NITTE EDUCATION TRUST | N.M.A.M. INSTITUTE OF TECHNOLOGY
(An Autonomous Institution affiliated to Visvesvaraya Technological University, Belagavi)
Nitte – 574 110, Karnataka, India

DIABEYE - COMPREHENSIVE DEEP LEARNING FRAMEWORK FOR DIABETIC RETINOPATHY CLASSIFICATION

Project Report submitted by

Vaibhavi Shenoy B (4NM21CS198)

Shravya K (4NM21CS156)

Sanketha S Hegde (4NM21CS142)

Kavana H K (4NM22CS408)

Under the Guidance of

Dr. Asmita Poojari

Assistant Professor Gd-III

Department of Computer Science & Engineering

In partial fulfillment of the requirements for the award of
Bachelor of Engineering in Computer Science and Engineering

Department of Computer Science Engineering
NMAM Institute of Technology, Nitte - 574110
(An Autonomous Institution affiliated to VTU, Belagavi)

November 2024

NITTE EDUCATION TRUST | N.M.A.M. INSTITUTE OF TECHNOLOGY
(An Autonomous Institution affiliated to Visvesvaraya Technological University, Belagavi)
Nitte – 574 110, Karnataka, India

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

Certified that the project work entitled

"DIABEYE – COMPREHENSIVE DEEP LEARNING FRAMEWORK FOR DIABETIC RETINOPATHY CLASSIFICATION "

is a bonafide work carried out by

Vaibhavi Shenoy B (4NM21CS198)

Shravya K (4NM21CS156)

Sanketha S Hegde (4NM21CS142)

Kavana H K (4NM22CS408)

in partial fulfillment of the requirements for the award of

Bachelor of Engineering Degree in Computer Science and Engineering*prescribed by* ***Visvesvaraya Technological University, Belagavi*** *during the year* ***2024-2025****.*

It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the report deposited in the departmental library. The project report has been approved as it satisfies the academic requirements in respect of the project work prescribed for the Bachelor of Engineering Degree.

Signature of the Guide	***Signature of the HOD***	***Signature of the Principal***

Semester End Viva Voce Examination

Name of the Examiners	**Signature with Date**
1.	
2.	

ACKNOWLEDGEMENT

Any achievement, be it academic or other than that does not rely upon individual efforts of a person alone but that of the support, help and cooperation one receives by the intellectuals, elders as well as friends. Some personalities in their own field have supported me to successfully complete this project work. I am deeply grateful to all those who have provided invaluable support and guidance throughout the journey of this project.

First and foremost, I would like to thank **Dr Niranjan N Chiplunkar**, Principal, NMAMIT, Nitte, for providing the opportunity to carry out the project work as a part of the academics.

I would like to thank **Dr Jyothi Shetty**, Head of the Department, Computer Science & Engineering, NMAMIT, Nitte, for her valuable suggestions and expert advice.

I deeply express my sincere gratitude to my guide **Dr Asmita Poojari** Assistant Professor, Department of CSE, NMAMIT, Nitte, for her able guidance, regular source of encouragement and assistance throughout this Project work.

I also extend my thanks to Panel Member, **Dr Shabari Shedthi B** Associate Professor, for her guidance and innovative ideas to be applied for the project.

I thank my Parents, and all the faculty members of Department of Computer Science & Engineering for their constant support and encouragement.

Last, but not the least, I would like to thank my team and friends who helped me throughout the development of the project.

ABSTRACT

In this world today, eye diseases have become one of the severe public health problems that affect millions of people globally. Changes in lifestyles and health patterns and behavior have seen many different conditions develop, including eye-related conditions. Diabetic Retinopathy(DR) is a complication of diabetes that affects vision by damaging the blood vessels at the back of the eye, which is known as the retina due to high blood sugar levels. This could result in swelling, leakage, or blockage of these vessels and subsequently cause change in vision and, at the advanced stages, blindness. The early detection of DR can salvage much of the lost vision.

DR progresses through several stages of increasing severity: No DR, Mild Non-Proliferative DR (NPDR), Moderate NPDR, Severe NPDR, and Proliferative DR. Each stage reflects a greater degree of retinal vessel damage. Recently, deep learning—a branch of machine learning inspired by neural networks in the human brain—has shown promise in analyzing complex patterns in large datasets, making it particularly effective for image classification tasks like DR detection.

Our project utilizes deep learning techniques to identify and classify DR stages. We aim to analyze retinal images from diabetic patients in coastal Karnataka, creating a balanced dataset for classification across four levels of DR severity. By training our real-time dataset on pre-trained deep learning models, we aim to determine which models can most accurately identify and classify DR stages in these images.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Diabetic retinopathy (DR) is the major ocular complication of diabetes mellitus, a metabolic disorder that is characterized by chronic hyperglycemia, high sugar levels in blood, caused by insulin secretion or insulin action. DR is common causes of preventable vision loss and blindness worldwide, especially among working-age adults. It is due to chronic diabetes that the condition emerges, though Type 1 and Type 2 diabetes both share the manifestation of DR. However, pathways through which and the time course for the development of DR may differ between Type 1 and Type 2 diabetes. This disease progresses over time and is particularly common among individuals with diabetes. It results from prolonged exposure to elevated blood sugar levels, which harm the tiny blood vessels in the retina. This condition, recognized as both a microvascular and neurodegenerative disorder, is considered a major cause of blindness globally, particularly affecting adults of working age. DR progresses insidiously. Patients having diabetes must receive regular retinal screenings. Diabetic retinopathy involves complex pathophysiology through cascading biochemical and physiological alterations. For a long time, high blood glucose causes damage to the capillaries of the retina, weakening, leaking, or obstructing them. Further, other factors like hypertension, dyslipidemia, and inflammation contribute to further complications in the disease process. This early response is protective but leads eventually to the formation of abnormal, weak vessels that bleed easily, thereby causing more severe complications like proliferative DR.

In the non-diabetic retinopathy stage, the retina seems normal, meaning that one has to continue monitoring and diabetes management. The next stage is Microaneurysms and is classified as mild non-proliferative diabetic retinopathy. In this stage, the tiny blood vessels in the retina can swell slightly and may leak small amounts of fluid, though this does not impact vision. This is the case with moderate non-proliferative diabetic retinopathy, where damage to blood vessels increases and may include symptoms like blurred vision, but some patients do not have such symptoms. Severe NPDR is that stage where the blood vessels get blocked, which leads to severe deprivation of blood flow and potential complications such as new abnormal growth of blood vessels. Proliferative DR is the advanced type, which is considered by the growth of fragile blood vessels that easily bleed leading to severe vision loss and possible retinal detachment.

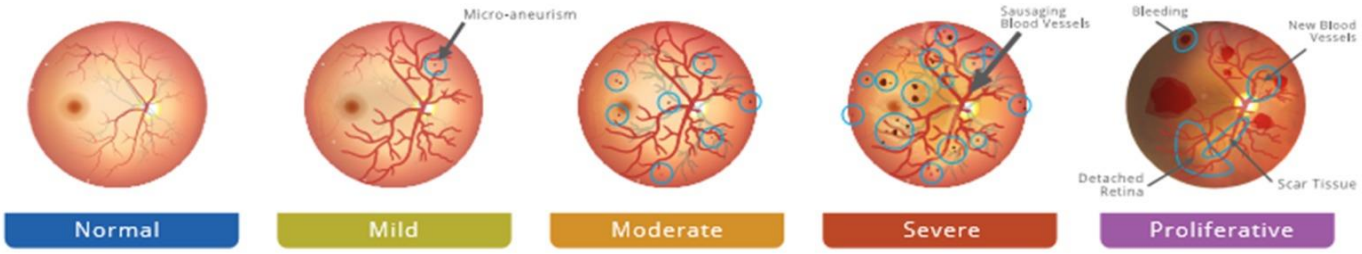


Figure 1: Different types of diabetic retinopathy

Figure 1 illustrates the various stages of diabetic retinopathy, categorized into the following levels: no diabetic retinopathy, mild, moderate, severe, and proliferative stages. In patients whose blood sugar is not very well-controlled, the disease can advance rapidly to a mild NPDR, during which the retinal blood vessels form small balloon-like swellings known as microaneurysms, sometimes also leaking a small amount of fluid into the retina. Fluid deposition can cause some swelling in the retinal tissue, a subtle effect that blurs the view but often does not trigger any noticeable discomfort at this stage. More vessels are engaged and more are blocked or closed; this impairs blood flow and oxygen delivery to the retina, allowing more fluid to accumulate and making the retinal swelling thicker. As DR advances to moderate NPDR, more vessels are affected by blockages or closures that decrease blood flow and oxygen delivery to the retina, which causes greater fluid accumulation and thicker retinal swelling. Symptoms may start to appear, such as slight blurring, colour changes, or small floating spots in vision, but these are usually mild.

In the final stage, the severe NPDR is characterized by extensive retinal damage. This action, instead of helping the deprived tissues by bringing in oxygen, causes the formation of weakened, abnormal vessels easily predisposed to leakage. This is a stage requiring very close follow-up with an eye specialist because this immediately results in proliferative diabetic retinopathy-the most advanced and threatening stage to vision. In PDR, new blood vessels begin proliferating or growing in uncontrolled fashion over the retina surface and into the vitreous humor-clear, jelly-like substance in the middle of the eye. These nerves are very strong and prone to tearing, causing hemorrhages in the eye that severely impair vision and can sometimes cause sudden loss of vision.

With longer durations, the more time a patient has been diabetic, the higher the chance for DR. The second important risk factor is poor glycemic control. In some parts of the world, nearly 40 percent of people diagnosed with diabetes can expect to experience some type of DR over their lifetime, and for those 50 years and older, it is higher.

However, in disease-prone areas, more patients tend to have more advanced DR conditions such as PDR leading to severe loss of vision due to delay in call.

Although there is no cure, several treatments may help slow the disease progression and reduce vision loss. The prevention of DR is management of blood glucose, blood pressure, and cholesterol-are three main factors that directly affect blood vessels. Laser photocoagulation treatment is usually applied for those showing signs of retinal damage, particularly NPDR and early PDR. The treatment uses laser energy to create small burns on leaking or abnormal blood vessels to seal them off from leaking. Another common treatment involves anti-VEGF injections that prevents the formation of abnormal blood vessels. These injections is for macular edema, retinal swelling, and early PDR to limit fluid buildup and reduce leakage. When bleeding is extreme, or if the retina is detached, vitrectomy surgery becomes necessary blood-filled vitreous is evacuated, and it is replaced by a clear solution, in addition, scar tissue can be removed from the retina.

