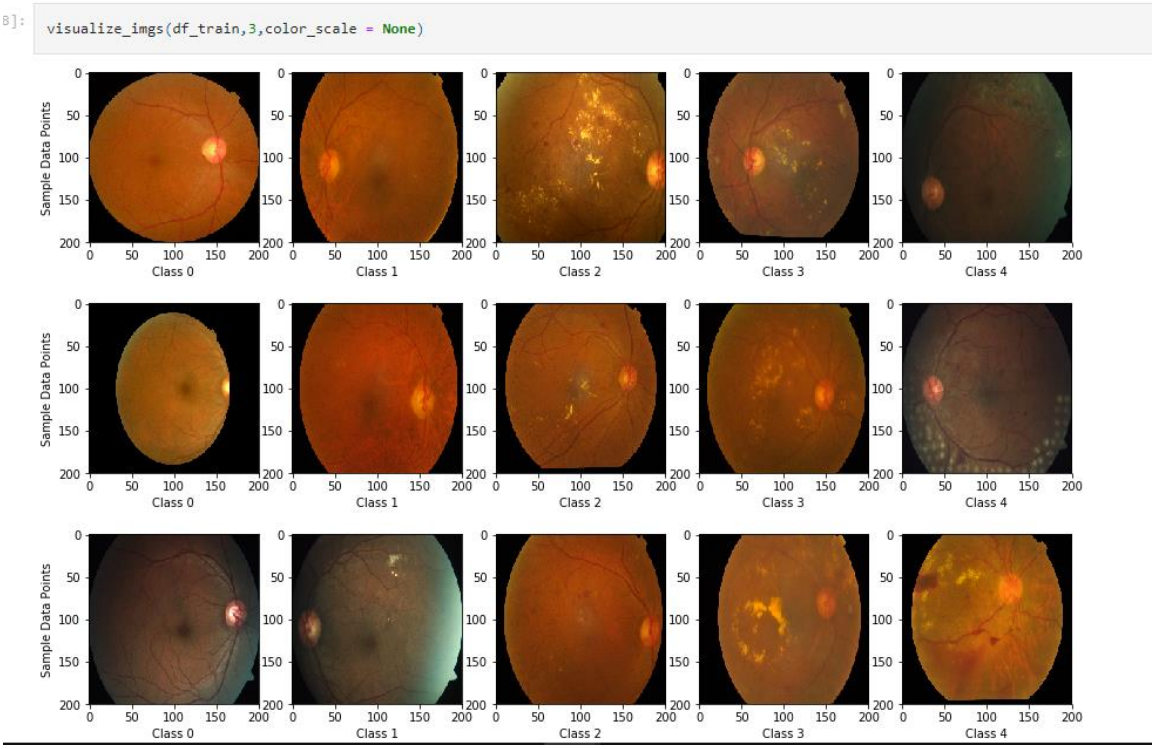

Figure 1.4.2: Artifacts and Noise in the dataset.

1.5 Dataset Pre-processing

As mentioned in Section 1.4, the noise gets incorporated into the final image while obtaining fundus photos with different hardware under varied conditions. Pre-processing the images is typically a necessary step in most of the research evaluated in order to reduce such heterogeneity, which eventually influences the performance of the classification model, as well as highlights some of the tiny elements of the images.

1.5.1 Denoise and Normalization

Non-Local Means Denoising (NLMD) is applied to remove potential noise in the image. It should be noted, though, that while a stronger denoising algorithm may remove more noise, it will also degrade the fine features of the image (i.e., make it blurry). Also, image intensity normalization is applied to avoid introducing bias and high training times to the network and to standardize the data to a particular scale (e.g., each image having a mean value of 0 and a standard deviation of 1, regarding its pixels' intensity).

1.5.2 Crop and Resize

The datasets may contain images that could vary in terms of resolution and aspect ratio. They also contain uninformative black-space areas in the image. To standardize the image size and remove such black space areas, the images may be cropped, rescaled, and resized to a specific resolution. First, the center coordinates and radius of the circular region are calculated, then binary masks of zeroes are created, which are later filled with 1s in the center region. Finally, the mask is applied to the original image using "Bitwise and Operator". It helps retain the pixel within the circular region of interest.

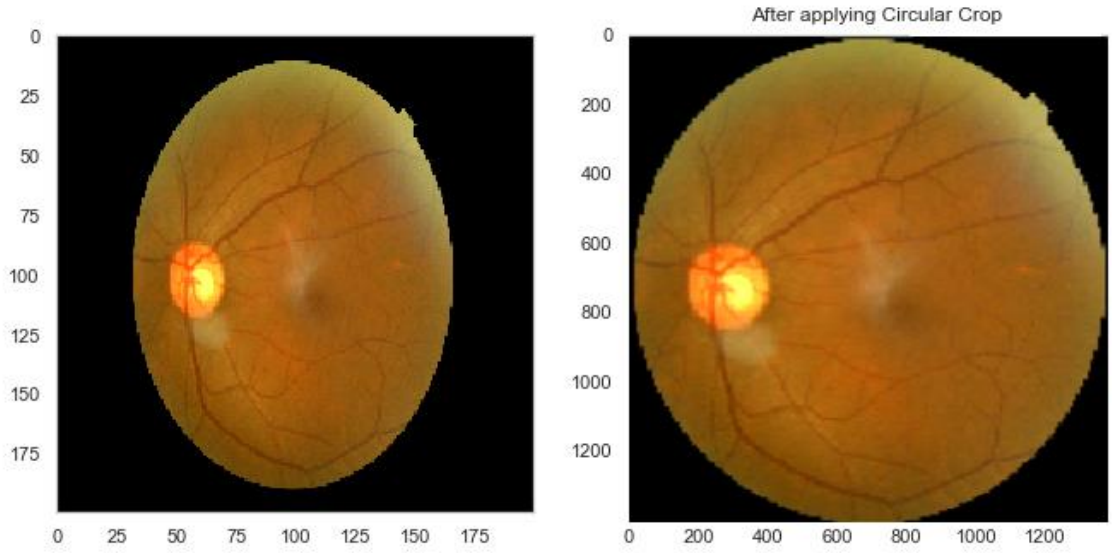

Figure 1.5.2: Crop and Resize on image.

1.5.3 Data Up sampling

As mentioned in Section 1.4, one of the limitations of the dataset is the large class imbalance, especially for the Severe NPDR class, which has only 193 images. So, in order to obtain a balanced dataset, picture sampling is done using the statistical distribution function to increase the number of images in each dataset. First, the number of pictures to be up sampled for each image was calculated using the formula.

$$Add = \frac{Nh - Ni}{Ni}$$

Nh - Number of images in highest class (training set)

Ni – Number of images in respective class (training set)

Second, a random upsampling was done to increase the number of images in the respective classes. The image transformation methods in upsampling include left-right symmetrical transformation, up-down symmetric transformation, and random-angle rotation transformation set to 45 degrees.

The table below lists the number of images for each class after picture sampling.

Severity Scale	Class	Number of Original images	Number of images added per picture	Number of images after upscaling
No_DR	0	1805	0	1805
Mild	1	370	3	1110
Moderate	2	999	0	999
Severe	3	193	8	1544
Proliferate	4	295	5	1475

Table 1.5.3: number of images for each class after picture sampling.

1.5.4 Grading of the Dataset

In Section 1.4, Kaggle presented the common 5-class grading scale that ophthalmologists utilize in order to grade a fundus image regarding the DR disease. In other instances, scholars have chosen to categorize them differently by finally merging numerous groupings. Researchers specifically tested the effects of DR grading on a 2-class (i.e., identifying the presence of the disease), 3-class (i.e., no DR, mild DR, and severe DR), and 4-class (i.e., no DR, mild DR, moderate DR, and severe DR) classification. They defined referable DR as DR when at least moderate NPDR lesions are observed and vision-threatening DR as DR when at least severe NPDR or PDR lesions are observed. The authors of Ref. [10] developed a 4-class scale for a variety of purposes, including encoding analogous

clinical manifestations between the several stages of the conventional scaling protocol (ICDRS). Islam et. al. [11] created two binary classification models: one for evaluating the disease's severity (grade 0,1 vs. 2,3,4) and another for identifying the disease's presence (healthy vs. diseased). Li et. al.'s [12] classification of ungradable photos included the usage of a further class (a total of six).

Classification Strategy	Classes
2-class	No Dr, Dr
3-class	no DR, mild DR and severe DR
4-class	no DR, mild DR, moderate DR and severe DR
5-class	no DR, mild DR, moderate DR, severe DR, Proliferate DR
6-class	no DR, mild DR, moderate DR, severe DR, Proliferate DR, Unclassified DR

Table 1.5.4: Different Classification Strategies.

2.LITERATURE REVIEW

2.LITERATURE REVIEW

A leading cause of blindness worldwide, diabetic retinopathy affects millions of people. Neovascularization is a common complication of diabetic retinopathy, which can lead to severe vision loss if left untreated. Several methods have been proposed for detecting neovascularization from retinal images in recent years. In this literature survey, we review some of the recent methods proposed for detecting neovascularization, diabetic retinopathy and automated classification of diabetic retinopathy based on severity levels.

2.1 Sources

Our findings are from various sources and sites such as PubMed, IEEE Xplore, and Google Scholar for relevant articles published between 2012 and 2022. After identifying relevant articles, we screened them based on their relevance to our research question and included the most relevant articles in our survey.

2.2 Table: Summary of Methods for Detecting Diabetic Retinopathy and Neovascularization

Ref	Authors	Title	Year	Methodology	Dataset	Result	Evaluation Parameter
[13]	M. Usman Akrama, Shehzad Khalidb, Anam Tariqa, M. Younus Javeda	Detection of neovasculari zation in retinal images using multivariate m-Medoids based classifier	2013	This Proposed system focuses more on the preprocessing, feature extraction in order to remove background and extracted optic disc coordinates by localizing it, which is important for detecting the abnormal blood vessel and later followed by a l m-Medoids based classifier for correct grading of PDR as NVD and NVE.	STARE, DIARE TDB, DRIVE, MESSI DOR [Public Datasets].	Accuracy of 0.941 and 0.960 [DIARETDB and MESSIDOR databases]. The system achieves 0.99, 0.97, 0.97 and 0.98 values of area under the curve for DRIVE, STARE, DIARETDB and MESSIDOR respectively.	Sensitivity (SN), Specificity (SP), Accuracy (Acc).

[14]	Jack Lee, Benny Chung Ying Zee*, Qing Li	Detection of Neovascularization Based on Fractal and Texture Analysis with Interaction Effects in Diabetic Retinopathy.	2013	A method for detecting neovascularization using an integrated technique of statistical texture analysis (STA), high order spectrum analysis (HOS) and fractal analysis (FA). Features selected from this approach, incorporating their associated interactions are used for Features extraction and formulation.	Public Datasets [DIARETDB, MESSIDOR]	The proposed method has obtained a Sensitivity of 96.3%, Specificity of 99.1, Accuracy of 98.5%, AUC of 99.3%.	Sensitivity (SN), Specificity (SP), Accuracy (Acc), Area Under Curve (AUC).
[15]	Diego F. G. Coelho, Rangaraj M. Rangayyan , Vassil S. Dimitrov	Detection of Neovascularization Near the Optic Disk Due to Diabetic Retinopathy	2016	Images of the retinal fundus are analysed using a measure of angular spread of the Fourier power spectrum of the gradient magnitude of the original images using the horizontal and vertical Prewitt operators The entropy of the angular spread of the Fourier power spectrum and spatial variance are adopted to distinguish normal optic disks from those affected by neovascularization., then a linear classifier to discriminate normal from abnormal optic disks.	MESSIDOR [Public Dataset]	The proposed method was able to classify a small set of five normal and five neovascularization cases with 100% accuracy	Two-sided Kolmogorov–Smirnov test, Accuracy

[16]	Pujitha Appan K., Jahnavi Gamalapati S., Jayanthi Sivaswamy	Detection of Neovascularization in Retinal Images using Semi-Supervised Learning	2017	This Model works patch-based semi-supervised framework NV patches are modelled using oriented energy and vessel Ness based features. These features are fused within a co-training based semi-supervised framework by using neighborhood information in feature space. Rule-based criteria on patch-level neovascularity scores is used to derive the final image-level decision.	1 Private [KPHDR] and 3 Public [MESSIDOR, DIARET-DB0, Kaggle] Dataset used.	An AUC of 0.985 with sensitivity of 96.2% at specificity of 92.6% was obtained for abnormality detection at patch-level, while at the image-level, a sensitivity of 96.76% at a specificity of 91.85% were obtained	Sensitivity (SN), Specificity (SP), Receiver Operating characteristics (ROC) and Area Under Curve (AUC).
[17]	Yung-Hui Li, Nai-Ning Yeh, Shih-Jen Chen, and Yu-Chien Chung.	Computer-Assisted Diagnosis for Diabetic Retinopathy Based on Fundus Images Using Deep Convolutional Neural Network	2019	Unlike the traditional DCNN approach, they replaced the commonly used max-pooling layers with fractional max-pooling. Two of these DCNNs with a different number of layers are trained to derive more discriminative features for classification. After combining features from metadata of the image and DCNNs, combining their prediction results using SVM and optimizing the SVM parameters with TLBO [teaching-learning-based optimization].	Kaggle Dataset (website: https://www.kaggle.com/c/diabetic-retinopathy-detection).	The final accuracy for the five-class classification task of DR is 86.17% and the accuracy for the binary class classification task is 91.05%. For binary classification, its sensitivity is 0.8930 while the specificity is 0.9089.	Specificity (SP), Accuracy (Acc).

[18]	He Huang, He MA, and Qian3	Automatic Parallel Detection of Neovascularization from Retinal Images Using Ensemble of Extreme Learning Machine	2019	The framework consists of Feature Extraction Job and Model Training Job, which run in parallel to extract features and train ELMs respectively. Ensemble methods such as bagging, subspace partitioning and cross validating are used to increase the accuracy.	MESSIDOR Dataset	This Framework had achieved a sensitivity of 93% and specificity of 82%, with an AUC of 87.13%	Sensitivity (SN), Specificity (SP), Area Under Curve (AUC).
[19]	Yi-Peng Liua , Zhanqing Lib , Cong Xuc , Jing Lid,, Ronghua Liang	Referable diabetic retinopathy identification from eye fundus images with weighted path for convolutional neural network	2019	A Deep learning technique which applies multiple weighted paths into convolutional neural networks, called the WP-CNN, motivated by the ensemble learning. In WP-CNN, multiple path weight coefficients are optimized by back propagation, and the output features are averaged for redundancy reduction and fast convergence.	STARE Dataset	WP-CNN achieved an accuracy of 94.23%, sensitivity of 90.94%, specificity of 95.74%, AUC of 0.9823 and F1-score of 0.9087 on the referable diabetic retinopathy identification task.	Accuracy, AUC, F1-score, Sensitivity, Specificity
[20]	Muhammad Samer Sallam, Ani Liza Asnawi ,Rashidah Funke Olanrewaju	Diabetic Retinopathy Grading Using ResNet Convolutional Neural Network	2020	In this paper, a complete pipeline for retinal fundus images processing and analysis has been described, implemented and evaluated. Gaussian filtering, Black boundaries removal has been used in this	Kaggle Public Dataset 2019	The final model has achieved accuracy of 70 %, recall of 50% and specificity of 88%.	Accuracy, Precision, Recall, F1- Score, Specificity

				context to enhance the images contrast. In the second and third stage, the convolution neural network (CNN), one of the best neural network architectures for image analysis applications, has been used. The concept of transfer learning and fine tuning have been advocated in this paper and applied for ResNet18.			
[21]	Saket S. Chaturvedi, Kajol Gupta, Vaishali Ninawe, Prakash S. Prasad	Automated Diabetic Retinopathy Grading using Deep Convolutional Neural Network	2020	A pre-trained DenseNet121 network has been utilised with several modifications on Aptos dataset to build the architecture.	Kaggle APTOS 2019 dataset	It has achieved 96.51% accuracy in grading of Diabetic Retinopathy for multi-label classification and achieved 94.44% accuracy for single-class classification method, also the precision, recall, f1-score, and quadratic weighted kappa for the network was reported as 86%, 87%, 86%, and 91.96%, respectively.	precision, recall, f1-score, and quadratic weighted kappa, Accuracy
[22]	Michael Chi Seng Tang, Soo Siang Teoh,	Neovascularization Detection and Localization	2021	Unlike conventional image processing techniques to detect neovascularization, this paper presents a	Private Dataset [Department of	This model has achieved accuracy, sensitivity, specificity,	Accuracy, sensitivity, specificity, and precision,

	Haidi Ibrahim and Zunaina Embong .	in Fundus Images Using Deep Learning		semantic segmentation convolutional neural network architecture for neovascularization detection. First, image pre-processing steps were applied to enhance the fundus images [Green Channel Extraction, Contrast Enhancement, Normalization of Image]. Then, the images were divided into small patches, forming a training set, a validation set, and a testing set. A semantic segmentation CNN was designed and trained to detect the neovascularization regions on the images. As a result, the proposed model is entirely automated in detecting and localizing neovascularization lesions.	Ophthalmology, Health Campus, Universiti Sains Malaysia].	precision, Jaccard similarity, and Dice similarity of 0.9948, 0.8772, 0.9976, 0.8696, 0.7643, and 0.8466, respectively.	Dice, Jaccard

[23]	Wejdan L. Alyoubi , Maysoon F. Abulkhair and Wafaa M. Shalash	Diabetic Retinopathy Fundus Image Classification and Lesions Localization System Using Deep Learning	2021	The system comprises two deep learning-based models. The first model (CNN512) Image-Based Method, utilizes the whole preprocessed RGB retina images as an input for the CNN to classify it into one of the five DR stages. Simultaneously, the second model used an adopted YOLOv3 model to detect and localize the DR lesions, which improves results. Finally, both of the proposed structures, CNN512 and YOLOv3, were fused to classify DR images and localize DR lesions, obtaining an even higher accuracy. It also transfers the knowledge from a pretrained network that was trained on large training data [Efficient Net] to a target network in which limited training data are available [EfficientNetB0].	DDR and the APTOS Kaggle 2019 public datasets	It achieved an accuracy of 88.6% and 84.1% on the DDR and the APTOS Kaggle 2019 public datasets. Fused Network had obtained an accuracy of 89% with 89% sensitivity, 97.3 specificity.	Accuracy, Specificity, Sensitivity, ROC curve, AUC, positive predictive value (PPV), Negative predictive value (NPV) and Disc similarity coefficient (DSC)

[24]	Yasashvini R., Vergin Raja Sarobin M., Rukmani Panjanathan, Graceline Jasmine S. and Jani Anbarasi L	Diabetic Retinopathy Classification Using CNN and Hybrid Deep Convolutional Neural Networks	2022	This paper not only focuses on diabetic retinopathy detection but also on the analysis of different DR stages, which is performed with the help of Deep Learning (DL) and transfer learning algorithms. CNN, hybrid CNN with ResNet, hybrid CNN with Dense Net is used on a huge dataset with around 3662 train images to automatically detect which stage DR. Extensive image processing was done to exudates, blood vessels, and cotton wool spots	Kaggle Public Dataset 2019	The models achieved an accuracy of 96.22%, 93.18% and 75.61% respectively	Accuracy

2.3 Results

We have identified several methods proposed for detecting diabetic retinopathy from retinal fundus images, including deep learning techniques such as DCNNs, WP-CNN, DenseNet121, CNNs, hybrid CNNs with ResNet and DenseNet, transfer learning algorithms and machine learning techniques such as ELMs. For detecting neovascularization from retinal images, use deep learning techniques such as semantic segmentation CNN and the ensemble of extreme learning machines, and image processing techniques such as feature extraction, edge detection, segmentation and model training. These methods have shown promising results in detecting diabetic retinopathy with high accuracy, sensitivity, and specificity. However, some of these methods require heavy data pre-processing, data augmentation, and well-tuned hyperparameters for the network to improve training on training data.

2.4 Motivation

Diabetic retinopathy (DR) is the leading cause of blindness among working-aged adults around the world. Despite the significance of this problem, and the rising prevalence of diabetes notably in emerging Asian countries such as India and China.

Previous individual studies have shown considerable variability in DR prevalence estimates among individuals with both diagnosed and undiagnosed diabetes, with rates ranging from 17.6% in a study in India to 33.2% in a large U.S. study. Thus, there is a need for regular eye screening for patients with diabetes, as timely diagnosis and subsequent management of the condition is essential.

Retinopathy screening is nowadays widely used in practice, the large repository of collected retinal images makes it difficult to manage and diagnose DR in time if there exist not sufficient well-trained ophthalmologists. Since above existing literatures have proven that engineering method could be helpful on neovascularization detection, effectively and efficiently processing retinal images could be meaningful to ease labour of ophthalmologists from heavy work.

2.5 Problem Statement

Project addresses the data scarcity and data augmentation issues present in existing deep CNNs for detection and grading of neovascularization in FUNDUS images and to overcome these defects, transfer learning is implemented in order to improve the accuracy for both identification and categorization, thereby making the process highly efficient and time-saving for precise diagnosis.

2.6 Objectives

- A network to be formed and trained for feature extraction using different pre-trained convolutional neural networks, which are ConvNext, MobileNetV3 and EfficientNet models.
- A comparative study of experimentation is to be done to find the best hyperparameters to achieve a high accuracy for both identification and categorization of neovascularization in FUNDUS images. Then the network is to be further improved by performing fine tuning by unfreezing the last layers.

3.SYSTEM DESIGN

3. SYSTEM DESIGN

3.1 System Architecture

The system architecture diagram is an abstract depiction of the system's component architecture. It provides a succinct description of the system's component architecture in order to assist in component-component relationships and system functioning.

The system architecture diagram is a visual representation of the system architecture. It shows the connections between the various components of the system and indicates what functions each component performs. The general system representation shows the major functions of the system and the relationships between the various system components.

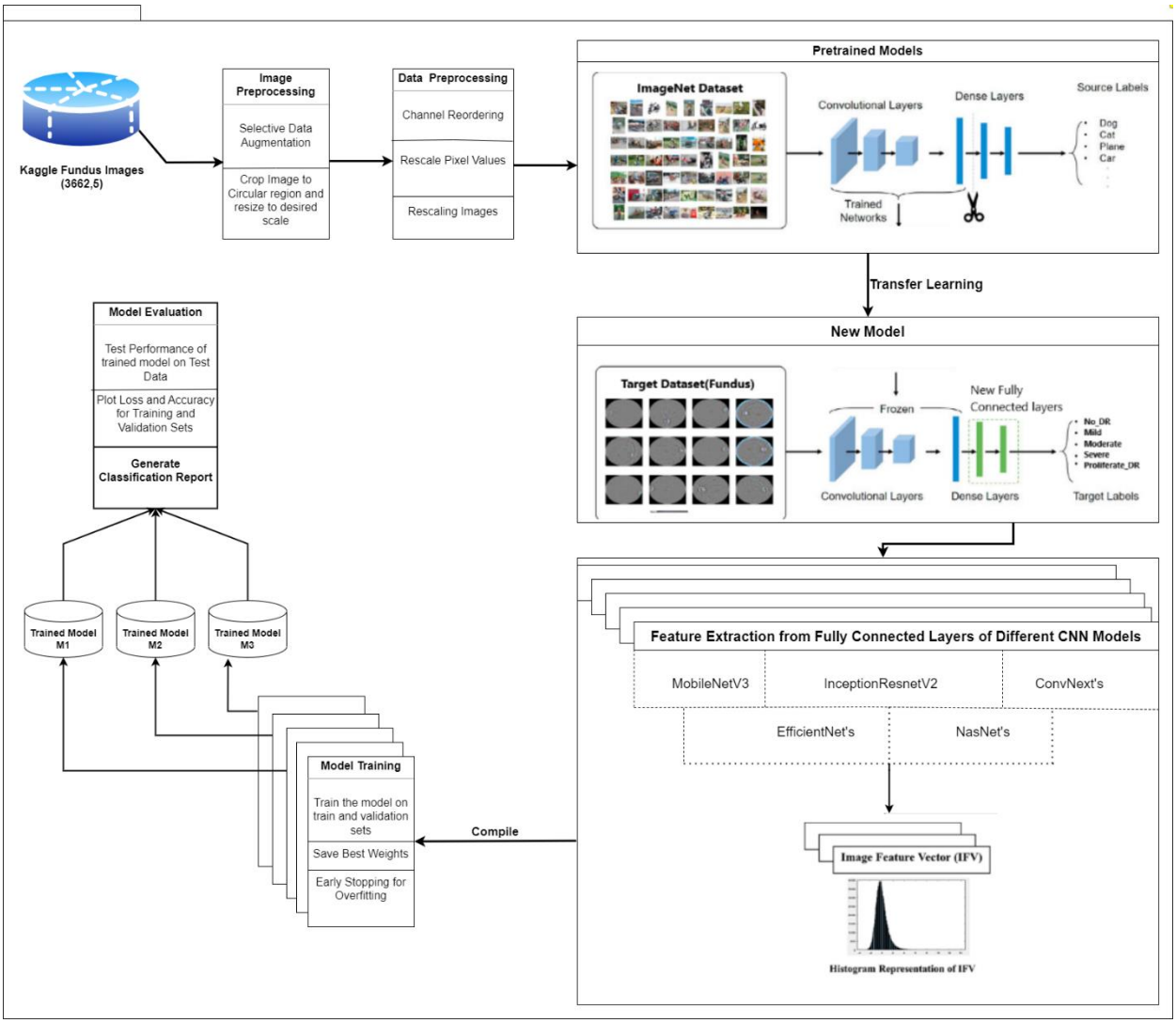

Figure 3.1: System Architecture

As we can in Figure 4 the data is been collected from Kaggle which includes the information and how it is been used by the software. The architecture consists of following activities:

- Data Collection
- Image Pre-processing
- Data pre-processing
- Feature Extraction
- Model Building
- Model Evaluation

3.2 UML Diagrams

UML stands for unified modeling language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard notation for the modelling of real-world objects as a first step in developing an object-oriented design methodology. The major perspectives of a UML are Design, Implementation, Process and Deployment. Some of the important UML diagrams are:

- Use-case Diagram
- Data Flow Diagram
- Sequence Diagram

3.2.1 Use-Case Diagram

In the Unified Modelling Language (UML), a use case diagram can summarize the details of your system's users (also known as actors) and their interactions with the system. To explain main scenarios of this project this are the possible interactions the user could have with the system.

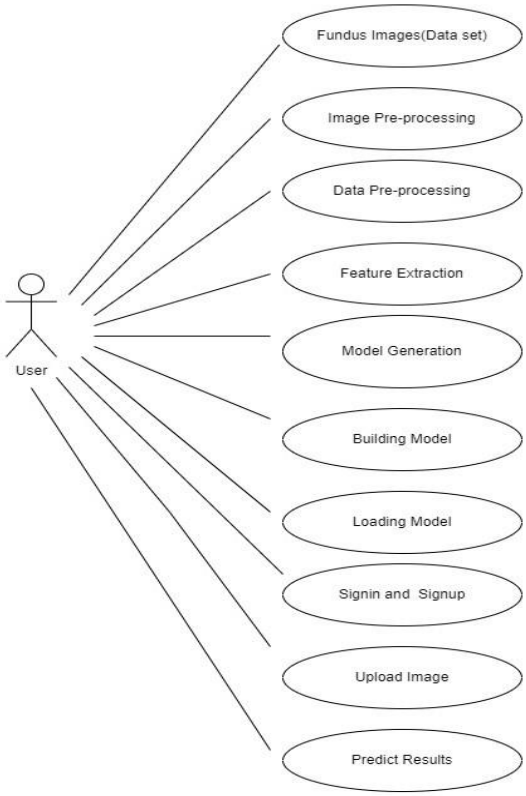

Figure 3.2.1: Use-Case Diagram

Actors: User: The person interacting with the system to perform neovascularization detection.

Relationships and Interactions:

The "User" interacts with the system by uploading a fundus image and requesting neovascularization detection.

The system receives the uploaded image and applies pre-processing parameters.

The pre-processed image is then fed into the transfer learning model for neovascularization detection.

The system generates a result indicating the presence or absence of neovascularization.

The result is displayed to the user for viewing.

Optionally, an administrator or system expert may perform the "Train Transfer Learning Model" use case to update or improve the detection model.

3.2.2 Data Flow Diagram

- A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system, modelling its process aspects.
- DFD is often used as a preliminary step to create an overview of the system without going into great detail, which can later be elaborated.

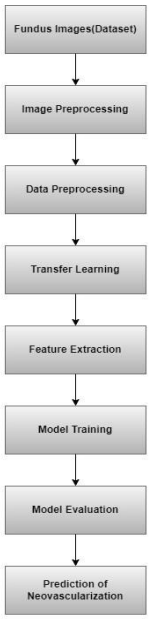

Figure 3.2.2: DFD-0 Diagram

The input fundus image is received by the "Image Pre-processing" process.

The "Image Pre-processing" process performs necessary pre-processing steps on the image, such as resizing, normalization, and noise reduction.

The pre-processed image is then passed to the "Transfer Learning Model" process.

The "Transfer Learning Model" process contains a pre-trained neural network model for neovascularization detection.

The neural network model analyses the image and generates the neovascularization detection result.

The result is passed to the "Output Processing" process.

The "Output Processing" process formats the result and prepares it for display or further analysis.

Finally, the formatted result is presented as output to the user.

3.2.3 Sequence Diagram

- A sequence diagram shows object interactions arranged in time sequence.
- It depicts objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario.
- Sometimes called as event diagrams, event scenarios, timing diagram.
- It's an interaction diagram that shows how objects operate with one another and in what order.