Due to advancements in early detection and treatment, recent years have improved the prospect for patients diagnosed with diabetic retinopathy. Some of the new technologies, including artificial intelligence-based diagnostic tools and telemedicine platforms, enable greater access to screenings for people in underserved communities. It also speeds up faster diagnosis. More hope remains in research for gene therapy, neuroprotective agents, and novel pharmacological treatment methods. Preventive care for diabetic retinopathy includes optimal blood glucose, blood pressure, and cholesterol levels while conducting routine eye examinations. This proactive approach is crucial in preventing a person from contracting diabetic retinopathy.

Deep Learning is the core technology that the dynamically moving world of artificial intelligence changes the ways machines understand, learn, and interact with complex data. Deep Learning AI essentially tries to replicate complicated neural networks found within the human brain, which enables computers to automatically discover patterns and make decisions through vast levels of unstructured data. This transformative field has seen its highest pace of breakthroughs begin in computer vision and natural language processing and all the way into applications in diagnostics of healthcare and in autonomous driving. In this introduction to Deep Learning, we uncover its foundational principles, applications, and the underlying mechanisms that empower machines to achieve human-like cognitive abilities. This article forms an opening toward a wider understanding of how Deep Learning is transforming the operational paradigm of

industries and pushing the limits of what is possible in AI, paving the way for a world in which intelligent systems can autonomously perceive, comprehend, and innovate.
Deep learning refers to a technique of machine learning that mimics how humans acquire specific kinds of knowledge. It falls under the broader umbrella of artificial intelligence. These models of deep learning can be trained to classify problems and even to identify patterns in photos and text and audio types and so forth. Additionally, it has also been used in automating tasks requiring human intelligence such as describing images or transcribing audio files.

Deep learning is the backbone of data science, including statistical and predictive modeling. It is quite useful for task of collecting, analysis, and interpretation of large data sets. It helps in easier and faster collection and processing of data and deep learning features neural networks constructed from multiple layers of software nodes that work together. Deep learning models are developed using various datasets and neural network structures. A fully connected deep neural network consists of an input layer followed by one or more layers that are sequentially connected. The different layers apply multiple nonlinear transformations to the input data so that the network can learn complex representations of the input data. Deep learning is becoming increasingly popular in machine learning due to its incredible success in many applications. Deep learning AI can be applied for supervised, unsupervised as well as reinforcement machine learning. it employs numerous ways to process all these.

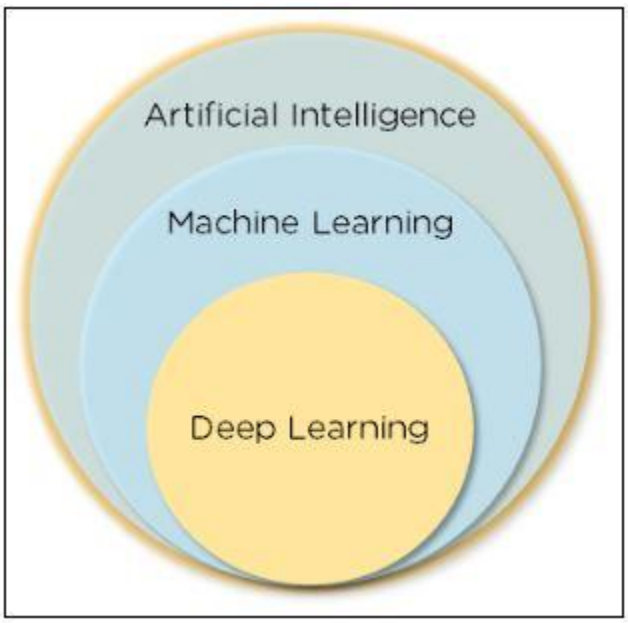


Figure 2: An Introduction to Deep Learning

Deep learning helps the computer to learn by example. To better understand deep learning, imagine the first word a child says is dog. The toddler will differentiate between a dog and a non-dog through the toddler's act of pointing to various objects and uttering the word dog. The parent says, "Yes, that is a dog," or, "No, that is not a dog." The more the toddler continues to point to objects, the more he becomes aware of what all dogs possess. What the toddler does, unwittingly, is clarify a complex abstraction: the concept of dog. They achieve this by creating a hierarchy where each layer of abstraction is developed based on knowledge we get from the previous layer within the hierarchy.

Deep learning requires a significant amount of computing power. Because they can process massive amounts of data in multiple cores with lots of memory, high speed GPUs are perfect for this use. This is another situation where distributed cloud computing could be useful. In order to train deep algorithms utilizing deep learning, this level of computing is required. However, overseeing a number of GPUs on-premises can be resource-intensive internally and very costly to scale. Typically, when a deep-learning software requirement would have to address in one of these three learning frameworks: JAX, Py Torch or TensorFlow. CNN can recognize features and patterns in images and videos, allowing object detection, image recognition, pattern recognition, and even face recognition. These networks apply the principles from linear algebra more specifically, matrix multiplication, to recognize a pattern inside an image.

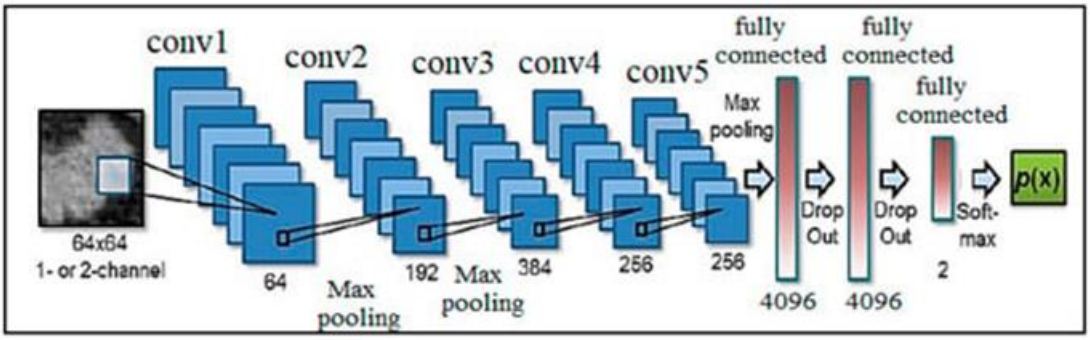


Figure 3: Basic CNN Architecture

CNNs are a special type of neural network. A neural network is made up of layers of nodes, consisting of an input layer, one or more hidden layers, and an output layer. A node is simply a point where two connections join; it also has associated with it a weight and threshold. If the output of any individual node is above the specified threshold value, then the node is activated, and this generates data to the next layer of the network. Otherwise, nothing passes to the next layer of the network.

A CNN contains at least three kinds of layers: a convolutional layer, pooling layer, and fully connected (FC) layer. In case of complex use, CNN may be up to thousands of layers that are stacked by incorporating the outcome of one layer into the other. Detailed patterns can be found through "convolution," or working and reworking the original input. As the layers go by, the CNN increases in its complexity while identifying greater portions of the image. Much earlier layers of such CNN, however, focus more on basic features, like colors and edges. The information of the image goes through different layers of a CNN to notice larger elements or shapes of an object until it finally identifies the intended object. This is what differentiates CNNs from other neural networks: performance with image, speech, and audio inputs. Pre- CNNs would use manual and time-consuming feature extraction methods to identify objects in images. However, CNNs now offer a much more scalable way to classify images and recognize objects and handle high-dimensional data. And CNNs can exchange data between layers for delivering more efficient data processing.

Aside from these, CNNs have a lot of drawbacks, including high computational costs that require many GPUs and cost money and time. It also requires thorough testing of settings, hyperparameters, and configurations, as well as highly skilled professionals with cross-domain experience. In fact, computer vision makes excellent use of convolution neural networks, one of the major types of deep neural networks. It is applicable to picture categorization, image recognition, and the collecting of a set of input images. To create more features from shapes and angles, CNNs use simpler features like edges and curves. CNN is an extremely powerful computer that collects and learns both local and global data. Convolution is made of CNN's hidden layers. The fundamental framework of CNN is provided in Figure 3.

1.1 Motivation

Diabetic Retinopathy is one of the leading causes of blindness in the elderly people in the country. It has various stages progressing through different levels of severity. There has been a lot of researches in this particular area. Traditional methods for detection of DR are as follows: First the patient takes doctor's appointment, their eyes are scanned and then evaluated, the patient then have to schedule a follow up evaluation after which the doctor review their result and consult corresponding treatments. This procedure almost takes 7 – 14 days. This process is both time-intensive and laborious.

Deep learning models have demonstrated outstanding capabilities in feature extraction, classification, and prediction tasks. These models can identify various features within retinal fundus images, leveraging training on collected datasets. The extracted features are analyzed and as a result the images can be categorized into which DR stage it belongs to.

CHAPTER 2

LITERATURE REVIEW

Table 1: Literature survey

SL No.	Paper Title	Methodology(Algorithms)	Results or inference
Paper 1	Diabetic Retinopathy Detection using Deep Learning.	This paper classifies using the deep learning architecture called "DenseNet 121Architecture". DenseNet architecture is an advance version of ResNet architecture. This architecture do not summation or add the outcome of the features of the layer with the incoming features but concatenate them.	The two architectures are VGG16 and DenseNet121 and the accuracies are 0.7326 and 0.9611 respectively. The QWK helped us to give the confidence of accuracy which we got from DenseNet architecture.
Paper 2	Diabetic retinopathy detection and classification using capsule networks.	Capsule networks (CapsNet) are proficient in retrieving the spatial information as well as more significant features without losing any information. Capsules are designed with a set of neurons that works totally with vectors. The neurons of a capsule are designed to work individually for different properties of an object, like position, size, and hue, etc. With this property, capsules are to study the characteristics of an image along with its deformations and viewing conditions.	The proposed model got 97.98% accuracy in identifying the problem. The proposed capsule network is trained with only Messidor dataset in which diabetic retinopathy is identified in four stage grades 0 to 3.
Paper 3	Automated Diabetic Retinopathy Detection and classification using ImageNet Convolution Neural Network using Fundus Images.	ImageNet CNN was used in this study to detect and classify images of the retina. The whole work consists of developing a model that classifies the fundus images into four classifications like normal, mild, severe and proliferative.	The proposed model achieved training-accuracy of 98.8% and validation-accuracy of 98.5% .

Paper 4	Diabetic Retinopathy (DR) Severity Level Classification Using Multimodel Convolutional Neural Networks.	Fused Ensemble Convolutional Inception Network is a new technique which we use to classify DR images. The number of inception blocks and nodes for each of the CNNs are generated randomly.To investigate the proposed method experimentally, two ensembles of multi-inception models comprised of 5 and 10 individual Inceptions were generated.	The accuracy results for the proposed method are 91.3 % and 93.2% for 5 nets and 10 nets respectively. From these results,all pre-trained models are for the classification of DR fundus Images.
Paper 5	Deep learning for diabetic retinopathy detection and classification based on fundus images.	The algorithm discussed in the document focuses on using deep learning and integrated gradients to assist in grading for diabetic retinopathy. It involves ensemble methods such as majority voting, averaging, bagging, stacking, and boosting to combine the outputs of multiple models for improved predictions. Majority voting and averaging are used for classification and regression problems, respectively. Bagging trains models on subsets of the dataset and combines their outputs. Stacking involves training a meta-model on individual model predictions. Boosting incorporates models trained multiple times based on performance errors. The algorithm aims to enhance the accuracy of diabetic retinopathy grading through these ensemble methods.	Diabetic retinopathy detection, such as identifying exudates, microaneurysms, and hemorrhages. The algorithm achieved accuracy rates ranging from 77.9% to 100% for different lesion types across different datasets. Ensemble methods like CNNs and decision trees were used to achieve high accuracy in classifying referable and vision-threatening diabetic retinopathy. The study also demonstrated the effectiveness of deep learning models like Inception V3 and custom CNNs in accurately detecting diabetic retinopathy-related lesions with accuracy rates exceeding 90% in some cases.
Paper 6	A Deep Learning Approach for Diabetic Retinopathy detection using Transfer Learning.	This paper focuses on implementing transfer learning approach. Transfer learning makes use of the knowledge gained while solving one problem and applying it to a different but related problemTthe methodology taken into consideration is Feature based TL. The models used in our research are pretrained on ImageNet Dataset.	In this paper, we have described the efficiency of transfer learning and the viability of using pretrained models such as SEResNeXt32x4d and EfficientNetb3. Our proposed methodology was able to achieve higher accuracy than conventional CNN.

Paper 7	Diabetic Retinopathy Classification using a Combination of EfficientNets.	Models are trained using one cycle method. Pretrained models on ImageNet are used for better and faster convergence. During training, the body is freezed first and the classification head is trained for 1 epoch. All layers are then unfreezed and the whole model trained for an average of 20 epochs. Different learning rates (Lr) for different models for each phase using the methods from are used. Adam optimization is used as an optimizer for all the models. Different architectures from the EfficientNet family (B1 to B6) are trained on different resolutions. All models ar trained on Tesla K80 GPU.	Ensemble of EfficientNet B1, B2, B3 and B5 boost the performance on test data as it seems to average out the errors of individual models and thus proper model combinations of such compatible models will result in better final predictions.
Paper 8	Diabetic Retinopathy Classification Using Deep Learning .	For feature extraction, we employed a Convolutional Neural Network (CNN) with ResNet152 pre-training. The entirely linked layers of the ResNet152 network were removed and replaced with one fully connected layer, each with 1024 units, before a final softmax layer for classification was added. We trained the model across 30 epochs using the Adam optimizer, with a batch size of 32 and a learning rate of 0.001.	Deep learning models like Xception, VGG, ResNet, and DenseNet have the potential to revolutionize the diagnosis and treatment of diabetic retinopathy due to their high accuracy rates. Xception : 75.46,VGG16:96.79 ResNet152:95.86, DenseNet201 : 86.32.
Paper 9	A deep learning system for detecting diabetic retinopathy across the disease spectrum.	The DeepDR system consists of three sub-networks: an image quality assessment sub-network, a lesion-aware sub-network, and a DR grading sub-network. It utilizes a Mask-RCNN architecture for lesion detection and segmentation, including microaneurysms, cotton-wool spots, hemorrhages, and hard exudates. The system employs hard parameter sharing and pre-trained DR grading to enhance performance. Annotations by ophthalmologists are validated based on IoU, and disagreements are resolved by senior supervisors. The system grades DR severity into five levels and generates clinical reports automatically. Performance metrics include AUC, sensitivity, specificity, IoU, and F-score for different types of retinal lesions.	Accurately detecting retinal lesions, leading to improved diagnostic accuracy for primary healthcare workers. The system demonstrated high AUC values, sensitivity, and specificity for different types of retinal lesions, including hard exudates and hemorrhages. Real-time quality feedback significantly enhanced DR diagnosis, particularly for referable DR cases. The system's performance was validated using statistical analyses with Python software and third-party packages for image processing and neural network computing, showcasing its effectiveness in improving DR screening accuracy.

Paper 10	Automatic Detection and Classification of Diabetic Retinopathy stages using CNN.	The study uses a special kind of neural network (CNN) to automatically detect diabetic retinopathy (DR) in eye pictures. The network learns to recognize signs of DR like tiny bulges and bleeding in the retina. To make the network work well, the researchers use different techniques like setting good starting values for the network, a special way to calculate errors, and a method to prevent the network from memorizing the training data too much. Finally, they check how well the network works by comparing its predictions to actual diagnoses.	Detection and classification of Diabetic Retinopathy (DR) stages achieved notable performance metrics. It attained approximately 95% accuracy in a two-class classification (DR vs. no DR) .
Paper 11	Diabetic Retinopathy Using Machine Learning.	The study proposes a system to automatically detect diabetic retinopathy (DR) from retinal images. The system first converts the image to a different color space to make it easier to find yellow colored exudates, a sign of DR. Then, it performs various image processing techniques to isolate features like exudates, hemorrhages and microaneurysms. Finally, it uses a combination of three machine learning algorithms (Support Vector Machine, K-Nearest Neighbors, Random Forest) to classify the image as normal or abnormal.	The researchers achieved 82% accuracy in detecting diabetic retinopathy (DR) using a combination of image processing and machine learning. They analyzed images for signs of DR like hemorrhages and exudates. After pre-processing the images, they extracted features and fed them into three different classifiers. By combining the outputs of these classifiers, they achieved a precision of 86%, recall of 81%, and F1-score of 80%. This means the system correctly identified 86% of DR cases and missed 14%. It also correctly identified 81% of healthy cases and missed 19%.
Paper 12	Using Deep Learning Architectures for Detection and Classification of Diabetic Retinopathy.	The study employs a hybrid model combining VGG16 as a feature extractor and XGBoost as a classifier, alongside the DenseNet 121 architecture for diabetic retinopathy detection. VGG16 is modified for medical image classification, enhancing its structural capabilities. DenseNet 121 utilizes multiple dense blocks and a global average pooling layer to create a compact representation of features, followed by a fully connected layer for classification.	A hybrid model combining VGG16 and XGBoost, and the DenseNet 121 model. The hybrid model achieved an accuracy of approximately 79.50%, while the DenseNet 121 model significantly outperformed it with an accuracy of 97.30%. The models were trained on the APTOS 2019 Blindness Detection dataset, with 80% of the data used for training and 20% for testing.

Paper 13	Diabetic Retinopathy Classification Using CNN and Hybrid Deep Convolutional Neural Networks.	The study employs convolutional neural networks (CNNs) and transfer learning to classify diabetic retinopathy stages. Three models are developed: a custom CNN, a hybrid CNN with ResNet, and a hybrid CNN with DenseNet. Image preprocessing techniques enhance feature visibility, while data augmentation addresses class imbalance. The DenseNet model, leveraging pre-trained weights, achieved the highest accuracy of 96.22%, significantly outperforming the other models. The approach automates the detection process, facilitating early diagnosis and intervention for diabetic retinopathy, ultimately aiding in preventing vision loss in diabetic patients.	Self-supervised learning methods like CABNet achieved an accuracy of 84% on the EyePACS dataset. Ensemble-based techniques showed exceptional performance, with Tymchenko et al. reaching 99.3% accuracy using a three-headed CNN architecture. Additionally, Zhang et al. reported a sensitivity of 92% and an accuracy of 92.6% on the Messidor dataset. Overall, these advanced methods demonstrate significant improvements in classification accuracy, highlighting the effectiveness of both self-supervised and ensemble strategies in enhancing DR detection and diagnosis.
Paper 14	Diabetic Retinopathy Grading Using ResNet Convolutional Neural Network.	For features extraction and classification convolutional neural network (CNN) has been used. This paper has advocated the concept of transfer learning from ResNet. ResNet have multiple versions with different number of layers.	The proposed model achieved the accuracy of 69.4%.
Paper 15	Diabetic Retinopathy Stage Classification using Convolutional Neural Networks.	In this paper, we deploy three state-of-the-art representative convolutional neural network architectures, AlexNet, VGG16 and InceptionNet V3, for DR stage classification. We go through the architectures of these three models and explain further how we set up different configurations to leverage these CNNs for DR stage image classification.	The average accuracy of AlexNet, VGG16, and InceptionNet V3 are 37.43%, 50.03%, and 63.23%, respectively.

The literature review is performed on the topic of Classification of Diabetic Retinopathy. The key takeaway from this is that the model is trained on the dataset set available on the internet.

CHAPTER 3

REQUIREMENT SPECIFICATIONS AND PROBLEM STATEMENT

- Collect the Realtime dataset which is fundus images of retina with equal number of images in five classes of DR. The images are preprocessed to standardize resolution, and data augmentation is applied to expand the image set and maintain balance.

- Model Training: Use convolutional neural network (CNN) models to train the dataset, designed with the objective of categorizing the stages of DR. The models used in this project are: Dense Net 121, Mobile Net, InceptionV3 and VGG16.

- Feature Engineering and Integration: Extraction of useful features from the retinal fundus images with the deep learning model to improve its predictability and boost diagnostic accuracy.

- Model Evaluation and Validation: Assess the performance of the developed deep learning model using standard metrics, such as accuracy. Conduct a performance comparison with current advanced methods to evaluate the model's effectiveness and generalizability.

- User Interface and Deployment: Design an easy-to-use interface for users. Embed the model into a deployable software application or web-based platform for smooth integration into clinical workflows.

- Interpretability and Explainability: These techniques should be applied to make models interpretable and explainable, so they provide users with an insight into the decision-making process Visualize saliency maps, activation maps, or attention mechanisms, so as to emphasize areas of interest in the dermoscopic images.