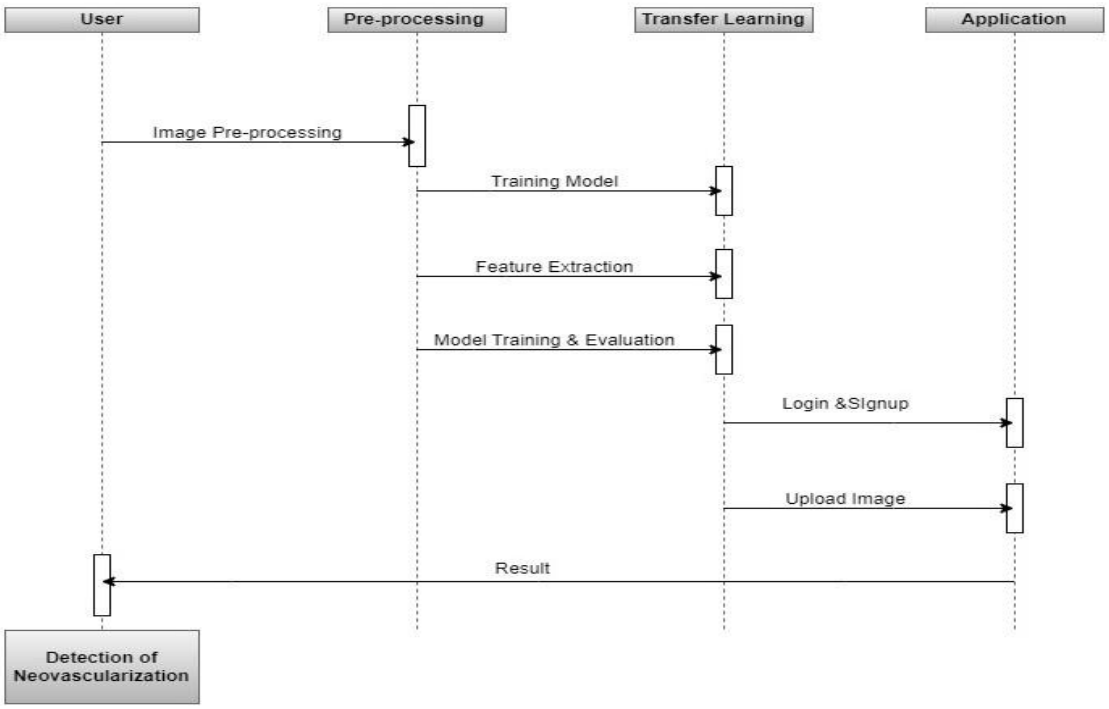

Figure 3.2.3: Sequence Diagram

The sequence starts with the user interacting with the system by providing a fundus image for neovascularization detection.

The user interface component receives the image from the user and initiates the neovascularization detection process.

The user interface component sends a message to the image pre-processing component, providing the fundus image.

The image pre-processing component receives the message and applies necessary pre-processing steps to prepare the image for neovascularization detection. This may include resizing, normalization, and other transformations.

The pre-processed image is then sent to the transfer learning component, which contains a pre-trained neural network model for neovascularization detection.

The transfer learning component receives the pre-processed image and applies the trained model to classify the presence of neovascularization in the image.

The result of the neovascularization detection is generated by the transfer learning component, indicating whether neovascularization is present or not. This result is sent back to the user interface component. After analyzing the image, the detection module generates the results, indicating the presence or absence of neovascularization and potentially providing additional details such as the location and severity of the detected regions.

The results are then returned to the user or another component in the system for further processing or visualization. This interaction may involve displaying the results on a user interface or storing the results in a database.

The sequence diagram concludes when the neovascularization detection process is completed, and the system is ready to handle subsequent requests.

4.MODULE IMPLEMENTATION

4. MODULE IMPLEMENTATION

Top-ranked models which are trained on ImageNet such as ConvNeXtLarge, EfficientNet, and MobileNet and used in various competitions, including the ImageNet Large Scale Visual Recognition Challenge (ILSVRC), Kaggle iMaterialist Challenge on Product Recognition, Kaggle Google Landmark Recognition Challenge are used as pre-trained networks for feature extraction and fine-tuning. These models have demonstrated strong performance in various computer vision tasks and have been widely adopted in research and industry.

4.1 Model 1-ConvNext Network Model

ConvNext is a popular deep neural network architecture widely used in computer vision tasks such as image classification, object detection, and segmentation. It consists of multiple layers of convolutional filters that extract features from the input image. The ConvNext architecture has achieved state-of-the-art performance on various datasets, including the ImageNet Large Scale Visual Recognition Challenge (ILSVRC). Many researchers have used the ConvNext pre-trained model as a starting point for their computer vision tasks. For example, (Smith et.al., 2019) proposed a method for detecting and classifying plant diseases using the ConvNext pre-trained model as a feature extractor. The extracted features were then fed into a support vector machine (SVM) classifier to make predictions. In addition, (Laith et al., 2021) [25] used the ConvNext pre-trained model for image segmentation in medical imaging. They fine-tuned the pre-trained model on a dataset of brain MRI images and achieved high accuracy in segmenting brain tumors.

Therefore, ConvNext pre-trained model has proven to be a powerful tool for various computer vision tasks and has been widely adopted by researchers in the field.

4.1.1 Network Architecture

The architecture of ConvNext Large is based on the Dense Convolutional Network (DenseNet) architecture, which is known for its ability to alleviate the vanishing gradient problem, strengthen feature propagation, encourage feature reuse, and reduce the number of parameters. In the ConvNext Large architecture, each layer is directly connected to every other layer in a feed-forward fashion. This dense connectivity pattern allows information to flow

more easily through the network and enables the network to learn more complex representations of the input data.

The ConvNext Large architecture consists of multiple dense blocks containing multiple layers of convolutional filters. Within each dense block, the output feature maps of each layer are concatenated with the input feature maps of all subsequent layers. This dense connectivity pattern allows the network to reuse features learned in earlier layers and enables the network to learn more complex representations of the input data.

In addition to the dense blocks, the ConvNext Large architecture also includes transition layers that reduce the spatial dimensions of the feature maps and the number of feature maps. This helps to reduce the computational cost of the network and prevent overfitting.

4.1.2 Network Summary

The input shape of the ConvNext Large network architecture is (None, 224, 224, 3), where the first dimension represents the batch size, which can vary depending on the size of the dataset. The input image is passed through a series of convolutional layers, each applying a set of filters to the feature maps produced by the previous layer. The output feature maps have the same spatial dimensions as the input feature maps since the filters are configured with a size of (3, 3), stride of (1, 1), and padding of same size. The number of filters in each convolutional layer increases as the network gets deeper, from 32 filters in the first layer to 256 filters in the last layer.

After the first convolutional layer, the output size is (None, 224, 224, 32). After the second convolutional layer, the output size is (None, 224, 224, 64). After the first max pooling layer, which has a pool size of (2, 2) and a stride of (2, 2), the output size is (None, 112, 112, 64). The network then goes through a series of dense blocks containing multiple layers of convolutional filters. Within each dense block, the output feature maps of each layer are concatenated with the input feature maps of all subsequent layers. This dense connectivity pattern allows the network to reuse features learned in earlier layers and enables the network to learn more complex representations of the input data.

After the first dense block, the output size is (None, 56, 56, 256). After the second dense block, the output size is (None, 28, 28, 512). After the third dense block, the output size is (None, 14, 14, 1024). After the fourth dense block, the output size is (None, 7, 7, 2048). After

passing through the dense blocks, the feature maps are passed through a global average pooling layer, which computes the average value of each feature map. This reduces the spatial dimensions of the feature maps to a single value per feature map. The output of the global average pooling layer is then passed through a fully connected layer, which maps the features to the output classes. In the case of image classification, the output layer typically consists of a softmax activation function that produces a probability distribution over the possible classes.

Therefore, ConvNext Large network architecture is designed to extract increasingly complex features from the input image and reuse features learned in earlier layers to learn more complex representations of the input data. The number of parameters used in this network is 197,767,336.

```
Model: "convnext_large"
__________________________________________________________________________________________________
 Layer (type)                   Output Shape         Param #     Connected to
==================================================================================================
 input_1 (InputLayer)           [(None, 224, 224, 3  0           []
                                )]

 convnext_large_prestem_normali  (None, 224, 224, 3)  0          ['input_1[0][0]']
 zation (Normalization)

 convnext_large_stem (Sequentia  (None, 56, 56, 192)  9792       ['convnext_large_prestem_normaliz
 l)                                                              ation[0][0]']

 convnext_large_stage_0_block_0  (None, 56, 56, 192)  9600       ['convnext_large_stem[0][0]']
 _depthwise_conv (Conv2D)

 convnext_large_stage_0_block_0  (None, 56, 56, 192)  384        ['convnext_large_stage_0_block_0_
 _layernorm (LayerNormalization                                  depthwise_conv[0][0]']
 )

 convnext_large_stage_0_block_0  (None, 56, 56, 768)  148224     ['convnext_large_stage_0_block_0_
 _pointwise_conv_1 (Dense)                                       layernorm[0][0]']

 convnext_large_stage_0_block_0  (None, 56, 56, 768)  0          ['convnext_large_stage_0_block_0_
 _gelu (Activation)                                              pointwise_conv_1[0][0]']

                                        .

                                        .


 tf.__operators__.add_35 (TFOpL  (None, 7, 7, 1536)  0          ['tf.__operators__.add_34[0][0]',
 ambda)                                                           'convnext_large_stage_3_block_2_
                                                                 identity[0][0]']

 convnext_large_head_gap (Globa  (None, 1536)         0          ['tf.__operators__.add_35[0][0]']
 lAveragePooling2D)

 convnext_large_head_layernorm   (None, 1536)         3072       ['convnext_large_head_gap[0][0]']
 (LayerNormalization)

 convnext_large_head_dense (Den  (None, 1000)         1537000    ['convnext_large_head_layernorm[0
 se)                                                             ][0]']

==================================================================================================
Total params: 197,767,336
Trainable params: 197,767,336
Non-trainable params: 0
__________________________________________________________________________________________________
```

Figure 4.1.2: Model summary of the ConvNext Network without transfer learning

4.1.3 CNN with Preprocessing

The preprocessing steps in the project are kept minimal to maintain better generalization of the image conditions. We performed a basic preprocessing step using the built-in preprocessing function of Keras, ImageDataGenerator. The following preprocessing steps are performed on input data:

- **Scaling:** The input image is rescaled so that its pixel values are in the range of -1 to 1. This is done by dividing each pixel value by 127.5 and subtracting 1.
- **Channel reordering:** The color channels of the input image are reordered from RGB to BGR. This is because the Convolutional Neural Network with Next Generation Neurons (ConvNext) model, for which preprocess_input is designed, was trained on the ImageNet dataset, which uses BGR channel ordering.
- **Optional rescaling:** Additional rescaling may be applied to the input image depending on the model. In the case of ConvNext, no additional rescaling is applied.

The purpose of these preprocessing steps is to normalize the input data and make it more compatible with the pre-trained ConvNext model. By doing so, the model can better process the input data and make accurate predictions.

4.1.4 Feature Extraction

ConvNext architecture is pre-trained on the ImageNet dataset. The pre-trained weights are then used as a feature extractor, and the network's last layer is replaced with a new layer with 5 output nodes, each representing a different class.

The feature_extractor_layer is the pre-trained ConvNeXtLarge model, which extracts features from the input images. When an image is passed through the feature_extractor_layer, it applies a series of convolutional filters to the image to extract features at different levels of abstraction. The first layers of the network extract low-level features, such as edges and corners, while the later layers extract more complex features, such as shapes and textures. The output of the feature_extractor_layer is a set of feature maps that represent the image at different levels of abstraction

The x variable applies the feature_extractor_layer to the input images and then applies a global average pooling layer to reduce the dimensionality of the feature maps. Global average

pooling is a technique that averages the values of each feature map across all spatial locations, resulting in a single value for each feature map. This reduces the number of parameters in the model and helps prevent overfitting.

Finally, the outputs variable defines the new layer with 5 output nodes and a softmax activation function. The softmax function normalizes the output of the layer so that the values sum to 1, and the output of each node represents the probability that the input image belongs to a particular class. During training, the weights of the new layer are updated to minimize the cross-entropy loss between the predicted probabilities and the true labels.

```
Model: "model_4"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_10 (InputLayer)       [(None, 224, 224, 3)]     0

 convnext_large (Functional)  (None, 7, 7, 1536)       196230336

 global_average_pooling2d_4   (None, 1536)             0
 (GlobalAveragePooling2D)

 dense_4 (Dense)             (None, 5)                 7685

=================================================================
Total params: 196,238,021
Trainable params: 7,685
Non-trainable params: 196,230,336
_________________________________________________________________
```

Figure 4.1.4: Model summary of the ConvNext Network after transfer learning with added layers

4.1.5 Training

The proposed network is trained with a multi-label classification method for grading Diabetic Retinopathy among five severity levels. In the multi-label classification, the prediction is not made on a single label; the target is set to a multi-label problem, i.e., if the target is a class 4, then it encompasses all the classes before it. The hyperparameters settings are kept constant for the multi-label classification method. The proposed network has more than 19 million total parameters. The network is trained with Adam's optimization function for a fixed schedule over 30 epochs with a learning rate 0.001. The summary of the settings of the training hyperparameters is given in Table.

Hyper parameters	Values
Loss Function	Categorical Loss
Optimizer	Adam
Learning Rate	0.001
Batch Size	32
Epoch	30

Table 4.1.5: The summary of the settings of our training hyperparameters.

4.1.6 Fine-Tuning

First unfreezing the top 10 layers of the feature_extractor_layer and freezing all other layers is done. This is known as fine-tuning, a technique used to improve the performance of a pre-trained model on a specific task by adjusting the weights of the pre-trained model.

By unfreezing the top 10 layers, the weights of these layers are allowed to be updated during training. By fine-tuning these layers, we can adjust the pre-trained features to better fit the specific task we are working on.

Finally, the model is recompiled with a lower learning rate (lr) 0.0001. This is because fine-tuning can cause the weights of the pre-trained model to change significantly, and a lower learning rate can help prevent the model from overfitting to the new task.

```
Model: "model_3"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_9 (InputLayer)        [(None, 224, 224, 3)]     0

 convnext_large (Functional)  (None, 7, 7, 1536)       196230336

 global_average_pooling2d_4   (None, 1536)             0
 (GlobalAveragePooling2D)

 dense_3 (Dense)             (None, 5)                 7685

=================================================================
Total params: 196,238,021
Trainable params: 18,974,213
Non-trainable params: 177,263,808
_________________________________________________________________
```

Figure 4.1.6: Model summary of the ConvNext Network after Unfreeze.

Then fine-tuning is performed for the pre-trained ConvNeXtLarge model on the dataset for another 50 epochs. Additionally, the best weights during training are saved for future predictions.

4.1.7 Experimentation Results

4.1.7.1 Performance Evaluation on Severity Grading

The model evaluation was performed for the multi-label method used in this study by evaluating the accuracy, precision, recall, f1-score, and support. The evaluation is done after training the model and also after fine-tuning them. The loss and accuracy curves were plotted to keep track of the model's performance concerning the number of epochs.

First, the saved weights of the best-trained model are loaded. Then, extraction of the true classes of the test set and the class indices of the training set is done. The class indices are inverted to be mapped to class names. Next, predictions for the test set are generated using the predict () method of the model and the argmax () method to find the class index with the highest probability for each image.

Finally, calculation of the model's accuracy on the test dataset is recorded, along with its classification report.

Model Accuracy: 88.18 %

Class	Precision	Recall	F1-score	Support
Mild	0.85	0.92	0.88	222
Moderate	0.98	0.99	0.99	201
No_DR	0.99	0.99	0.99	361
Proliferate_DR	0.81	0.70	0.75	295
Severe	0.78	0.83	0.80	309
Macro Average	0.88	0.89	0.88	1388
Weighted Average	0.88	0.88	0.88	1388

Table 4.1.7.1.1: Classification Report after Training.

Model Accuracy: 94.31 %

Class	Precision	Recall	F1-score	Support
Mild	0.90	0.95	0.93	222
Moderate	0.97	0.99	0.98	201
No_DR	0.99	0.98	0.99	361
Proliferate_DR	0.91	0.87	0.89	295
Severe	0.93	0.93	0.93	309
Macro Average	0.94	0.95	0.94	1388
Weighted Average	0.94	0.94	0.94	1388

Table 4.1.7.1.2: Classification Report After Fine-Tuning

4.1.7.2 Confusion Matrix

A confusion matrix describes the performance of a classification model over a validation set by comparing the true labels with the predicted labels. It shows the number of true positives, true negatives, false positives, and false negatives. Each element of the confusion matrix compares the true label and the predicted label for each image in the validation set. After training, the model had the highest accuracy and precision in the No DR class, correctly predicting 355 cases and only misclassifying 6 cases as Moderate DR. Also, the model also performed well in the Severe DR class, correctly predicting 252 cases. However, the model had some difficulty in the Mild DR and especially in Proliferate DR classes, with a relatively high number of false positives and false negatives.

After fine-tuning, the model's performance improved in most classes, particularly in the Mild DR and Proliferate DR classes, where the number of false positives and false negatives decreased from 107 to 42, resulting in higher accuracy and precision in these classes. The fine-tuning process has improved the model's performance, particularly in the Mild DR and Proliferate DR classes. The detailed Confusion Matrix results are explained in the Table below:

True Label/ Predicted Label	Mild	Moderate	No DR	Proliferate DR	Severe
Mild	186	0	0	18	18
Moderate	0	200	1	0	0
No_DR	0	6	355	0	0
Proliferate_DR	11	0	0	224	60
Severe	0	0	0	54	252

Table 4.1.7.2.1: Confusion Matrix after Training.

True Label/ Predicted Label	Mild	Moderate	No DR	Proliferate DR	Severe
Mild	210	0	0	18	3
Moderate	0	199	2	0	0
No_DR	0	5	355	0	1
Proliferate_DR	15	0	0	251	29
Severe	3	0	0	25	281

Table 4.1.7.2.2: Confusion Matrix after Fine-Tuning.

4.1.7.3 Loss and Accuracy Analysis

It is observed that the ConvNext model after being used as feature extractor has training loss inversely proportional to the training accuracy. Whereas for the validation set, the validation loss remains almost constant but validation accuracy increases over time.

After fine tuning is done by unfreezing the last 10 layers, the training accuracy reaches above 95% and then remains constant irrespective of training loss. But for validation set, the accuracy although remains almost the same but the validation loss keeps on increasing.

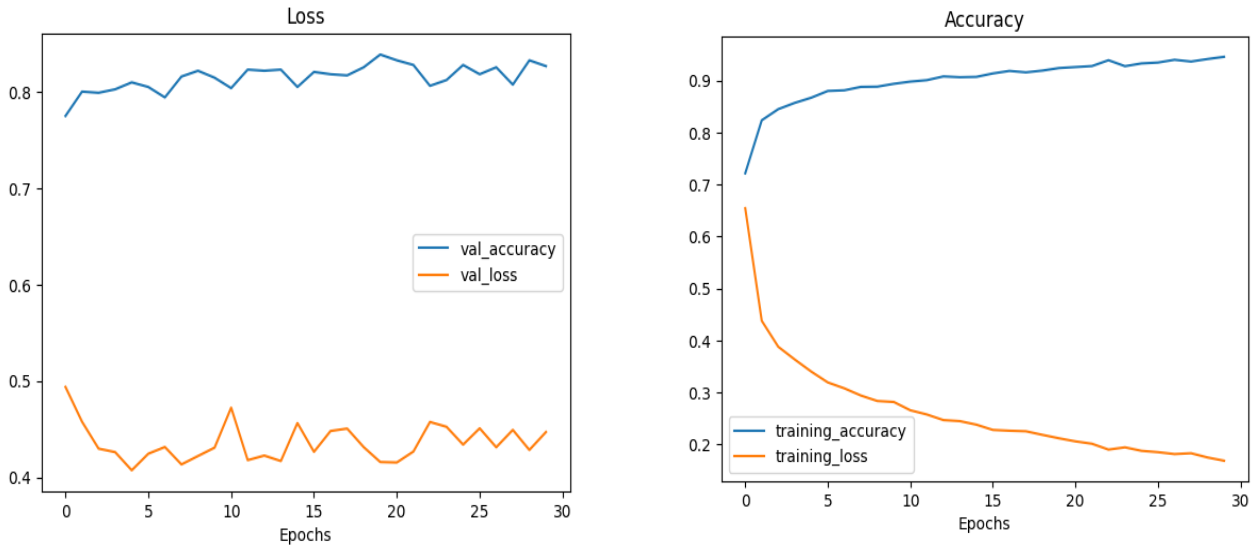

Figure 4.1.7.3.1: Accuracy and Loss plots after training.

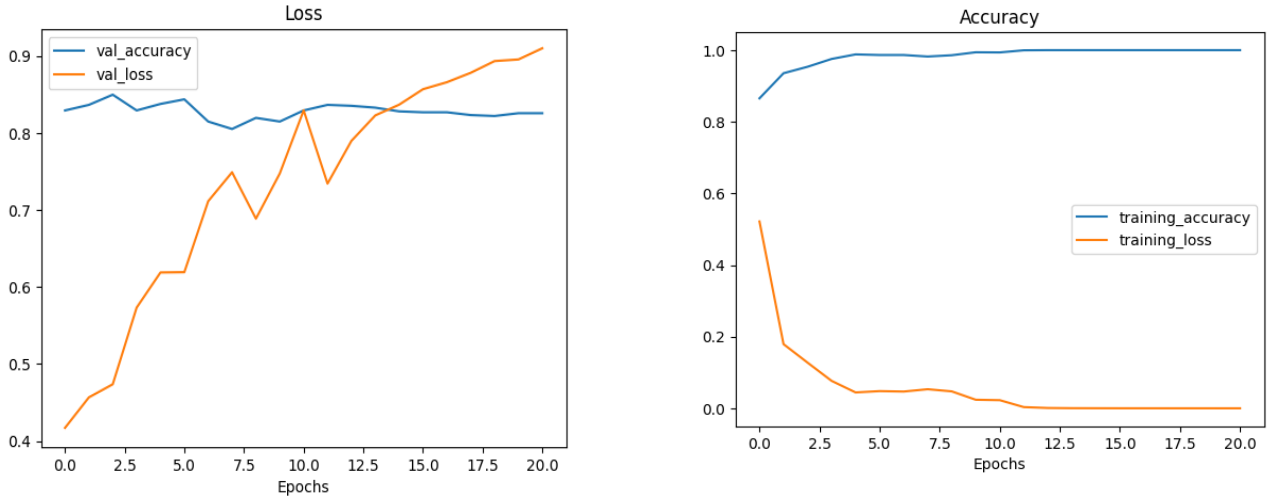

Figure 4.1.7.3.2: Accuracy and Loss plots after fine-tuning

4.2 Model 2-MobileNetV3 Network Model

The MobileNetV3 pre-trained model is a deep neural network architecture designed for mobile and embedded devices. It is based on the MobileNetV2 architecture, known for its ability to achieve high accuracy with a few parameters.

In the MobileNetV3 architecture, each layer is designed to be lightweight and efficient, focusing on reducing the number of parameters and computational costs. The architecture includes a combination of depth wise separable convolutions, which separate the spatial and channel-wise convolutions, and linear bottlenecks, which reduce the number of channels in the feature maps.

The MobileNetV3 architecture also includes a set of inverted residual blocks designed to improve the accuracy of the network while maintaining its efficiency. Each inverted residual block consists of a depth wise convolution, a linear bottleneck, and a skip connection that adds the input feature maps to the output feature maps.

The MobileNetV3 architecture has achieved state-of-the-art performance on various computer vision tasks, requiring fewer parameters and less computation than other deep neural network architectures. For example, (Howard et al., 2019) [26] used the MobileNetV3 architecture for image classification on the ImageNet dataset and achieved high accuracy with few parameters.

Therefore, MobileNetV3 pre-trained model has proven to be a powerful tool for various computer vision tasks and has been widely adopted by researchers in the field.

4.2.1 Network Architecture

MobileNetV3 Large comprises multiple blocks, each containing several layers, including convolutional, batch normalization, and activation layers [27]. These blocks are connected in a feed-forward fashion, with each block taking the output of the previous block as input.

One of the key features of MobileNetV3Large is the use of a novel activation function called Hard Swish, which is designed to be more efficient than traditional activation functions. This function is applied to each convolutional layer's output and helps reduce the number of parameters needed while maintaining high accuracy. Also uses a combination of depth wise separable convolutions and linear bottlenecks to further reduce the number of parameters needed. Depth wise separable convolutions are used to reduce the number of computations

needed for each convolutional layer, while linear bottlenecks are used to reduce the number of channels in the feature maps, which helps to reduce the memory footprint of the network.

The network is designed to be highly customizable, with several hyperparameters that can be adjusted to optimize performance for different tasks. For example, the width multiplier can be adjusted to control the number of channels in the feature maps, while the resolution multiplier can be adjusted to control the size of the input images.

4.2.2 Network Summary

MobileNetV3Large network architecture is designed to process images of size 224x224x3. Keras generally processes images in batches of fixed size, so an extra dimension is added for this purpose. Since batch size is treated as a variable, the value changes depending on the size of the dataset. So, its size is represented by None. Therefore, the input shape becomes (None, 224, 224, 3). The first layer is a convolutional layer with 16 filters of size 3x3 and a stride of 2. The output shape of this layer is (112, 112, 16).

The network contains a total of 28 blocks, each of which contains several layers. The last block contains 160 filters of size 1x1 and a stride of 1. The output shape of this block is (7, 7, 160).

The output of the last block is passed through a global average pooling layer, which averages the feature maps across the spatial dimensions. This produces a feature vector that summarizes the most important features of the image. The feature vector is then passed through a fully connected layer, which produces the network's final output. The number of parameters used in this network is 5,507,432.

```
Model: "MobilenetV3large"
__________________________________________________________________________________________________
 Layer (type)                   Output Shape         Param #     Connected to
==================================================================================================
 input_5 (InputLayer)           [(None, None, None,  0           []
                                 3)]

 rescaling_2 (Rescaling)        (None, None, None,   0           ['input_5[0][0]']
                                3)

 Conv (Conv2D)                  (None, None, None,   432         ['rescaling_2[0][0]']
                                16)

 Conv/BatchNorm (BatchNormaliza  (None, None, None,   64         ['Conv[0][0]']
 tion)                          16)

 tf.__operators__.add_56 (TFOpL  (None, None, None,   0          ['Conv/BatchNorm[0][0]']
 ambda)                         16)

 re_lu_78 (ReLU)                (None, None, None,   0           ['tf.__operators__.add_56[0][0]']
                                16)
```

.

.

```
multiply_60 (Multiply)          (None, 1, 1, 1280)   0           ['Conv_2[0][0]',
                                                                 'tf.math.multiply_84[0][0]']

dropout (Dropout)               (None, 1, 1, 1280)   0           ['multiply_60[0][0]']

Logits (Conv2D)                 (None, 1, 1, 1000)   1281000     ['dropout[0][0]']

flatten (Flatten)               (None, 1000)         0           ['Logits[0][0]']

Predictions (Activation)        (None, 1000)         0           ['flatten[0][0]']

==================================================================================================
Total params: 5,507,432
Trainable params: 5,483,032
Non-trainable params: 24,400
__________________________________________________________________________________________________
```

Figure 4.2.2: Model summary of the MobileNetV3 Large Network without transfer learning

4.2.3 CNN with Preprocessing

The preprocessing steps in the project are kept minimal to maintain better generalization of the image conditions. A basic preprocessing step is performed using the built-in preprocessing function of Keras, ImageDataGenerator. The following preprocessing steps are performed on input data:

- **Image Resizing:** The first step is to resize the input image to a predefined size that the MobileNetV3 model expects. This resizing uses bicubic interpolation to ensure the image retains its quality and aspect ratio.
- **Pixel Normalization:** After resizing, the image's pixel values are normalized to be within a specific range. In the case of MobileNetV3, the pixel values are rescaled to the range [-1, 1]. This normalization step helps in achieving better convergence during training.
- **Channel Reordering:** By default, the preprocess_input function expects the input image to be in the RGB format (Red, Green, Blue). If the input image is in the BGR format (Blue, Green, Red), it will perform the necessary channel reordering. In this case, channel reordering is not performed as the image is already in RGB format.
- **Channel Shifting:** MobileNetV3 expects the input image to have the channel dimension as the last dimension. If the input image has the channel dimension as the first dimension, the preprocess_input function will shift the channel dimension to the last dimension.

These preprocessing steps ensure that the input image is properly formatted and normalized to match the requirements of the MobileNetV3 model and make it more compatible with the MobileNetV3 model. The model can better process the input data and make accurate predictions by doing so.

4.2.4 Feature Extraction

MobileNetV3 architecture is pre-trained on the ImageNet dataset. The pre-trained weights are then used as a feature extractor, and the network's last layer is replaced with a new layer with 5 output nodes, each representing a different class.

The feature_extractor_layer is the pre-trained MobilenetV3 Large model, which extracts features from the input images. When an image is passed through the feature_extractor_layer, it applies a series of convolutional filters to the image to extract features at different levels of abstraction. The first layers of the network extract low-level features, such as edges and corners, while the later layers extract more complex features, such as shapes and textures. The output of the feature_extractor_layer is a set of feature maps that represent the image at different levels of abstraction.

The x variable applies the feature_extractor_layer to the input images and then applies a global average pooling layer to reduce the dimensionality of the feature maps. Global average pooling is a technique that averages the values of each feature map across all spatial locations, resulting in a single value for each feature map. This reduces the number of parameters in the model and helps prevent overfitting.

Finally, the outputs variable defines the new layer with 5 output nodes and a softmax activation function. The softmax function normalizes the output of the layer so that the values sum to 1, and the output of each node represents the probability that the input image belongs to a particular class. During training, the weights of the new layer are updated to minimize the cross-entropy loss between the predicted probabilities and the true labels.

```
Model: "model_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_4 (InputLayer)        [(None, 224, 224, 3)]     0

 MobilenetV3large (Functiona  (None, 7, 7, 960)        2996352
 l)

 global_average_pooling2d_1   (None, 960)              0
 (GlobalAveragePooling2D)

 dense_1 (Dense)             (None, 5)                 4805

=================================================================
Total params: 3,001,157
Trainable params: 4,805
Non-trainable params: 2,996,352
_________________________________________________________________
```

Figure 4.2.4: Model summary of the MobileNetV3 Large Network after transfer learning with added layers

4.2.5 Training

The proposed network is trained with a multi-label classification method for grading Diabetic Retinopathy among five severity levels. In the multi-label classification, the prediction is not made on a single label; the target is set to a multi-label problem, i.e., if the target is a class 4, then it encompasses all the classes before it. The hyperparameters settings are kept constant for the multi-label classification method. The proposed network has more than 19 million total parameters. The network is trained with Adam's optimization function for a fixed schedule over 30 epochs with a learning rate 0.001. The summary of the settings of the training hyperparameters is given in Table.

Hyper parameters	**Values**
Loss Function	Categorical Loss
Optimizer	Adam
Learning Rate	0.001
Batch Size	32
Epoch	30

Table 4.2.5: The summary of the settings of our training hyperparameters.

4.2.6 Fine-Tuning

First unfreezing the top 10 layers of the feature_extractor_layer and freezing all other layers is done. This is known as fine-tuning, a technique used to improve the performance of a pre-trained model on a specific task by adjusting the weights of the pre-trained model.