- Performance Optimization and Scalability: Optimise the computational efficiency and memory footprint of the model to be run in real-time on a variety of hardware platforms. Look at methods of model compression, quantization, and parallelization to enhance scalability and deployment in resource-constrained environments. Continuously monitor and fine-tune the model performance based on feedback from clinical users and evolving datasets.

- Documentation and Knowledge Transfer: Document the entire development process including data collection architecture of model training procedure and results evaluation. Detailed documentation and tutorial to support knowledge transfer and reproducibility for

researchers and developers while promoting collaboration and knowledge sharing within the scientific community, as presented in peer-reviewed journals and conferences.

3.1 Software requirements

Programming Language and Frameworks:

• Python is the programming language used for model development, data manipulation and scripting

• Deep learning frameworks such as TensorFlow or Keras for building and training convolutional neural networks (CNNs)

• Libraries such as numpy, matplotlib for data visualization.

Development Environment:

• Integrated Development Environment (IDE) such as Jupyter Notebook for code development and experimentation 14

• Version control system (e.g. Git) for collaborative development, tracking changes and managing codebase

Data Management:

• Data preprocessing tools for standardizing image resolution and data augmentation.

Evaluation and Visualization:

• Libraries for model evaluation and validation, such as Scikit-learn, TensorFlow/Keras metrics, or PyTorch's evaluation modules

• Visualization tools for generating performance metrics, confusion matrices and ROC curves (e.g. Matplotlib, Seaborn)

User Interface and Deployment:

• Web development frameworks like React JS for building user interface and backend server

• Frontend technologies CSS for designing and implementing the user interface

• Cloud platforms (e.g. AWS, Google Cloud, Streamlit.io) or on-premise servers for hosting the deployed application

Documentation and Collaboration:

• Documentation tools such as Sphinx or MkDocs for generating project documentation and user manuals • Collaboration platforms like GitHub for version control, issue tracking, and project management

• Communication tools (e.g. Google Meet) for facilitating collaboration and communication among team members

Performance Optimization:

• Profiling tools for analyzing and optimizing model performance (e.g. TensorFlow)

• Libraries and techniques for model compression, quantization and parallelization to enhance scalability and deployment efficiency

Table 2: Software Requirements

Operating System	Windows, macOS, and Linux distribution
Programming Languages	Python
Deep Learning Framework	TensorFlow, Keras
Libraries and Packages	Seaborn, matplotlib, NumPy and pandas
Development Tools	Jupyter Notebook, Google Colab

3.2 Hardware requirements

• Central Processing Unit (CPU): Minimum intel Core i5 or AMD Ryzen 5 processor. Modern multi-core CPUs for running training and inference tasks. Higher clock speeds and multiple cores can expedite data preprocessing and model training

• Graphics Processing Unit (GPU): NVIDIA GeForce, Tesla P100, or Quadro GPUs with CUDA support for accelerating deep learning computations. GPUs significantly reduce training time and improve model convergence, especially for large datasets and complex architectures Operating System Windows, macOS, and Linux distributions Programming Languages Python, Deep Learning Framework TensorFlow, Keras Libraries and Packages Seaborn, matplotlib, pillow, glob, NumPy and pandas Development Tools Kaggle Notebook, Google Colab, Visual Studio Code or Jupyter Notebook.

• Random Access Memory (RAM): Minimum of 8GB of RAM to handle large datasets and model parameters during training. Recommended RAM size varies depending on dataset size and model complexity.

• Storage: Solid State Drives (SSDs) or Hard Disk Drives (HDDs) with ample storage capacity(at least 256GB) for storing datasets, model checkpoints, and software dependencies. SSDs offer faster read/write speeds, which can expedite data loading and preprocessing

• Networking: High-speed internet connection for downloading datasets, software libraries, and updates. Local area network (LAN) connectivity for collaboration and data sharing.

• Specialized Hardware Accelerators: Dedicated deep learning accelerators such as NVIDIA Tesla GPUs or Google Cloud TPUs for high-performance training and inference. These accelerators offer enhanced computational power and memory bandwidth, enabling faster model iterations and deployment

• Cloud Computing Resources: Access to cloud computing platforms for scalable compute resources and GPU instances. Cloud-based infrastructure facilitates distributed training, model deployment, and collaboration across geographically distributed teams

• Data Storage and Backup: Reliable data storage solutions such as Network Attached Storage (NAS) or cloud-based storage services for backing up and archiving datasets. Data redundancy and backup mechanisms to prevent data loss and ensure data integrity throughout the project lifecycle

• Peripheral Devices: Standard peripherals such as monitors, keyboards, and mice for interacting with the development environment. Graphics tablets or stylus input devices for annotating and processing dermoscopic images, if applicable.

• Temperature Control and Ventilation: Adequate cooling solutions (e.g., air conditioning, cooling fans) to prevent overheating of hardware components, especially during prolonged training sessions.

Table 3: Hardware Requirements

Processor	Intel Core i5 or AMD Ryzen processor
RAM	8 GB RAM
Hard disk	256 GB HDD or SSD
GPU	NVIDIA GeForce RTX 3050 or higher

3.3 Problem statement

Given the difficulties associated with manual diagnosis, there is a need for an automated system to reliably categorize the stages of diabetic retinopathy from retinal fundus images. This process is particularly error-prone in regions like coastal Karnataka, where data availability is limited. This system will enhance screening efficiency, enable early identification of high-risk patients, and assist doctors in diagnosis and treatment.

3.4 Objectives

- Develop a system to classify high-quality fundus images into four diabetic retinopathy stages: mild, moderate, severe, and proliferative, enhancing patient care through precise stage differentiation.

- Use actual fundus images and health data from coastal Karnataka to create a region-specific, accurate model reflecting local characteristics and diabetic retinopathy prevalence.

- Merge Kaggle datasets with real-time data from coastal Karnataka to create a more extensive and diverse training and validation set, enhancing the model's ability to generalize across different populations.

- Implement a deep learning model, specifically using convolutional neural networks for medical image analysis, to identify key features in fundus images corresponding to different diabetic retinopathy stages.

- Integrate a multitask learning approach into the model, allowing it to not only classify the stages of diabetic retinopathy but also predict disease progression or associated conditions using the features extracted from the images.

- Develop a tool to assist medical professionals in early detection, aiming to prevent further vision loss and control disease progression.

- Include a prediction component to estimate the likelihood of a patient's condition worsening, aiding clinicians in making informed treatment decisions.

- Design the system as an application to enhance clinicians' decision-making abilities by providing accurate classifications and predictions, supporting high-quality clinical decisions, improved patient outcomes, and personalized care.

- Evaluate the performance of the model trained on both real-time and combined datasets using metrics like accuracy, sensitivity, and specificity. Adjust the model based on validation set outcomes to improve accuracy and ensure its practical effectiveness in real-world scenarios.

3.5 Methodology

To determine the best model for precisely classifying fundus images into one of the four stages of diabetic retinopathy (DR), we methodically trained deep learning models using

a dataset gathered from patients at K.S. Hegde Medical Hospital in Mangaluru. Preprocessing techniques were applied, including resizing images to a suitable scale, normalization, and applying enhancements to improve visible features. To expand our dataset, we used augmentation techniques such as horizontal and vertical flips and 90-degree rotations.

For feature extraction, we employed different layers of a Convolutional Neural Network (CNN). The model was trained using the Adam optimizer, with categorical cross-entropy applied for multi-class classification and binary cross-entropy for binary classification. Performance was evaluated using metrics such as accuracy, precision, recall, and F1-score. This strategy enabled the model to make precise predictions of DR stages, supported by strong performance metrics.

Data Acquisition

This project, aimed at classifying medical images using pre-trained deep learning models, depends significantly on the datasets used. Two main datasets were incorporated in this study: one collected from a local hospital in our area and another publicly available dataset from online sources. These datasets formed a solid basis for training a CNN model, which was able to identify features to accurately categorize retinal fundus images into one of four stages of diabetic retinopathy (DR).

Real-Time Data Collection

K S Hegde Hospital in Mangaluru, Karnataka, has collected fundus image data from its patients, contributing valuable information from one of the region's largest healthcare institutions. This locally-sourced dataset forms the foundation of our project, enabling extraction of features that reflect health patterns unique to the local population. By focusing on this specific community, the model captures health trends that are often underrepresented in global diabetes research. This approach addresses the limitations of open datasets, which rarely include individuals from certain geographic or ethnic backgrounds, making our model more applicable and meaningful for local healthcare needs.

Retinal images will be taken with highly sophisticated fundus photography by which the details to be seen are blood vessels, macula and optic disc for the important features. The resolution of instruments used may be that of such resolutions and sensibility that

can easily detect microaneurysms, hemorrhages and neovascularization. These are small portion of very early diabetic retinopathy symptoms. Such resolutions of this range are required in order for proper diagnosis and classification into DR stages since at even lesser resolutions, details which cannot be spotted even in image structure is present. Individual retinal images captured were extensively annotated soon after captured by eye specialist of K S Hegde Hospital. Segmentation of DR stages was annotated into the separate sections that form: No DR, Mild NPDR, Moderate NPDR, Severe NPDR or PDR. The resultant labels then are the established ground truths on which the model gets trained upon. Remarkably reliable and accurately is the case with the model. Since the labeling process involved ophthalmologists' knowledge, the dataset is, no doubt, to become a valuable resource for the models on deep learning which shall be trained toward DR detection and classification.

Along with real-time data collected from medical institutions, this study also used a respected public dataset from Kaggle, namely the APTOS 2019 Blindness Detection dataset, which is well-known in clinical imaging research. This dataset contains a diverse set of retinal images with varying quality and resolutions. The images are categorized into five stages of diabetic retinopathy (DR): No DR, Mild, Moderate, Severe, and Proliferative DR.

Integrating the Kaggle dataset expands the coverage of the real-time dataset by incorporating a wide variety of diabetic retinopathy cases from different populations and geographic areas. This is especially advantageous as it enables the Convolutional Neural Network (CNN) model to be trained on retinal images from various demographic groups, which may include ethnic and age groups that are underrepresented in the real-time data.

The diversity within the Kaggle dataset also presents images across a spectrum of quality, from high-resolution, well-lit images to those that may be less clear or slightly blurred. This range simulates real-world challenges, making the dataset a valuable benchmark to evaluate model performance across different image qualities.

By combining both the hospital data and the Kaggle dataset, a more thorough evaluation of the model is achieved. The Kaggle dataset serves as a valuable benchmark, allowing for an assessment of the model's ability to generalize across a wider range of populations beyond the local dataset, which represents coastal Karnataka. Such comparisons allow for insights into the model's adaptability and effectiveness in varied demographic settings, which is crucial for deploying AI in healthcare contexts.

Merging the two datasets broadens the model's experience with a wide variety of DR cases, enabling a more comprehensive evaluation across different scenarios. This approach not only refines the model for local use but also demonstrates its potential for application across larger, more diverse populations.

Table 4: Image count and class distribution in the Realtime dataset.

Class number	Class name	Number of images	Percentage of images
0	No DR	1000	20%
1	Mild	1000	20%
2	Moderate	1000	20%
3	Severe	1000	20%
5	Proliferative DR	1000	20%

Table 5: Image count and class distribution in the Kaggle dataset

Class number	Class name	Number of images	Percentage of images
0	No DR	1000	20%
1	Mild	1000	20%
2	Moderate	1000	20%
3	Severe	1000	20%
5	Proliferative DR	1000	20%

Data Preprocessing:

Given the high-quality images used in training the deep learning model, data preprocessing was essential. Ensuring a consistent dataset helped enhance feature visibility and contributed to model robustness, preparing it for diverse scenarios that may arise in practice. For this study, both real-time data gathered from a hospital and a Kaggle dataset were utilized.

The hospital images underwent preprocessing to meet high standards for diabetic retinopathy classification. Real-world medical images often vary widely in quality due to factors like lighting conditions, the equipment used, and patient-specific characteristics. By applying preprocessing, we aimed to create consistent, high-quality images, reducing noise so that the deep learning model could focus on diagnostically relevant features.

The preprocessing phase began with resizing all images to a consistent resolution to align with the neural network's input specifications. Each image was adjusted to either 224x224 or 512x512 pixels. Standardizing image sizes is crucial for the CNN model, as it allows uniform feature extraction across the dataset while preserving essential retinal details for accurate diagnosis.

Next, pixel normalization was applied to adjust intensity values, reducing the impact of lighting variations. This process ensures a consistent intensity range across images, helping to minimize distortion from overexposed or underexposed areas. As a result, the model learned from genuine retinal features rather than inconsistencies due to brightness differences.

To enhance contrast, histogram equalization was used to make critical retinal features more prominent. This technique redistributes intensity values, highlighting subtle structures like faint blood vessels, microaneurysms, and hemorrhages. In the early stages of diabetic retinopathy, it makes these anomalies more detectable within the retinal architecture.

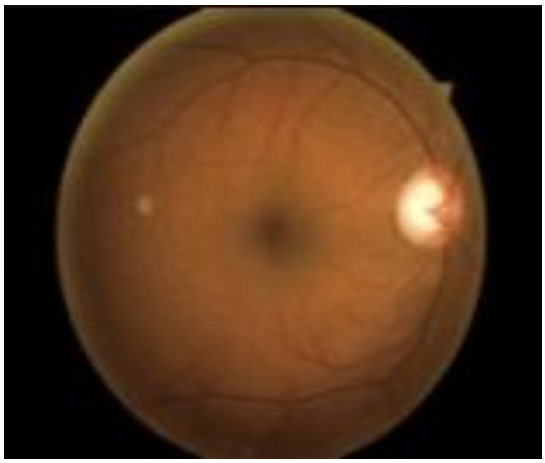
Additionally, cropping was performed to focus on diagnostically significant regions of the fundus images, particularly around the macula and optic disc. This approach excludes irrelevant background areas, helping the model concentrate on the specific retinal regions

affected by diabetic retinopathy. By honing in on these areas, the model's robustness is improved, ensuring it learns from features most relevant to disease progression.

To effectively combine the Kaggle dataset with the real-time data, additional preprocessing was carried out to ensure both datasets shared a consistent format. Images from the Kaggle dataset were resized to either 224x224 or 512x512 pixels to match the resolution of the real-time images. Pixel normalization was applied across both datasets to standardize brightness and contrast, with no additional enhancement or cropping. This unified preprocessing approach ensures each image undergoes the same preparation, enabling a fair and unbiased comparison during model training and testing. Consistency in preprocessing enhances the model's ability to generalize, enabling accurate classification of DR stages across both local and international datasets. This focus on uniformity prepares the model to perform well for local demographics while also benefiting from the broader diversity in the combined dataset, ultimately boosting robustness and accuracy.

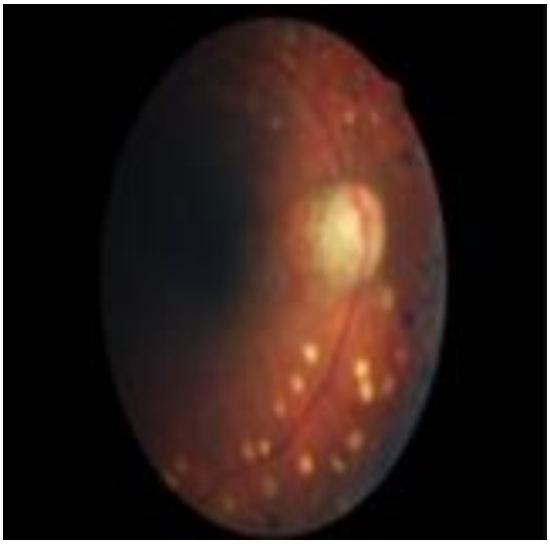
No DR

Mild NPDR



Moderate NPDR

Severe NPDR



Proliferative

Figure 4: Pre-processed fundus images with class label

Maintaining Dataset Balance: Ensuring a balanced dataset is crucial in classifying diabetic retinopathy (DR) stages, as well as in other medical imaging tasks. When certain DR stages are underrepresented, models may excel in identifying some classes but struggle with others, leading to bias. In this study, we maintain balanced representation for each DR stage—No DR, Mild, Moderate, Severe, and Proliferative DR—allowing the model to learn the distinguishing features of each category equally. This balance eliminates the need for methods like oversampling, under sampling, or reweighting to address class imbalances, as each category is proportionally represented in both the real-time dataset from KS Hegde Medical College and the additional Kaggle dataset. Training on a balanced dataset promotes consistent performance across all stages, avoiding the risk of overfitting to majority classes at the expense of minority ones. In medical

applications, particularly those affecting patient care, it is essential to minimize the risk of classifying severe DR as a milder stage, as accurate classification at all levels supports more dependable clinical outcomes. By upholding this balance, the model can achieve high accuracy in detecting DR stages across varied conditions and datasets.

Balanced data also provide a stable foundation for the model's evaluation, giving it the ability to generalize effectively without the need for artificially adjusting class distributions. However, if inconsistencies are detected in further studies, more advanced techniques, such as integrating additional data from minority groups, may become necessary. This approach allows for a robust and well-generalized model that can accurately identify varying levels of DR, avoiding common issues stemming from unbalanced datasets.

In the first phase, it focused on training CNN models with high-resolution fundus images obtained from patients who suffer from diabetes in coastal Karnataka, particularly at KS Hegde Medical College in Mangalore. This type of image holds unique demographic and clinical characteristics that ascertain fine retinal characteristics common to such patients, like the early manifestations of microaneurysms or hemorrhages. This data is useful for proper identification of DR stages and potential early health indicators, such as cardiovascular risks, that relate to local healthcare. The preprocessing steps involved transformations, normalization, and contrast enhancement of images to place emphasis on the crucial features. The architectures used with CNN include DenseNet-121, MobileNet, InceptionV3, and VGG-16, which have been modified for accurate classification of DR in this population. We have trained the model on the Kaggle "APTOS 2019 Blindness Detection" dataset. It is a heterogeneous collection of retinal images labeled by DR stage and belonging to a wide geographic source. That heterogeneity in the Kaggle set gives an ideal benchmark where the model learns generalized DR patterns and boosts its potential to recognize DR stages in more varied populations. Refining the same CNN architectures on this dataset enables us to assess whether patterns learned from the local dataset are transferable to different demographics, providing an initial indication of the model's generalization capability. The final combination of the local coastal Karnataka dataset with the Kaggle dataset results in an extensive, varied collection of images and labels. Compatibility across both datasets was ensured through standardized preprocessing steps, including resampling, normalization, and contrast adjustment. This combined dataset captures distinctive features of the local population as well as absorbs greater diversity, thus providing both region-specific and global patterns with a chance to be identified by the CNN models. We can then compare the results of this merged dataset with the previous results of training individual datasets to determine whether merging the data improves performance. The method above is a

combination of methods enhancing the model's robustness across various demographics, a key factor for AI-driven healthcare solutions to be widely applicable for DR detection.

3.5.1 Model evaluation

In this stage, the performance of each CNN model—MobileNet, DenseNet-121, InceptionV3, and VGG-16—must be rigorously evaluated to ensure accurate classification across all five levels of diabetic retinopathy (DR). This evaluation includes detailed comparisons across various datasets to validate each model's reliability and adaptability to different data sources. The study utilizes three types of datasets: real-time images collected from KS Hegde Medical Academy, the Kaggle "APTOS 2019 Blindness Detection" dataset, and a combined dataset integrating both sources. This multi-dataset approach enables a comprehensive analysis of key performance factors, including accuracy, generalization, and reliability. By comparing results from each dataset, we can assess how well each model performs on local data versus a more varied, globally representative dataset, and whether combining datasets enhances the model’s overall performance in classifying DR stages. This method allows for a deeper understanding of each model’s capability to generalize effectively across different demographics while maintaining high classification accuracy.

CHAPTER 4
SYSTEM DESIGN

To train these images, Fundus preprocessing input images are used and then filters are applied to multiple layers, scaled, normalized, and enhanced. The Rectified Linear Unit, or ReLU, is called after each convolution layer. To train these images, the input image is first used for the fundus and then filters are applied to different layers. While feature maps with down sampling minimize computations, the max pooling layers minimize spatial dimensions while maintaining more significant features.

Training becomes stable with batch normalization. This results in a significantly reduced reliance on initialization and a significantly faster convergence. In CNN topologies, layers are specifically designed to interact with a random subset of neurons in each layer during training. Therefore, the model needs to generate an optimal representation that is better for both snapshot classification and image classification from Kaggle datasets. Now that the convolutional process has finished outputting the output, the output is flattened into a single layer and then sent through the convolutional process to combine the combined spatial features into high-resolution representations.

This will include additional SoftMax activations in the sigmoid for binary predictions such as health and multi-unit DR levels. These architectures need to be trained appropriately using loss functions such as binary cross-entropy for health prediction and categorical cross-entropy for DR level classification. Using four CNN architectures improves the representation of input images in classification by creating different types of feature extraction.