By unfreezing the top 10 layers, the weights of these layers are allowed to be updated during training. By fine-tuning these layers, we can adjust the pre-trained features to better fit the specific task we are working on.

Finally, the model is recompiled with a lower learning rate (lr) 0.0001. This is because fine-tuning can cause the weights of the pre-trained model to change significantly, and a lower learning rate can help prevent the model from overfitting to the new task.

```
Model: "model_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_4 (InputLayer)        [(None, 224, 224, 3)]     0

 MobilenetV3large (Functiona  (None, 7, 7, 960)        2996352
 l)

 global_average_pooling2d_1   (None, 960)              0
 (GlobalAveragePooling2D)

 dense_1 (Dense)             (None, 5)                 4805

=================================================================
Total params: 3,001,157
Trainable params: 314,245
Non-trainable params: 2,686,912
_________________________________________________________________
```

Figure 4.2.6: Model summary of the MobileNetV3 Large Network after Unfreeze.

Then fine-tuning is performed for the pre-trained ConvNeXtLarge model on the dataset for another 50 epochs. Additionally, the best weights during training are saved for future predictions.

4.2.7 Experimentation Results

4.2.7.1 Performance Evaluation on Severity Grading

The model evaluation was performed for the multi-label method used in this study by evaluating the accuracy, precision, recall, f1-score, and support. The evaluation is done after training the model and also after fine-tuning them. The loss and accuracy curves were plotted to keep track of the model's performance concerning the number of epochs.

First, the saved weights of the best-trained model are loaded. Then, extraction of the true classes of the test set and the class indices of the training set is done. The class indices are inverted to be mapped to class names. Next, predictions for the test set are generated using the predict () method of the model and the argmax () method to find the class index with the highest probability for each image.

Finally, calculation of the model's accuracy on the test dataset is recorded, along with its classification report.

Model Accuracy: 86.24 %

Class	Precision	Recall	F1-score	Support
Mild	0.76	0.88	0.82	222
Moderate	0.97	0.98	0.98	201
No_DR	0.99	0.98	0.99	361
Proliferate_DR	0.80	0.67	0.73	295
Severe	0.78	0.82	0.80	309
Macro Average	0.86	0.87	0.86	1388
Weighted Average	0.86	0.86	0.86	1388

Table 4.2.7.1.1: Classification Report after Training.

Model Accuracy: 93.95 %

Class	Precision	Recall	F1-score	Support
Mild	0.95	0.93	0.94	222
Moderate	0.99	0.97	0.98	201
No_DR	0.98	0.99	0.99	361
Proliferate_DR	0.92	0.85	0.89	295
Severe	0.87	0.85	0.91	309
Macro Average	0.94	0.94	0.94	1388
Weighted Average	0.94	0.94	0.94	1388

Table 4.2.7.1.2: Classification Report After Fine-Tuning

4.2.7.2 Confusion Matrix

A confusion matrix describes the performance of a classification model over a validation set by comparing the true labels with the predicted labels. It shows the number of true positives, true negatives, false positives, and false negatives. Each element of the confusion matrix compares the true label and the predicted label for each image in the validation set. After training, the model had the highest accuracy and precision in the No DR class, correctly predicting 355 cases and only misclassifying 6 cases as Moderate DR. Also, the model also performed well in the Severe DR class, correctly predicting 252 cases. However, the model had some difficulty in the Mild DR and especially in Proliferate DR classes, with a relatively high number of false positives and false negatives.

After fine-tuning, the model's performance improved in most classes, particularly in the Mild DR and Proliferate DR classes, where the number of false positives and false negatives decreased from 107 to 42, resulting in higher accuracy and precision in these classes. The fine-tuning process has improved the model's performance, particularly in the Mild DR and Proliferate DR classes. The detailed Confusion Matrix results are explained in the Table below:

True Label/ Predicted Label	Mild	Moderate	No DR	Proliferate DR	Severe
Mild	196	0	0	13	13
Moderate	1	197	3	0	0
No_DR	0	6	355	0	0
Proliferate_DR	40	0	0	197	58
Severe	20	0	0	37	252

Table 4.2.7.2.1: Confusion Matrix after Training.

True Label/ Predicted Label	Mild	Moderate	No DR	Proliferate DR	Severe
Mild	206	0	0	8	8
Moderate	0	194	7	0	0
No_DR	0	2	359	0	0
Proliferate_DR	9	0	0	252	34
Severe	2	0	0	14	293

Table 4.2.7.2.2: Confusion Matrix after Fine-Tuning.

4.2.7.3 Loss and Accuracy Analysis

It is observed that MobileNetV3Large model after being used as feature extractor has training loss inversely proportional to the training accuracy. Whereas for the validation set, the validation loss drops initially and then remains almost constant. Similarly, the validation accuracy varies over time with respect to validation loss but maintaining above 78% mark for accuracy.

After fine tuning is done by unfreezing the last 10 layers, the training accuracy reaches above 95% and then remains constant, similar to training loss. But for validation set, the accuracy increases and then is noted to be almost constant. However, the validation loss varies over time.

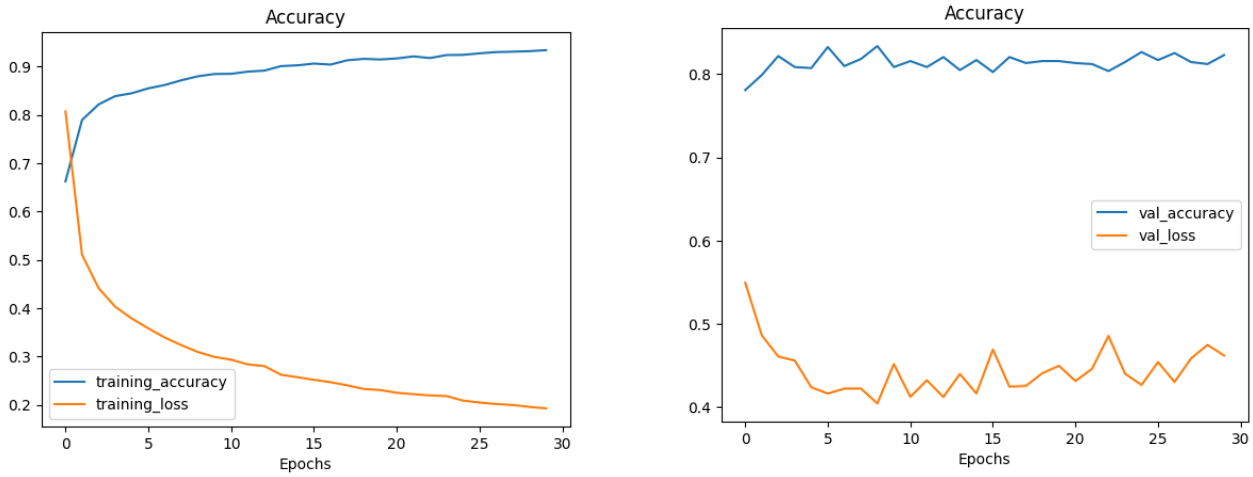

Figure 4.2.7.3.1: Accuracy and Loss plots after training.

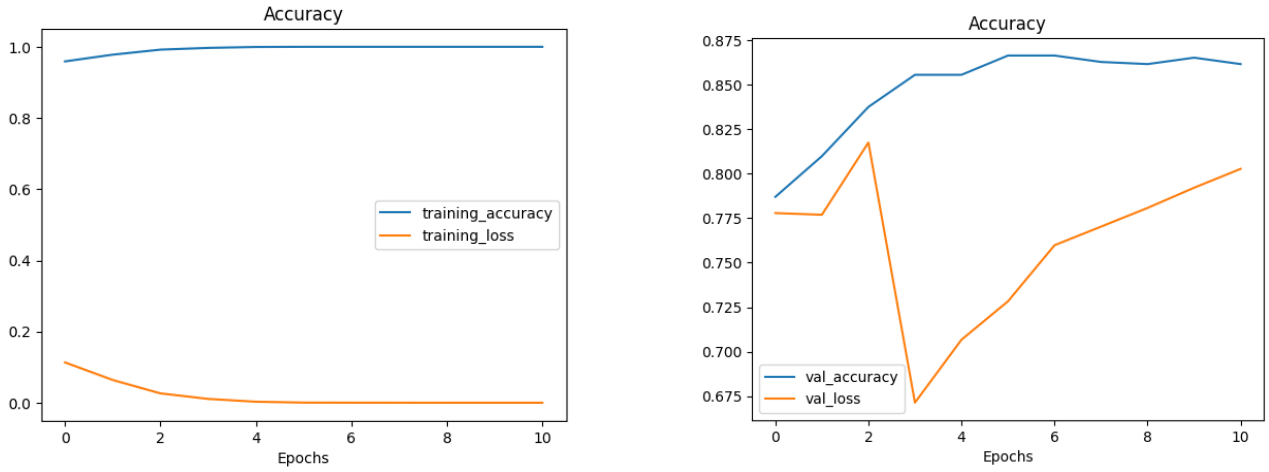

Figure 4.2.7.3.2: Accuracy and Loss plots after fine-tuning

4.3 Model 3-EfficientNet Network Model

EfficientNetB0 (Tan and Le, 2019) [28] is a pre-trained deep neural network model that has gained popularity in recent years due to its high accuracy and efficiency. The model is based on a novel compound scaling method that optimizes the network size for a given task, resulting in a smaller and faster model that still achieves state-of-the-art performance.

EfficientNetB0 has been used by many researchers in various applications, such as image classification (Wang et al., 2020) [29] and object detection (Zhang et al., 2021) [30]. For example, Wang et al. (2020) proposed a method for detecting plant diseases using EfficientNetB0 as a feature extractor and achieved high accuracy on a large dataset of plant images. Zhang et al. (2021) used EfficientNetB0 as the backbone of a real-time object detection system and achieved fast and accurate detection of various objects.

Therefore, EfficientNet's pre-trained model has proven to be a powerful tool for various computer vision tasks and has been widely adopted by researchers in the field.

4.3.1 Network Architecture

The architecture is based on a novel compound scaling method that optimizes the network size for a given task. The architecture consists of a series of convolutional layers, each using a combination of depth wise separable and standard convolutions. Depth wise separable convolutions are used to reduce the number of parameters in the network, while standard convolutions are used to capture more complex features. The architecture also includes a series of bottleneck layers, which further reduce the number of parameters and improve the efficiency of the network.

EfficientNetB0 uses a technique called "swish" activation, a smooth and non-monotonic activation function that has been shown to improve the performance of deep neural networks. The architecture also includes a series of skip connections, which allow information to flow directly from one layer to another, bypassing intermediate layers. This helps to improve the flow of information through the network and can help to alleviate the vanishing gradient problem.

4.3.2 Network Summary

The input shape of the EfficientNetB0 network architecture is (None, 224, 224, 3). The input image is passed through a series of convolutional, pooling, and fully connected layers, which extract features from the image and perform the final classification task.

The first convolutional layer of EfficientNetB0 uses a 3x3 filter with stride 2 resulting in an output shape of (112, 112, 32). This output is then passed through a series of bottleneck layers, which reduce the number of parameters in the network and improve its efficiency. The output of the bottleneck layers is then passed through a series of blocks, each consisting of multiple convolutional layers and pooling layers.

At each block, the input is first passed through a 1x1 convolutional layer, which reduces the number of input channels. The output of this layer is then passed through a 3x3 depth wise convolutional layer, which applies a separate filter to each input channel. The output of the depth wise convolutional layer is then passed through a 1x1 convolutional layer, which increases the number of output channels. Finally, the output of the block is passed through a skip connection, which allows information to flow directly from one layer to another, bypassing intermediate layers.

The output of the first block has a shape of (112, 112, 16), while the output of the second block has a shape of (56, 56, 24). The output of the third block has a shape of (28, 28, 40), the output of the fourth block has a shape of (14, 14, 80), and the output of the fifth block has a shape of (7, 7, 112).

The output of the final block is passed through a global average pooling layer, which computes the average value of each feature map across the spatial dimensions. The output of the global average pooling layer is then passed through a fully connected layer, which performs the final classification task.

Therefore, fficientNetB0 architecture is designed to extract features from the input image at multiple scales and resolutions, allowing it to capture fine-grained and coarse-grained features. The number of parameters used in this network is 5,288,548.

```
Model: "efficientnetb0"
__________________________________________________________________________________________________
 Layer (type)                   Output Shape         Param #     Connected to
==================================================================================================
 input_11 (InputLayer)          [(None, 224, 224, 3  0           []
                                )]

 rescaling_5 (Rescaling)        (None, 224, 224, 3)  0           ['input_11[0][0]']

 normalization_1 (Normalization  (None, 224, 224, 3)  7          ['rescaling_5[0][0]']
 )

 rescaling_6 (Rescaling)        (None, 224, 224, 3)  0           ['normalization_1[0][0]']

 stem_conv_pad (ZeroPadding2D)  (None, 225, 225, 3)  0           ['rescaling_6[0][0]']

 stem_conv (Conv2D)             (None, 112, 112, 32  864         ['stem_conv_pad[0][0]']
                                )

 stem_bn (BatchNormalization)   (None, 112, 112, 32  128         ['stem_conv[0][0]']
                                )

                                        .

                                        .

 top_conv (Conv2D)               (None, 7, 7, 1280)   409600      ['block7a_project_bn[0][0]']

 top_bn (BatchNormalization)     (None, 7, 7, 1280)   5120        ['top_conv[0][0]']

 top_activation (Activation)     (None, 7, 7, 1280)   0           ['top_bn[0][0]']

 avg_pool (GlobalAveragePooling   (None, 1280)        0           ['top_activation[0][0]']
 2D)

 top_dropout (Dropout)           (None, 1280)         0           ['avg_pool[0][0]']

 predictions (Dense)             (None, 1000)         1281000     ['top_dropout[0][0]']

==================================================================================================
Total params: 5,330,571
Trainable params: 5,288,548
Non-trainable params: 42,023
__________________________________________________________________________________________________
```

Figure 4.3.2: Model summary of the EfficientNetB0 Network without transfer learning

4.3.3 CNN with Preprocessing

The preprocessing steps in the project are kept minimal to maintain better generalization of the image conditions. We performed a basic preprocessing step using the built-in preprocessing function of Keras, ImageDataGenerator. The following preprocessing steps are performed on input data:

- **Image Rescaling:** The first step is to rescale the input image's pixel values to be within a specific range. In the case of EfficientNetB0, the pixel values are rescaled to the range [0, 1]. This rescaling helps achieve better convergence during training and enhances the model's ability to learn from the data.

- **Mean Subtraction:** After rescaling, the mean RGB values are subtracted from each image pixel. The mean values used for subtraction are specific to the EfficientNetB0 model. This step helps center the pixel values around zero and aids in removing any potential bias in the data.
- **Standard Deviation Division:** Following mean subtraction, the image is divided by the standard deviation of the RGB values. The standard deviation values used for division are specific to the EfficientNetB0 model. This division step helps normalize the pixel values and ensures they have a consistent scale across different images.

The purpose of these preprocessing steps is to normalize the input data and make it more properly formatted with the pre-trained EfficientNetB0 model. The model can better process the input data and make accurate predictions by doing so.

4.3.4 Feature Extraction

EfficientNet architecture is pre-trained on the ImageNet dataset. The pre-trained weights are then used as a feature extractor, and the network's last layer is replaced with a new layer with 5 output nodes, each representing a different class.