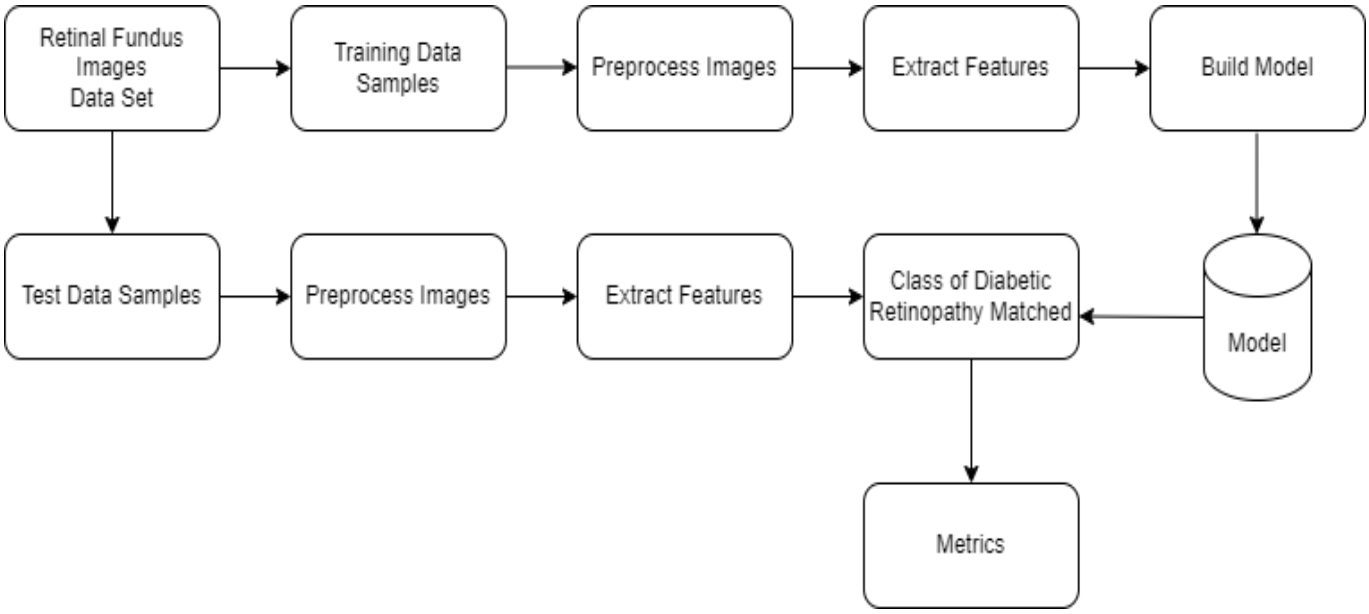


Figure 5: Basic architecture of CNN

4.1 System architecture

DiabEye is a website that helps people with diabetes check their eye health. Users can upload photos of their eyes and the website will analyze them to see if there are signs of a serious eye problem called diabetic retinopathy. It's a simple way to monitor eye health and take steps to protect vision.

4.2 Data flow diagram

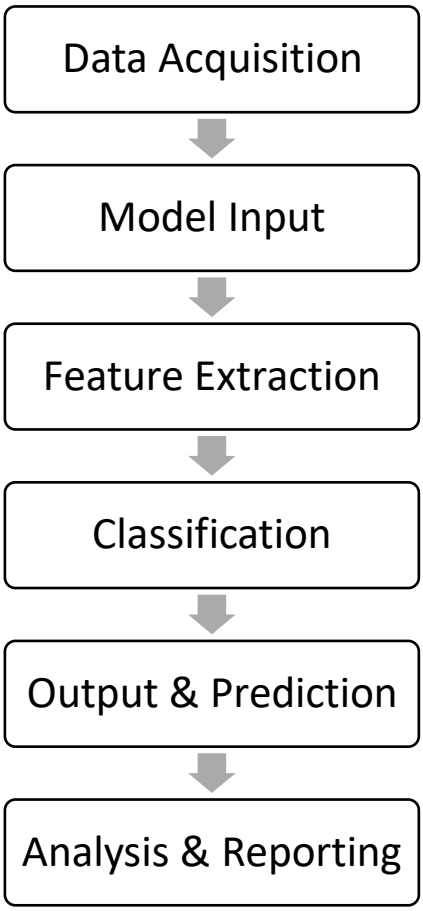


Figure 13: Flow Diagram

The data flow diagram helps to analyze different steps involved in the backend part of the project. The above figure shows the different steps involved. The initial stage is the acquisition of the data. The basic idea of our project is real-time data collection. The data is collected from a local hospital in our area which is confirmed by a doctor, therefore the collected data is labeled. Since the data is labeled, we use supervised learning technique. In our case, the input of the model is an image of the retinal fundus. The dataset is divided into 70% train data, 20% validation data and 10% test data. The data is then trained on existing deep learning models based on the CNN architecture. Our undertaking involves training the dataset on four deep learning models which includes MobileNet, Dense Net 121, Inception V3 and VGG 16. The model is trained on 50 epochs in each case. The model extracts useful capabilities from the retinal fundus pictures and learns from it. After

the version is trained the accuracies of every model is compared. The comparison led us to the conclusion of considering the Dense Net 121 model because of its highest training accuracy. The prediction code for the image classification is then applied on Dense Net model. The model is then examined on the images for checking out purposes. Along with training the models on our dataset, we also performed comparative analysis by training the models on Kaggle dataset and the merged dataset. The results are analyzed and visualized with help of different graphs.

4.3 Use case diagram

A use case diagram is a visual representation in Unified Modeling Language (UML) that illustrates how a system interacts with external actors, such as users or other systems. It presents a high-degree evaluation of the machine's functionality, focusing at the "what" as opposed to "how". In a use case diagram, actors are depicted as stick figures or icons, representing entities that have interaction with the machine. Use cases, shown as ellipses, represent specific functionalities offered by the system. Each use case encapsulates a hard and fast of actions that the machine performs to deliver a particular value to an actor.

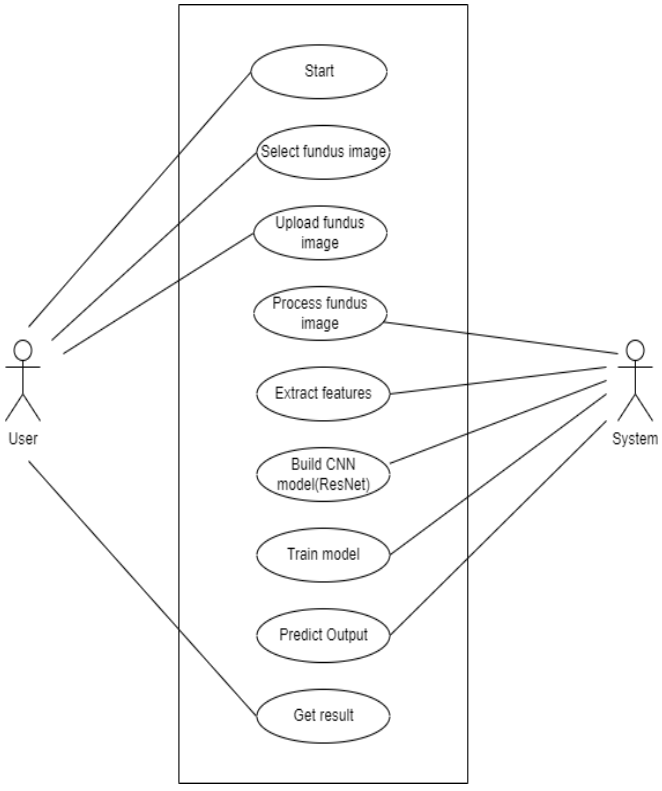


Figure 14: Use case Diagram

Select Fundus Image: The user initiates the process by selecting the fundus image that they want to identify. This selection could be from a list of available fundus images.

Upload Image: The selected image is then uploaded to the image recognition system. This upload manner can happen through an internet interface, or some other user interface supplied by the device.

Process Image: Upon uploading, the system processes the image to prepare it for analysis. This preprocessing step may additionally contain various operations together with resizing the image to a standard size.

Extract Feature: The machine then extracts relevant capabilities from the processed image. Features could include color histograms, texture descriptors, part facts, or any other applicable records that distinguishes one photo to other.

Train Model: The dataset which we have collected are trained on existing deep learning models based on CNN architecture. We have taken into consideration four models which incorporates MobileNet, VGG 16, Inception V3 and Dense Net 121. Out of those Dense Net 121 Model has confirmed the highest schooling accuracy.

Predict Output: Once the model is educated, it can be used to predict the form of DR level in a brand new fundus photo. When a new image is uploaded, the trained CNN model analyzes its features and makes a prediction based on its learned knowledge.

Get Result: Finally, the system returns the identification results to the user. Users can then use this information for numerous purposes inclusive of research or schooling.

CHAPTER 5

SYSTEM IMPLEMENTATION

```
from flask import Flask, request, jsonify
from flask_cors import CORS
import tensorflow as tf
from tensorflow.keras.preprocessing.image import load_img, img_to_array
import numpy as np
from tensorflow.keras.preprocessing.image import ImageDataGenerator
import os
from keras.preprocessing import image
```

Figure 15: Image Processing Imports

The above code imports several libraries for building a web application and processing images using deep learning techniques.

```
app = Flask(__name__)
CORS(app)  # Enable CORS for all routes

# Load your model once when the app starts
loaded_model = tf.keras.models.load_model(r"C:\Users\HP\Downloads\DiabEye\modelmobile\DenseModel.h5")

# Define image parameters
img_width, img_height = 224, 224
batch_size = 32
```

Figure 16: Flask App Initialization and Model Loading

The above code prepares a Flask application to receive and process images. It loads a pre-trained model used for image classification or analysis.

```
test_datagen = ImageDataGenerator(rescale=1./255)

# Directory for test images
test_dir = r'C:\Users\HP\Downloads\DiabEye\ddata\test'

# Get class labels from the generator
test_generator = test_datagen.flow_from_directory(
    test_dir,
    target_size=(img_width, img_height),
    batch_size=batch_size,
    class_mode='sparse',  # Use 'sparse' for integer-encoded labels
    shuffle=False
)
```

Figure 17: Test Data Generator Configuration

The above code prepares the test data for model. It creates an Image Data Generator to preprocess the images and then creates a data generator to read the images from the test directory and prepare them in batches for the model to use during evaluation.

```python
@app.route('/predict', methods=['POST'])
def predict():
    if 'file' not in request.files:
        return jsonify({'error': 'No file part'}), 400

    file = request.files['file']
    if file.filename == '':
        return jsonify({'error': 'No selected file'}), 400

    # Load and preprocess the image
    img_path = os.path.join("uploads", file.filename)
    os.makedirs("uploads", exist_ok=True)  # Ensure the uploads directory exists
    file.save(img_path)

    img = image.load_img(img_path, target_size=(img_width, img_height))
    img_array = image.img_to_array(img)
    img_array = np.expand_dims(img_array, axis=0)  # Expand dimensions to create a batch with a single image
    img_array /= 255.  # Normalize the pixel values
```

Figure 18: Flask Route for Image Prediction

The above code snippet is responsible for handling the image upload, preprocessing, and potentially making a prediction using a machine learning model.

```python
    # Make prediction
    prediction = loaded_model.predict(img_array)

    # Get the predicted class
    predicted_class_index = np.argmax(prediction)
    predicted_class_name = class_labels[predicted_class_index]
    confidence_level = prediction[0][predicted_class_index]

    # Return the results
    return jsonify({
        'confidence': float(confidence_level),
        'predicted_class_index': int(predicted_class_index),  # Convert to standard Python int
        'predicted_class_name': predicted_class_name
    })

if __name__ == '__main__':
    app.run(debug=True)
```

Figure 19: Prediction and Response Generation

The above code snippet takes an image and makes a prediction using a pre-trained model and returns the prediction results in a JSON format.

CHAPTER 6

SYSTEM TESTING

6.1. Unit Testing

Unit testing focuses on testing the individual software components in isolation to ensure that every module performs specific tasks it has been designed for. For example, you may test a data acquisition module to verify if the module requires to retrieve fundus images correctly from a database or outside APIs. In the case of a preprocessing module, you have to ensure that the image in view is appropriately resized, normalized, and artifacts removed from it. Apart from testing functionality, unit tests ensure that edge cases, including missing or corrupted images and error conditions, are dealt with correctly. This kind of testing allows developers to identify bugs directly and correct them in time, without adding code to the rest of the system.

6.2. Integration Testing

After individual components have passed their unit tests, integration tests verify that they can all work together. It ensures that the flow of data from one module to the other, like from the data acquisition module to preprocessing, then to feature extraction, and finally to model training and inference. The method will also verify inputs and outputs between modules to ensure they there are in proper format. For example, it should verify that the preprocessed images are passed in the correct way into the CNN model for feature extraction and training. This step catches problems that may come out when the modules interact, although they may be alright isolated.

6.3. System Testing

System testing is testing the system as one integral unit to validate its functionality and performance in a real-world environment which includes uploading several fundus images, so as to see how good the system does in classifying the image into four DR stages. It should classify DR stage correctly: No DR, Mild NPDR, Moderate NPDR, Severe NPDR and Proliferative DR.correc. System testing also evaluates the responsiveness of a system under the realistic workload without crash or slow-down, by which it determines the performance of the system under increased demand.

6.4. User Acceptance Testing (UAT)

The most important aspect of user acceptance testing is that it will ensure the system meets the expectations of the end-users. UAT can be done by getting real users to

interact with the system by uploading images, viewing diagnostic results, and giving their findings on the usability, reliability, and effectiveness of the system. Testing is mainly on whether the system is easy to use. For example, if the users feel that the system is too slow or hard to understand, necessary enhancements are made on the basis of that feedback before being deployed.

6.5. Performance Testing

Performance testing tests the system's working and performance with different operational conditions. This is achieved through the assessment of some metrics, including but not limited to, inference latency: which at which the system will process and deliver predictions for new fundus images; throughput or how many images the system will process within a given timeframe; and finally, resource utilization the consumption of the CPU, memory, and the GPU. Performance testing establishes the bottlenecks that slow down the system, such as extended inference times when processing a large number of images. It also optimizes configurations such as batch size, parallel processing, and resource allocation to ensure it performs optimally under different loads.

6.6. Security Testing

This is because the system will deal with very sensitive patient information. Areas of potential vulnerabilities should be identified and mitigated. For instance, one would check for unauthorized access to the system, verify whether only a authorized users-clinicians or personnel in healthcare can see the data or diagnostic results, and ensure data transmission is secured by encrypting during transfer. The security testing also verifies whether the system adequately protects sensitive information against typical security threats such as data breaches or cyberattacks ensuring compliance with privacy-related legislations like HIPAA or GDPR.

6.7. Regression Testing

Regression testing ensures that when new features, updates, or fixes to bugs are introduced into the system they do not inadvertently cause problems in previously functional parts. For example, if a new feature is added to improve the preprocessing of images, regression testing will ensure that this change does not negatively impact the training and inference capabilities of the model. The system's existing functionality is particularly asserted by re-running some test cases during proof that its change of implementation has not compromised. Continuous regression testing ensures that the system remains reliable and stable throughout its lifecycle, even as it evolves.

6.8. Re-Running Test Cases from Previous Testing Phases

Test cases run again from previous testing phases, such as unit tests, integration tests, and system tests, are needed after implementing system changes, such as updates or patches. It is important to make sure that new changes will not break functionality that already exists and still make the system behave as expected. Through the revisiting of test cases, the developer can verify if the same issues which were solved earlier have resurfaced or if the system has become vulnerable because of the changes made in the system.

6.9. Documentation and Reporting

Documentation and reporting are essential to keep a record of all testing activities and results. Include writing and maintaining detailed test plans that describe the scope, objectives, and methods for each phase of testing. Test cases are developed to explain specific input scenarios and expected outcomes. Results of actual testing - all pass/fail status, performance, and bugs-are documented. A test report is created at the end of each test phase summarizing findings including defective or problematic issues identified and how they were resolved. Additionally, proper documentation will indicate to all concerned whether a system has been tested satisfactorily or otherwise and serve as a reference for future maintenance or debugging.

CHAPTER 7
RESULT

The model that was developed for the classification of five different stages in Diabetic Retinopathy namely No DR, Mild NPDR, Moderate NPDR, Severe NPDR and Proliferative DR. With DenseNet model, it was found to identify with a good accuracy of 94.00%.

After performing experiments, we could achieve the following result of experiment where accuracy for our project is showing. We used four CNN model architectures for the same data and saw the accuracies for all of them. We could see the accuracy along with loss for the following architectures: DenseNet 121, Mobile Net Inception V-3 and VGG-16 in Table 7, Table 8 and Table 9 respectively.