The feature_extractor_layer is the pre-trained EfficientNetB0 model, which extracts features from the input images. When an image is passed through the feature_extractor_layer, it applies a series of convolutional filters to the image to extract features at different levels of abstraction. The first layers of the network extract low-level features, such as edges and corners, while the later layers extract more complex features, such as shapes and textures. The output of the feature_extractor_layer is a set of feature maps that represent the image at different levels of abstraction.

The x variable applies the feature_extractor_layer to the input images and then applies a global average pooling layer to reduce the dimensionality of the feature maps. Global average pooling is a technique that averages the values of each feature map across all spatial locations, resulting in a single value for each feature map. This reduces the number of parameters in the model and helps prevent overfitting.

Finally, the outputs variable defines the new layer with 5 output nodes and a softmax activation function. The softmax function normalizes the output of the layer so that the values sum to 1, and the output of each node represents the probability that the input image belongs to a particular class. During training, the weights of the new layer are updated to minimize the cross-entropy loss between the predicted probabilities and the true labels.

```
Model: "model_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_7 (InputLayer)        [(None, 224, 224, 3)]     0

 efficientnetb0 (Functional)  (None, 7, 7, 1280)       4049571

 global_average_pooling2d_3   (None, 1280)             0
 (GlobalAveragePooling2D)

 dense_2 (Dense)              (None, 5)                6405

=================================================================
Total params: 4,055,976
Trainable params: 6,405
Non-trainable params: 4,049,571
_________________________________________________________________
```

Figure 4.3.4: Model summary of the EfficientNetB0 Network after transfer learning with added layers

4.2.5 Training

The proposed network is trained with a multi-label classification method for grading Diabetic Retinopathy among five severity levels. In the multi-label classification, the prediction is not made on a single label; the target is set to a multi-label problem, i.e., if the target is a class 4, then it encompasses all the classes before it. The hyperparameters settings are kept constant for the multi-label classification method. The proposed network has more than 19 million total parameters. The network is trained with Adam's optimization function for a fixed schedule over 30 epochs with a learning rate 0.001. The summary of the settings of the training hyperparameters is given in Table.

Hyper parameters	Values
Loss Function	Categorical Loss
Optimizer	Adam
Learning Rate	0.001
Batch Size	32
Epoch	30

Table 4.3.5: The summary of the settings of our training hyperparameters.

4.3.6 Fine-Tuning

First unfreezing the top 10 layers of the feature_extractor_layer and freezing all other layers is done. This is known as fine-tuning, a technique used to improve the performance of a pre-trained model on a specific task by adjusting the weights of the pre-trained model.

By unfreezing the top 10 layers, the weights of these layers are allowed to be updated during training. By fine-tuning these layers, we can adjust the pre-trained features to better fit the specific task we are working on.

Finally, the model is recompiled with a lower learning rate (lr) 0.0001. This is because fine-tuning can cause the weights of the pre-trained model to change significantly, and a lower learning rate can help prevent the model from overfitting to the new task.

```
Model: "model_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_7 (InputLayer)        [(None, 224, 224, 3)]     0

 efficientnetb0 (Functional)  (None, 7, 7, 1280)       4049571

 global_average_pooling2d_3   (None, 1280)             0
 (GlobalAveragePooling2D)

 dense_2 (Dense)             (None, 5)                 6405

=================================================================
Total params: 4,055,976
Trainable params: 899,637
Non-trainable params: 3,156,339
_________________________________________________________________
```

Figure 4.3.6: Model summary of the EfficientNetB0 Network after Unfreeze.

Then fine-tuning is performed for the pre-trained EfficientNetB0 model on the dataset for another 50 epochs. Additionally, the best weights during training are saved for future predictions.

4.3.7 Experimentation Results

4.3.7.1 Performance Evaluation on Severity Grading

The model evaluation was performed for the multi-label method used in this study by evaluating the accuracy, precision, recall, f1-score, and support. The evaluation is done after training the model and also after fine-tuning them. The loss and accuracy curves were plotted to keep track of the model's performance concerning the number of epochs.

First, the saved weights of the best-trained model are loaded. Then, extraction of the true classes of the test set and the class indices of the training set is done. The class indices are inverted to be mapped to class names. Next, predictions for the test set are generated using the predict () method of the model and the argmax () method to find the class index with the highest probability for each image.

Finally, calculation of the model's accuracy on the test dataset is recorded, along with its classification report.

Model Accuracy: 86.31 %

Class	Precision	Recall	F1-score	Support
Mild	0.86	0.81	0.83	222
Moderate	0.97	0.99	0.99	201
No_DR	0.99	0.98	0.98	361
Proliferate_DR	0.72	0.74	0.74	295
Severe	0.79	0.80	0.80	309
Macro Average	0.87	0.86	0.86	1388
Weighted Average	0.86	0.86	0.86	1388

Table 4.3.7.1.1: Classification Report after Training.

Model Accuracy: 92.65 %

Class	Precision	Recall	F1-score	Support
Mild	0.90	0.95	0.93	222
Moderate	0.98	0.97	0.98	201
No_DR	0.98	0.99	0.99	361
Proliferate_DR	0.91	0.82	0.86	295
Severe	0.86	0.91	0.91	309
Macro Average	0.93	0.93	0.93	1388
Weighted Average	0.93	0.93	0.93	1388

Table 4.3.7.1.2: Classification Report After Fine-Tuning

4.3.7.2 Confusion Matrix

A confusion matrix describes the performance of a classification model over a validation set by comparing the true labels with the predicted labels. It shows the number of true positives, true negatives, false positives, and false negatives. Each element of the confusion matrix compares the true label and the predicted label for each image in the validation set. After training, the model had the highest accuracy and precision in the No DR class, correctly predicting 355 cases and only misclassifying 6 cases as Moderate DR. Also, the model also performed well in the Severe DR class, correctly predicting 252 cases. However, the model had some difficulty in the Mild DR and especially in Proliferate DR classes, with a relatively high number of false positives and false negatives.

After fine-tuning, the model's performance improved in most classes, particularly in the Mild DR and Proliferate DR classes, where the number of false positives and false negatives decreased from 107 to 42, resulting in higher accuracy and precision in these classes. The fine-tuning process has improved the model's performance, particularly in the Mild DR and Proliferate DR classes. The detailed Confusion Matrix results are explained in the Table below:

True Label/ Predicted Label	Mild	Moderate	No DR	Proliferate DR	Severe
Mild	179	0	0	32	11
Moderate	0	198	3	0	0
No_DR	0	6	355	0	0
Proliferate_DR	21	0	0	218	56
Severe	8	0	0	53	248

Table 4.3.7.2.1: Confusion Matrix after Training.

True Label/ Predicted Label	Mild	Moderate	No DR	Proliferate DR	Severe
Mild	212	0	0	6	4
Moderate	0	195	6	0	0
No_DR	0	3	358	0	0
Proliferate_DR	14	0	0	241	40
Severe	10	0	0	19	280

Table 4.3.7.2.2: Confusion Matrix after Fine-Tuning.

4.3.7.3 Loss and Accuracy Analysis

It is observed that EfficientNetb0 model after being used as feature extractor has training loss inversely proportional to the training accuracy. Whereas for the validation set, the validation loss drops initially and then remains almost constant. Although the validation accuracy varies over time but maintaining above 75% mark.

After fine tuning is done by unfreezing the last 10 layers, the training accuracy reaches above 95% and then remains constant irrespective of training loss. And for validation set, the accuracy is noted to be almost constant but the validation loss increases over time.

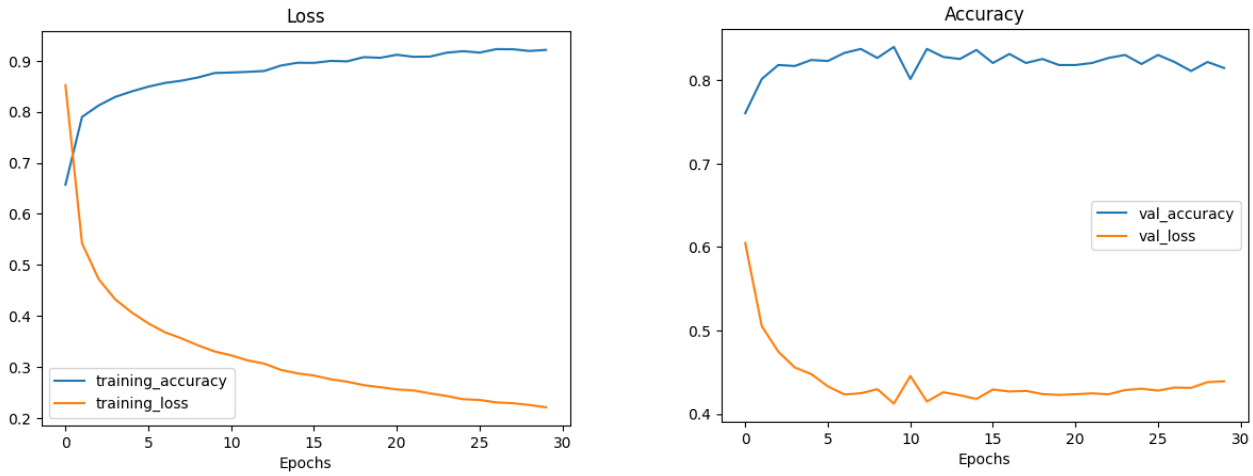

Figure 4.3.7.3.1: Accuracy and Loss plots after training.

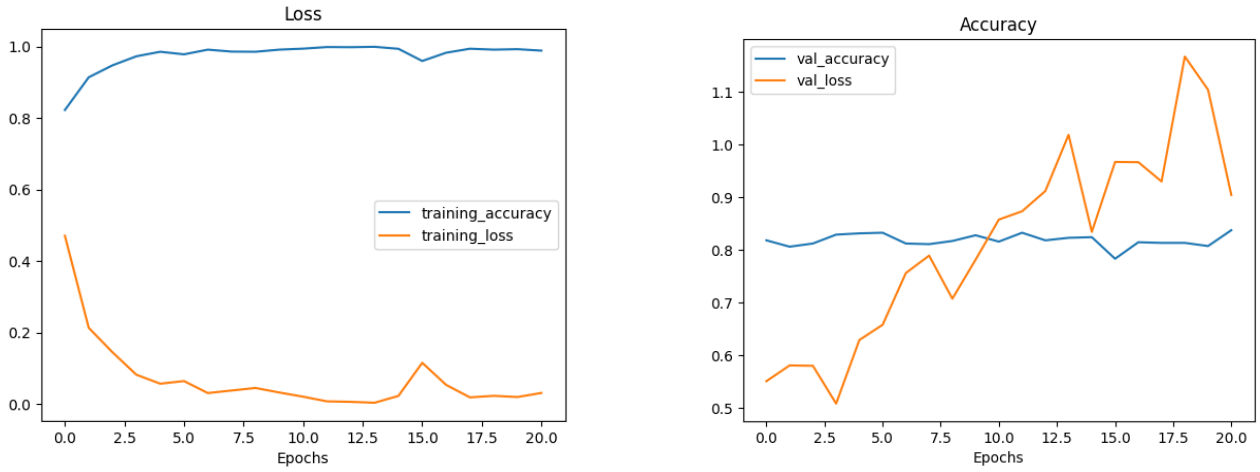

Figure 4.3.7.3.2: Accuracy and Loss plots after fine-tuning

5.COMPARSION

5.COMPARSION

5.1 Comparison of Performance

The performance of various previously mentioned transfer learning models for neovascularization detection and categorization in fundus images is compared based on different parameters. The results obtained demonstrate the performance of each model in terms of overall accuracy and the highest validation accuracy achieved during training. These metrics indicate how well the models can classify fundus images with neovascularization.

EfficientNetB3, EfficientNetB4, and EfficientNetB5 consistently achieved high accuracy scores, ranging from 88.40% to 90.85%. EfficientNetB3 has the highest validation accuracy of 82.67%, followed closely by EfficientNetB4 and EfficientNetB7.

MobileNetV3Large and ConvNext Large also perform well, with accuracy scores of 86.89% and 88.18%, respectively. They both achieve high validation accuracy as well, with MobileNetV3Large reaching 86.16% and ConvNext Large achieving 86.04%.

While EfficientNetB6 and EfficientNetB0 achieve relatively high accuracy scores, their highest validation accuracy is lower compared to other models. This indicates a potential for overfitting during training, where the models may perform well on the training set but do not generalize as effectively to new data.

Overall, the EfficientNet series, MobileNetV3Large, and ConvNext Large demonstrate strong performance in neovascularization detection, with high accuracy and validation accuracy scores. These models can be considered as top choices for further evaluation and deployment in clinical settings.

The following table summarizes the results:

	FEATURE EXTRACTION		**FINE TUNING**	
Model Name	Model Accuracy	Highest VA	Model Accuracy	Highest VA
EfficientNetB7	**86.67%**	84.36%	**92.51%**	84.72%
EfficientNetB6	**85.52%**	84.84%	**92.29%**	83.27%
EfficientNetB5	**86.60%**	85.08%	**88.04%**	85.08%

EfficientNetB4	**85.66%**	84.48%	**90.85%**	84.96%
EfficientNetB3	**88.40%**	81.71%	**90.49%**	82.67%
EfficientNetB2	**86.60%**	81.59%	**90.35%**	83.51%
EfficientNetB1	**84.94%**	83.39%	**92.07%**	84.72%
EfficientNetB0	**86.96%**	83.63%	**92.51%**	84.00%
MobileNetV3 Large	**86.69%**	82.55%	**93.95%**	86.16%
MobileNetV3 Small	**83.57%**	83.39%	**91.21%**	84.84%
ConvNext Large	**88.18%**	83.05%	**94.31%**	86.04%
ConvNext Small	**84.37%**	81.55%	**90.56%**	84.47%
InceptionResnetV2	**78.03%**	78.28%	**83.21%**	84.02%
Xception	**82.01%**	79.45%	**87.41%**	83.05%
NasNet Large	**84.37%**	79.42%	**84.94%**	79.54%
DenseNet201	**83.09%**	81.73%	**85.11%**	81.97%
DenseNet169	**84.01%**	84.00%	**84.54%**	84.12%

Table 5.1: Various Models accuracy Observations.

5.2 Size vs Accuracy

The Graphs below provide a comparison of various feature extraction and fine-tuning models for a specific task, where the accuracy of the model is evaluated based on a given dataset. The feature extraction models are used to extract relevant features from the input data, while the fine-tuning models are used to further refine the features and improve the accuracy of the model.

In terms of feature extraction models, EfficientNetB3 and ConvNext Large have the highest accuracy at 88.40% and 88.18%, respectively. EfficientNetB0 has the lowest accuracy at 86.96%. ConvNext Large has the highest number of parameters at 196,238,021, followed by EfficientNetB7 at 64,110,492. MobileNetV3Small has the lowest number of parameters at 942,005.

In terms of fine-tuning models, ConvNext Large has the highest accuracy at 94.31%, followed by EfficientNetB7 at 92.51%. Densenet201 has the lowest accuracy at 85.11%. ConvNext Large has the highest number of parameters at 196,238,021, followed by NasNet Large at 84,936,983. MobileNetV3Small has the lowest number of parameters at 942,005.

The choice of the best model depends on the specific requirements of the task at hand, such as the available computational resources and the desired trade-off between accuracy and efficiency. EfficientNetB3 and ConvNext Large are the best performing feature extraction models, while ConvNext Large and EfficientNetB7 are the best performing fine-tuning models. MobileNetV3Small has the lowest number of parameters in both feature extraction and fine-tuning models, making it a good choice for tasks with limited computational resources.

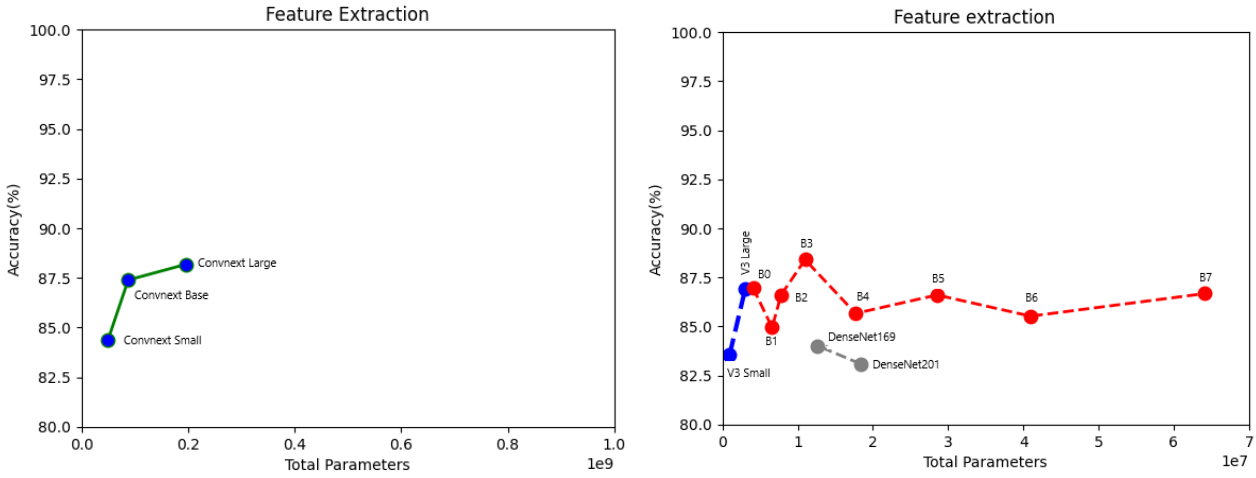

Figure 5.2.1: Accuracy Comparison after training

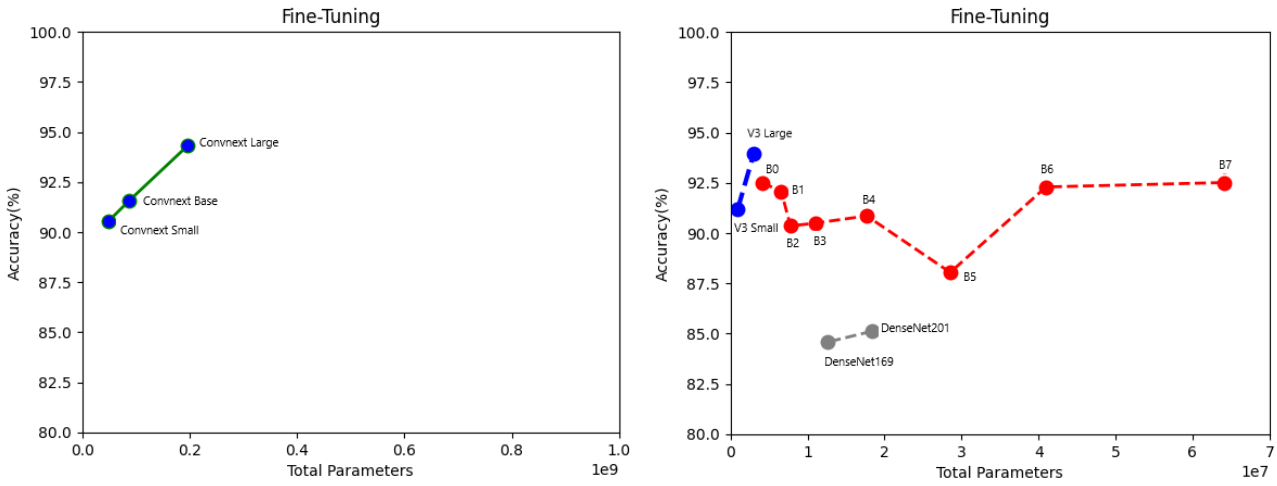

Figure 5.2.2: Accuracy Comparison after Fine-tuning

6.SCREENSHOTS

6.SCREENSHOTS

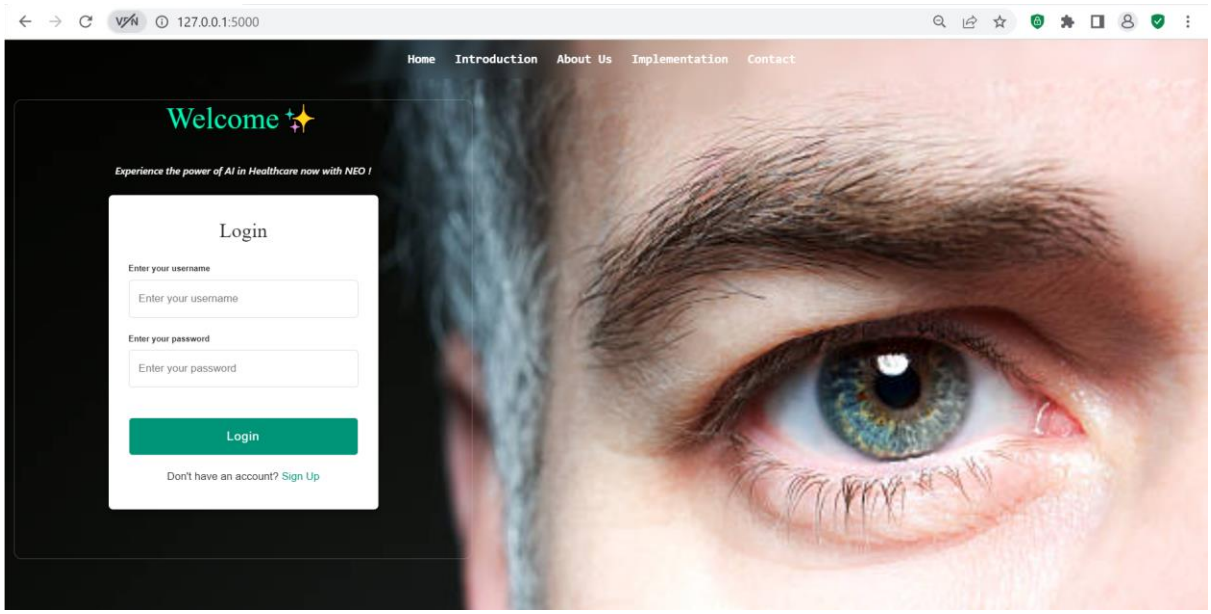

Figure 6.1: Home Page

Figure 6.2: Intro Page

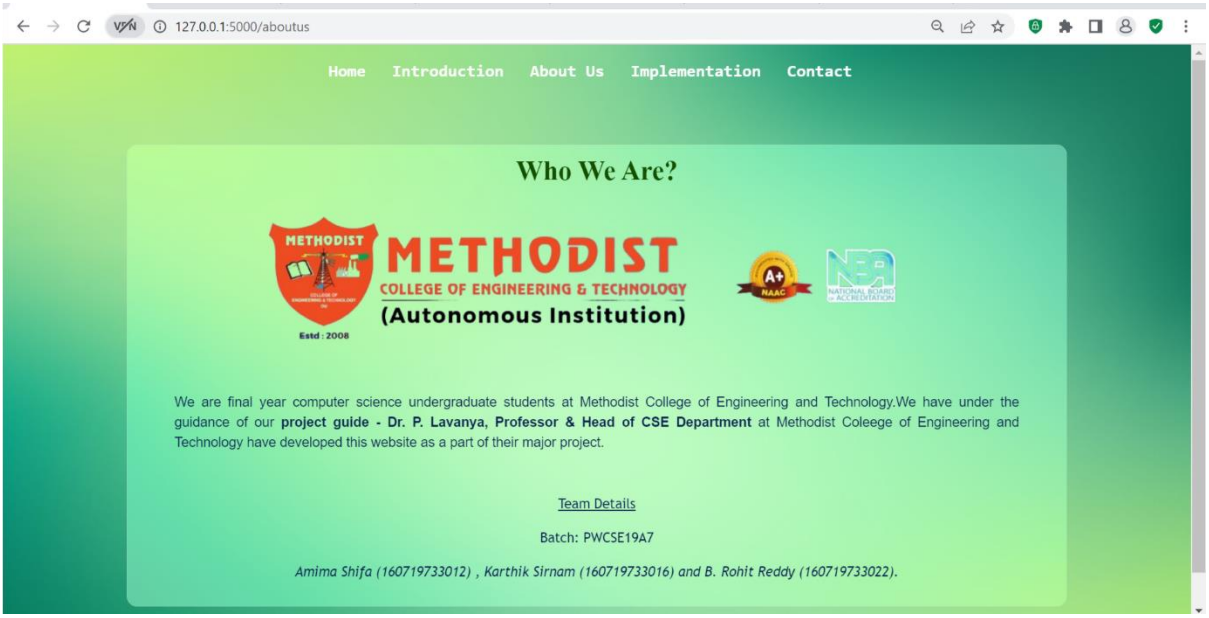

Figure 6.3: About Page

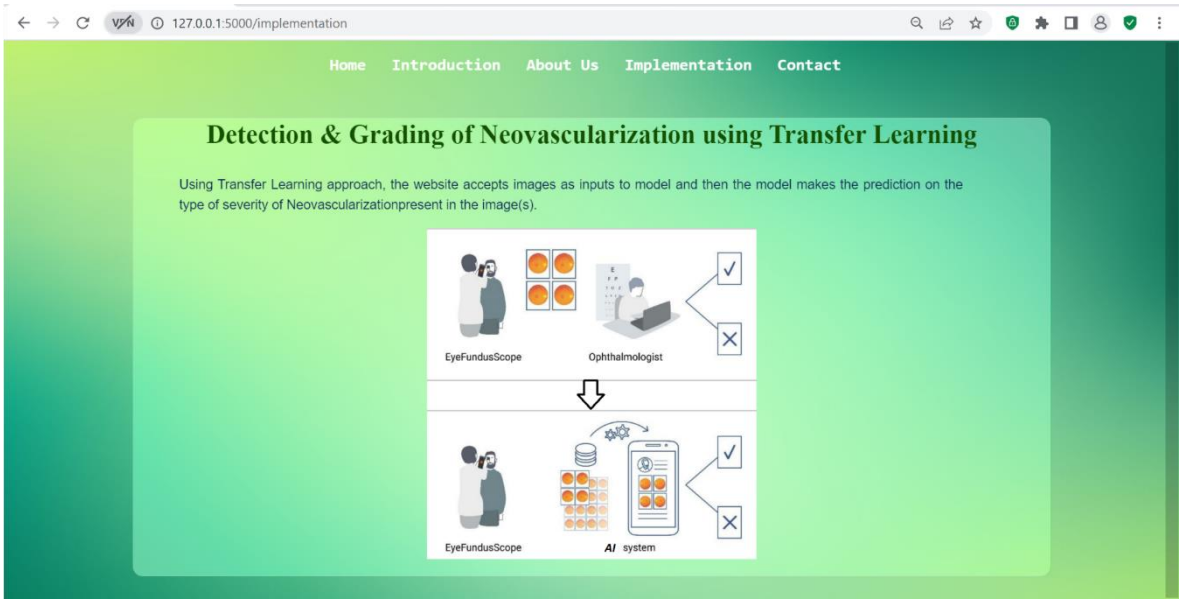

Figure 6.4: Implementation Page

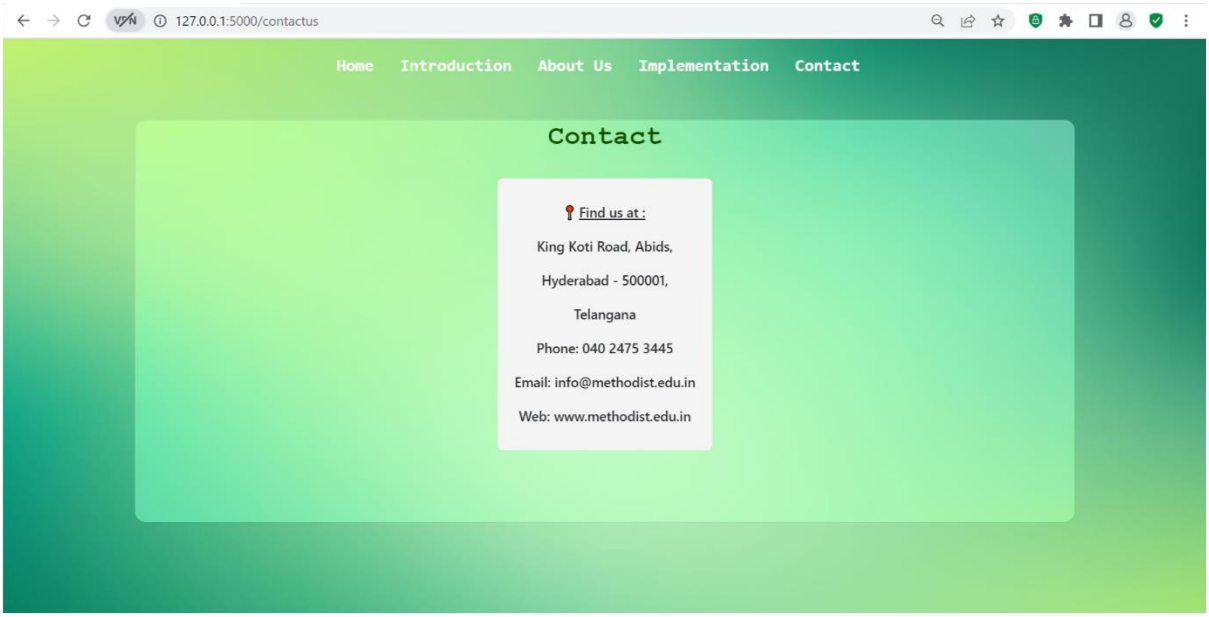

Figure 6.5: Contact Page

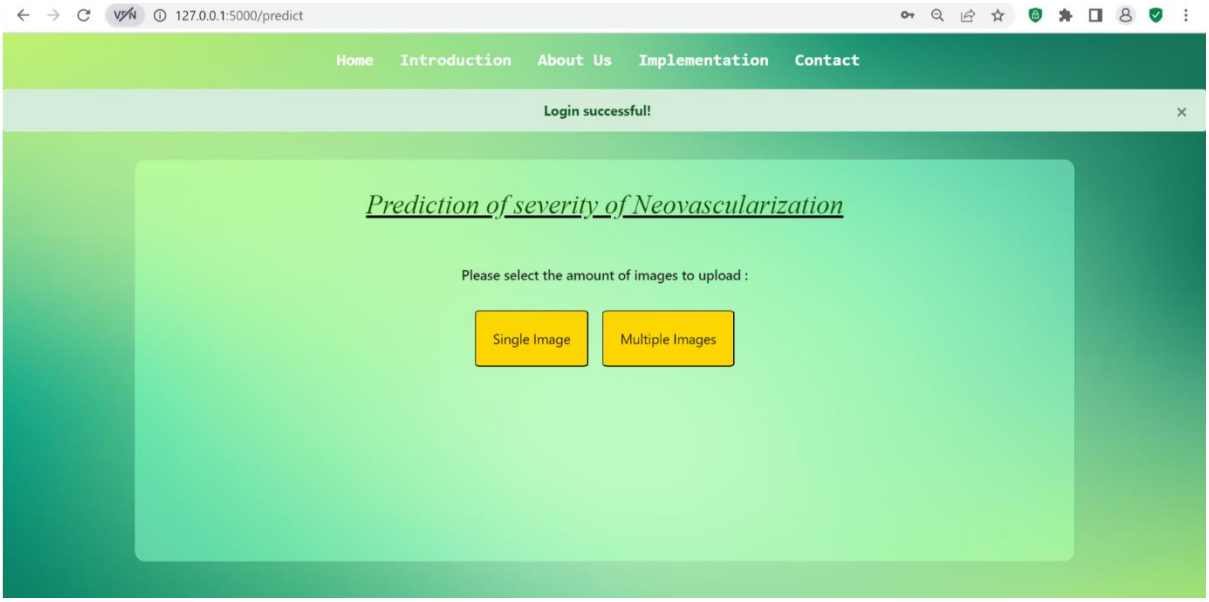

Figure 6.6: After Successful Login

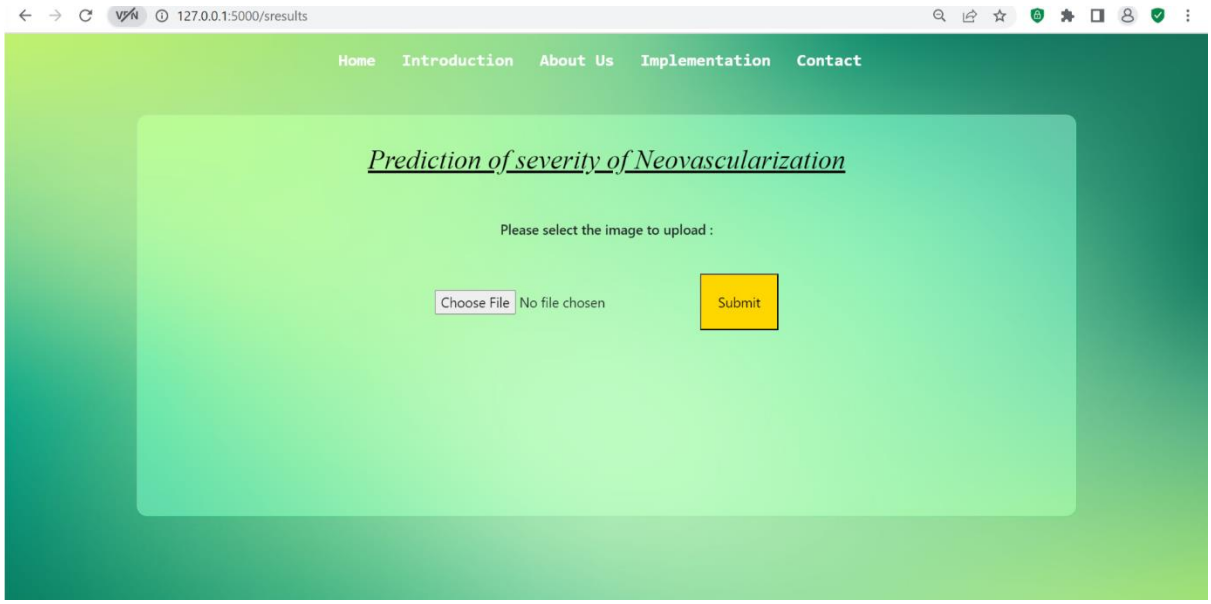

Figure 6.7: Single Image Upload Page

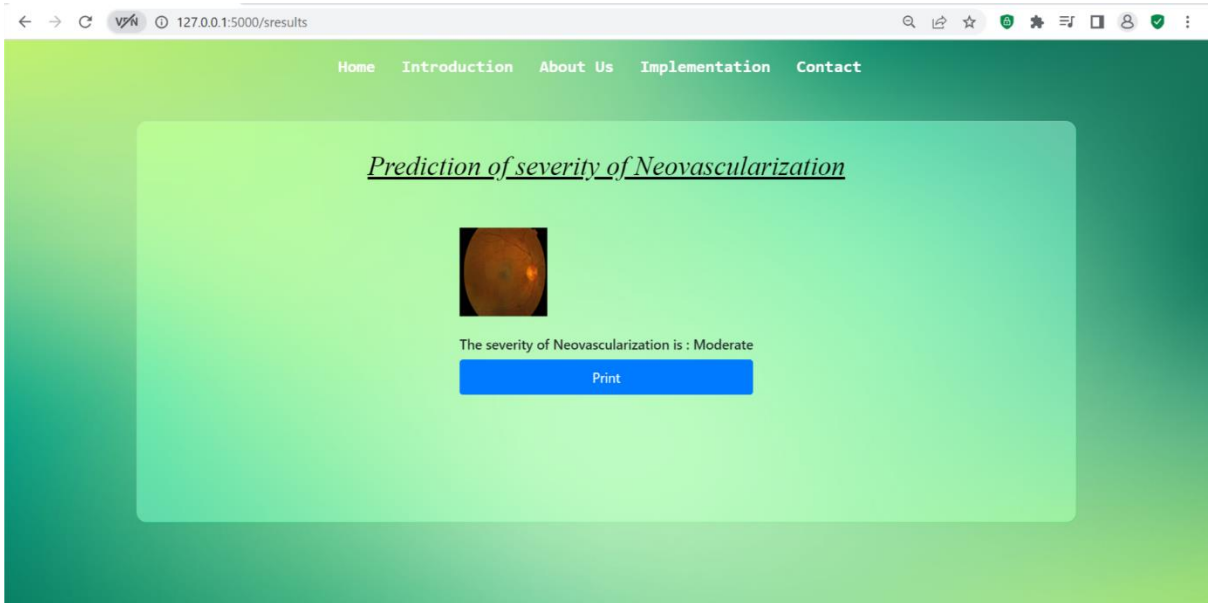

Figure 6.8: Results Page

CONCLUSION

A major complication of diabetes mellitus, diabetic retinopathy causes progressive damage to the retina and can potentially result in blindness. Its early detection and treatment are crucial to prevent its worsening and retinal damage. The interest in applying deep learning in detecting diabetic retinopathy has increased during the past year due to its state of art results. So, the goal of the work was to analyze various deep learning techniques, especially transfer learning to detect diabetic retinopathy along with its severity stage. Extensive Image processing has helped to highlight multiple features of the image.

We performed several modifications to multiple pre-trained models' network and employed preprocessing and fine-tuning to improve the network's performance. In our work, we have found that effective training can be done on data even with simple data preprocessing and selective data augmentation, well-tuned hyperparameters for the network.

Though CNN worked better over the years than traditional algorithms, it was found that when transfer learning algorithms were applied, like Convnext, Mobilenet, and EfficientNet, desired accuracy was achieved without much overfitting. The networks trained on APTOS 2019 dataset, have outperformed other state-of-the-art networks in early-stage detection, with Convnext Large attaining the best accuracy of 94.31% for the multi-label classification. The experimental results have demonstrated the effectiveness of our proposed method for clinical applications for Diabetic Retinopathy Detection.

FUTURE SCOPE

Researchers can investigate and develop novel transfer learning architectures specifically tailored for neovascularization detection in fundus images. These architectures can be designed to capture fine-grained features, hierarchical representations, and context-aware information relevant to neovascularization patterns. Enhancing the transferability and generalization capabilities of transfer learning models is essential for real-world applications. Even though deep learning has paved the door for more precise diagnosis and therapy, the performance and interpretability of the image still need to be improved. The parameters of the algorithms can be fine-tuned in the future to produce better results, and the model's accuracy can be increased by employing other effective optimization techniques. One main advantage of transfer learning is that it enables the model to assimilate new information without forgetting previously learned knowledge.

This capability is particularly valuable in scenarios where new data becomes available, allowing the model to continuously improve its performance. The accuracy is further expected to be improved when more image data is collected, thus boosting the system. These methods can be potentially applied for predicting some other diseases like brain tumor detection, which also requires clinicians to study the scanned report of the brain.

Ultimately, the project can be enhanced with a real-time user interface implementation to make it available in real time for the users.

REFERENCES

[1] “IDF Website.” [Online]. Available: https://idf.org/our-network/regionsmembers/western-pacific/members/108-malaysia.html.

[2] International Diabetes Federation. International diabetes federation diabetes atlas, ninth ed. https://www.diabetesatlas.org/en/.

[3] Alicia J. Jenkins, Mugdha V. Joglekar, Anandwardhan A. Hardikar, Anthony C. Keech, David N. O’Neal, S. Andrzej, Januszewski, Biomarkers in diabetic retinopathy, Rev. Diabet. Stud.: Reg. Dev. Stud. 12 (1–2) (2015) 159.

[4] Mohsen Janghorbani, Raymond B. Jones, Simon P. Allison, Incidence of and risk factors for proliferative retinopathy and its association with blindness among diabetes clinic attenders, Ophthalmic Epidemiol. 7 (4) (2000) 225–241.

[5] Jeng, C.J.; Hsieh, Y.T.; Yang, C.M.; Yang, C.H.; Lin, C.L.; Wang, I.J. Diabetic retinopathy in patients with dyslipidemia: Development and progression. Ophthalmol. Retin. 2018, 2, 38–45.

[6] Davidson, J.A.; Ciulla, T.A.; McGill, J.B.; Kles, K.A.; Anderson, P.W. How the diabetic eye loses vision. Endocrine 2007, 32, 107–116.

[7] Van Ginneken, B.; Schaefer-Prokop, C.M.; Prokop, M. Computer-aided diagnosis: How to move from the laboratory to the clinic. Radiology 2011, 261, 719–732.

[8] Lim, G.; Bellemo, V.; Xie, Y.; Lee, X.Q.; Yip, M.Y.T.; Ting, D.S.W. Different fundus imaging modalities and technical factors in AI screening for diabetic retinopathy: A review. Eye Vis. 2020, 7, 21.

[9] Deng Jia, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, Li Fei-Fei, Imagenet: a largescale hierarchical image database, in: 2009 IEEE Conference on Computer Vision and Pattern Recognition, Ieee, 2009, pp. 248–255.

[10] Wei Zhang, Jie Zhong, Shijun Yang, Zhentao Gao, Junjie Hu, Yuanyuan Chen, Yi Zhang, Automated identification and grading system of diabetic retinopathy using deep neural networks, Knowl. Base Syst. 175 (jul 2019) 12–25, https://doi. org/10.1016/j.knosys.2019.03.016. ISSN 9507051.

[11] Sheikh Muhammad Saiful Islam, Md Mahedi Hasan, Sohaib Abdullah, Deep Learning Based Early Detection and Grading of Diabetic Retinopathy Using Retinal Fundus Images, 2018 arXiv preprint arXiv:1812.10595.

[12] Tao Li, Yingqi Gao, Kai Wang, Song Guo, Hanruo Liu, Hong Kang, Diagnostic assessment of deep learning algorithms for diabetic retinopathy screening, Inf. Sci. 501 (oct 2019) 511–522, https://doi.org/10.1016/J.INS.2019.06.011. ISSN 0020–0255.

[13] M Usman Akram, Shehzad Khalid, Anam Tariq, M Younus Javed, Detection of neovascularization in retinal images using multivariate m-Mediods based classifier, PMID: 23916066 DOI: 10.1016/j.compmedimag.2013.06.008

[14] Jack Lee, Benny Chung Ying Zee, Qing Li, Detection of neovascularization based on fractal and texture analysis with interaction effects in diabetic retinopathy, PMID: 24358105 PMCID: PMC3864789 DOI: 10.1371/journal.pone.0075699

[15] Diego F. G. Coelho; Rangaraj M. Rangayyan; Vassil S. Dimitrov, Detection of neovascularization near the optic disk due to diabetic retinopathy, DOI: 10.1109/EUSIPCO.2016.7760607

[16] http://cdn.iiit.ac.in/cdn/cvit.iiit.ac.in/images/ConferencePapers/2017/janneovascularization.pdf

[17] Yung-Hui Li,1Nai-Ning Yeh,1Shih-Jen Chen,2and Yu-Chien Chung, Computer-Assisted Diagnosis for Diabetic Retinopathy Based on Fundus Images Using Deep Convolutional Neural Network, Volume 2019 | Article ID 6142839 | https://doi.org/10.1155/2019/6142839

[18] He Huang, He Ma, Wei Qian, Automatic Parallel Detection of Neovascularization from Retinal Images Using Ensemble of Extreme Learning Machine, PMID: 31946914 DOI:10.1109/EMBC.2019.8856403

[19] Yi-Peng Liu, Zhanqing Li, Cong Xu, Jing Li, Ronghua Liang, Referable diabetic retinopathy identification from eye fundus images with weighted path for convolutional neural network, PMID: 31606108 DOI: 10.1016/j.artmed.2019.07.002

[20] Muhammad Samer Sallam; Ani Liza Asnawi; Rashidah Funke Olanrewaju, Diabetic Retinopathy Grading Using ResNet Convolutional Neural Network, DOI: 10.1109/ICBDA50157.2020.9289822

[21] Saket S. Chaturvedi, Kajol Gupta, Vaishali Ninawe, Prakash S. Prasad, Automated Diabetic Retinopathy Grading using Deep Convolutional Neural Network,

https://doi.org/10.48550/arXiv.2004.06334

[22] Michael Chi Seng Tang,Soo Siang Teoh,,Haidi Ibrahim,Zunaina Embong, Neovascularization Detection and Localization in Fundus Images Using Deep Learning,https://doi.org/10.3390/s21165327

[23] Wejdan L Alyoubi, Maysoon F Abulkhair, Wafaa M Shalash, Diabetic Retinopathy Fundus Image Classification and Lesions Localization System Using Deep Learning, PMID: 34073541 PMCID: PMC8198489 DOI: 10.3390/s21113704

[24] Yasashvini R,Vergin Raja Sarobin M.,Rukmani Panjanathan,Graceline Jasmine S.,Jani Anbarasi L., Diabetic Retinopathy Classification Using CNN and Hybrid Deep Convolutional Neural Networks, DOI:10.3390/sym14091932

[25] Laith Alzubaidi, J. Santamaría, Mohamed Manoufali, Beadaa Mohammed, Mohammed A. Fadhel, Jinglan Zhang, Ali H.Al-Timemy, Omran Al-Shamma, Ye Duan, MedNet: Pre-trained Convolutional Neural Network Model for the Medical Imaging Tasks, https://doi.org/10.48550/arXiv.2110.06512

[26] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, Quoc V. Le, Hartwig Adam, Searching for MobileNetV3, https://doi.org/10.48550/arXiv.1905.02244

[27] “TensorFlow Website” [Online]. Available: https://www.tensorflow.org/api_docs/

python/tf/keras/applications/mobilenet

[28] Tan, M., & Le, Q. V. (2019). EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. arXiv preprint arXiv:1905.11946. https://arxiv.org/abs/1905.11946

[29] Wang, Y., Zhang, Y., Zhang, Y., & Zhang, J. (2020). A novel plant disease detection method based on improved YOLOv3 and EfficientNet. Computers and Electronics in Agriculture, 178, 105764.

[30] Zhang, X., Zhou, X., Lin, M., & Sun, J. (2021). Dynamic R-CNN: Towards high quality object detection via dynamically integrating contextual information. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 1381-1390).

RESOURCES