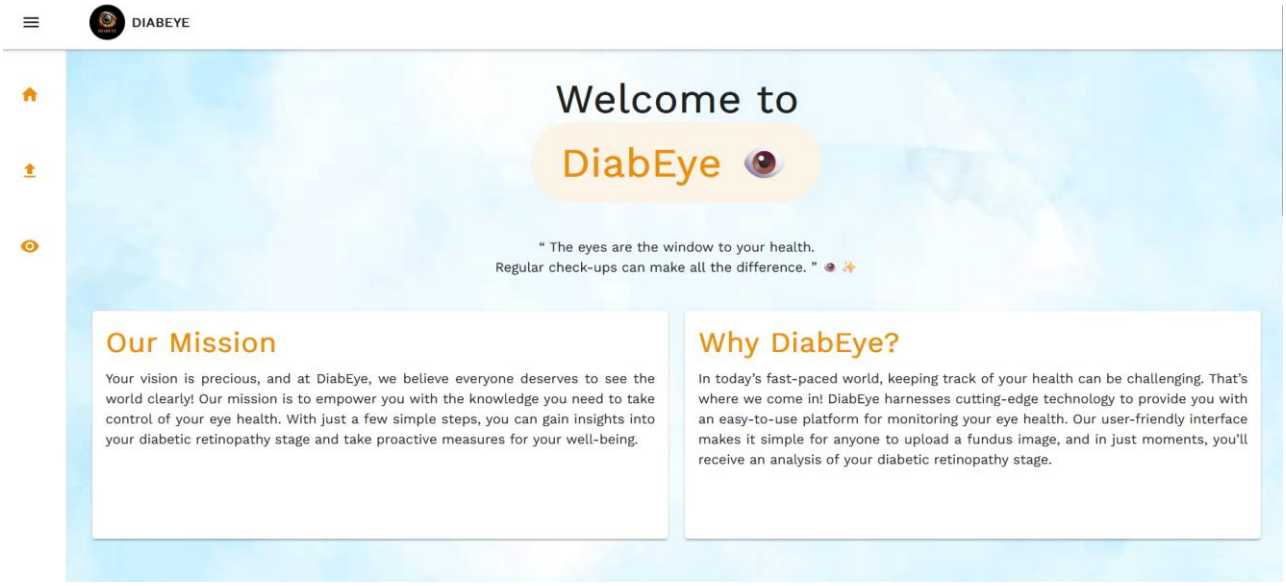


Figure 6: DiabEye Website Homepage

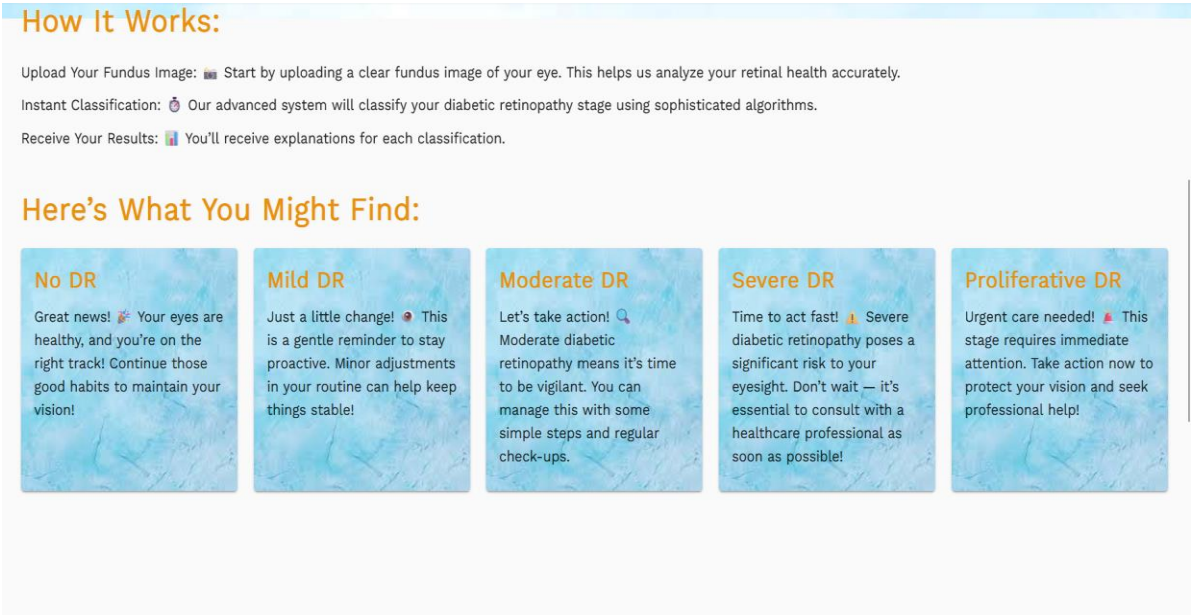


Figure 7: DiabEye Workflow

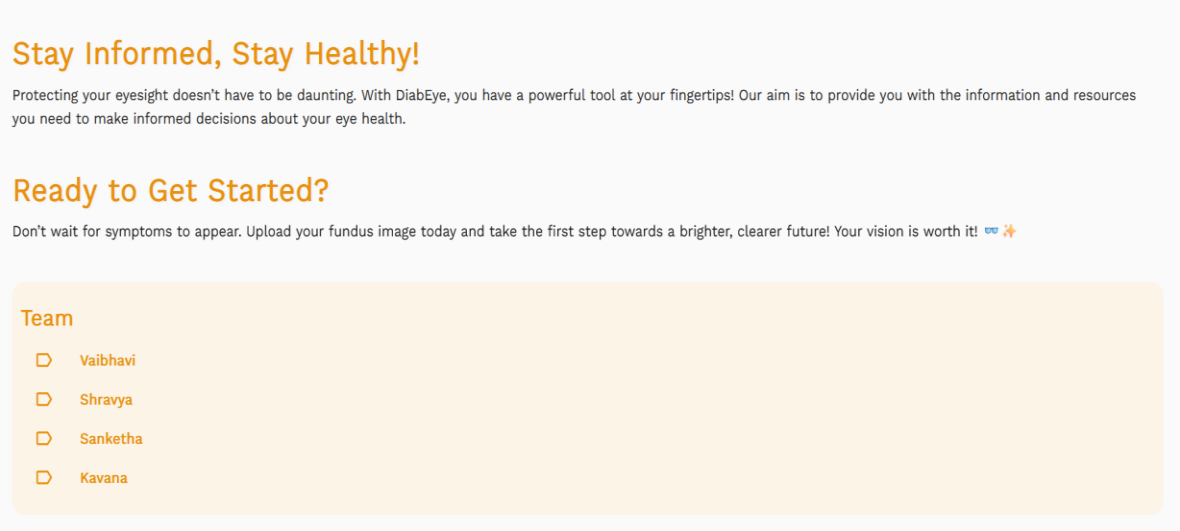


Figure 8: DiabEye Team Information

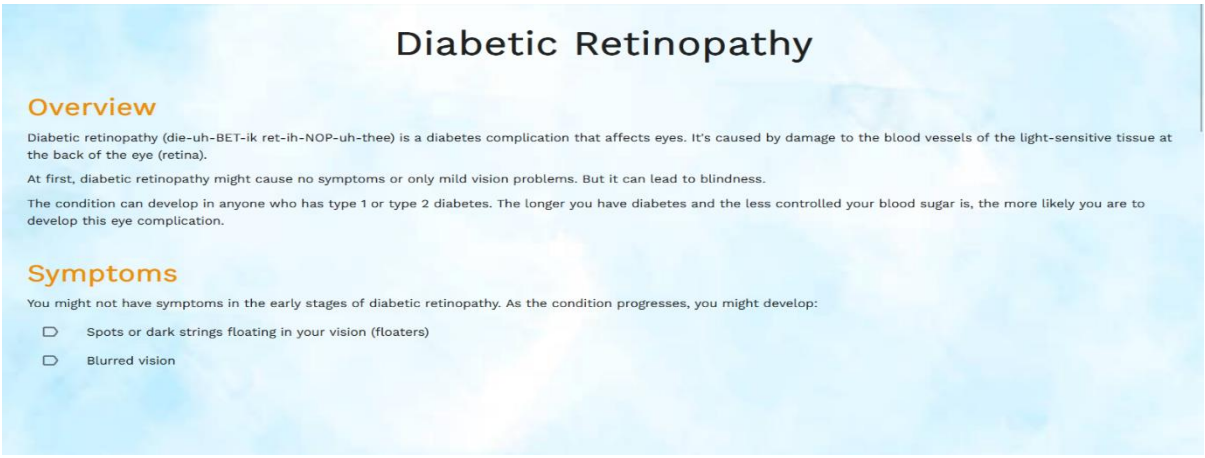


Figure 9: Diabetic Retinopathy Overview

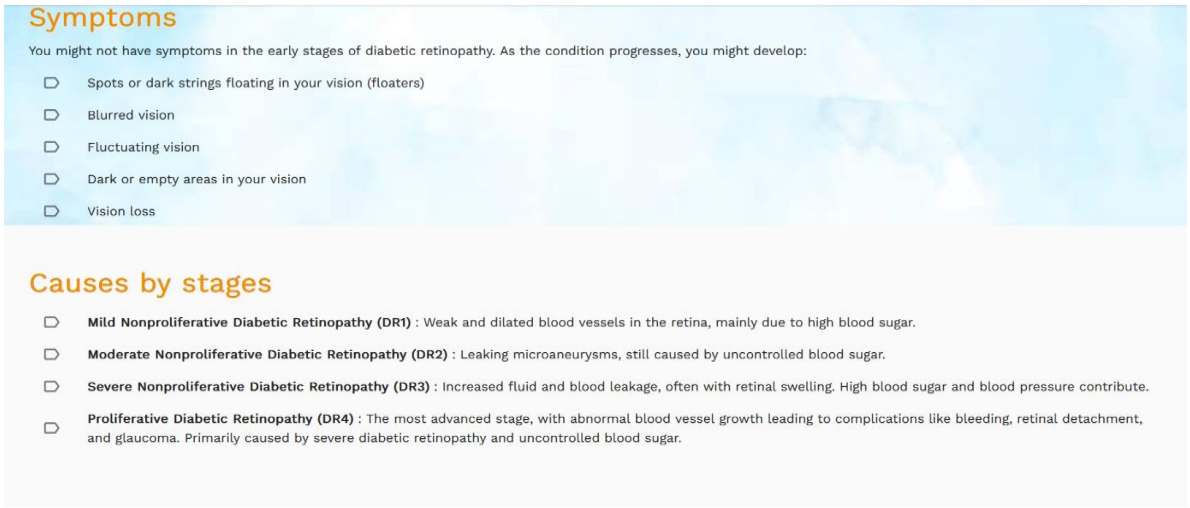


Figure 10: Diabetic Retinopathy Symptoms and Causes

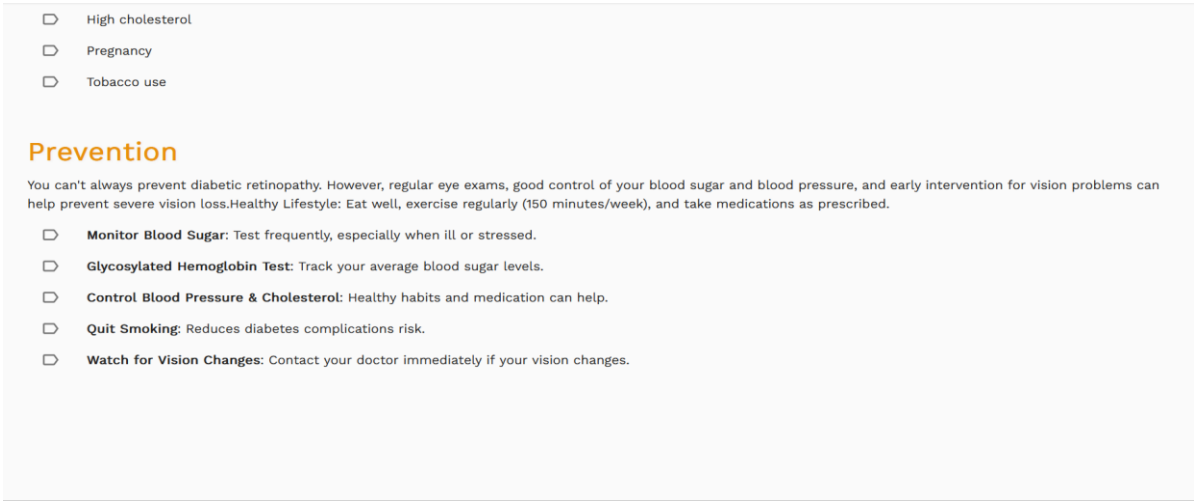


Figure 11: Diabetic Retinopathy Prevention

Figure 12: DiabEye Prediction Result

Table 7. Accuracy of Real-time dataset.

CNN Model	Dataset	Training Accuracy	Testing Accuracy	Validation Accuracy
Mobile-Net	Realtime	96.88%	91.40%	75.00%
InceptionV-3	Realtime	75.00%	71.40%	75.00%
DenseNet	Realtime	96.60%	94.00%	93.35%
VGG-16	Realtime	43.75%	43.00%	62.50%

Table 8. Accuracy of Kaggle dataset.

CNN Model	Dataset	Training Accuracy	Testing Accuracy	Validation Accuracy
Mobile-Net	Kaggle	90.62%	46.40%	75.00%
InceptionV-3	Kaggle	62.50%	44.40%	62.50%
DenseNet	Kaggle	74.12%	40.00%	40.93%
VGG-16	Kaggle	30.36%	28.40%	29.13%

Table 9. Accuracy and Loss of Kaggle and Real-time dataset.

CNN Model	Dataset	Training Accuracy	Testing Accuracy	Validation Accuracy
Mobile-Net	Merged	65.62%	64.10%	56.25%
InceptionV-3	Merged	81.25%	55.50%	56.25%
DenseNet	Merged	85.37%	67.00%	67.14%
VGG-16	Merged	36.58%	35.00%	35.58%

```
Final Training Accuracy: 96.60%
Final Validation Accuracy: 93.35%
Found 500 images belonging to 5 classes.
16/16 [==============================] - 59s 4s/step - loss: 0.1661 - accuracy: 0.9400
Test Accuracy: 94.00%
```

Figure 20: MobileNet model for real-time dataset

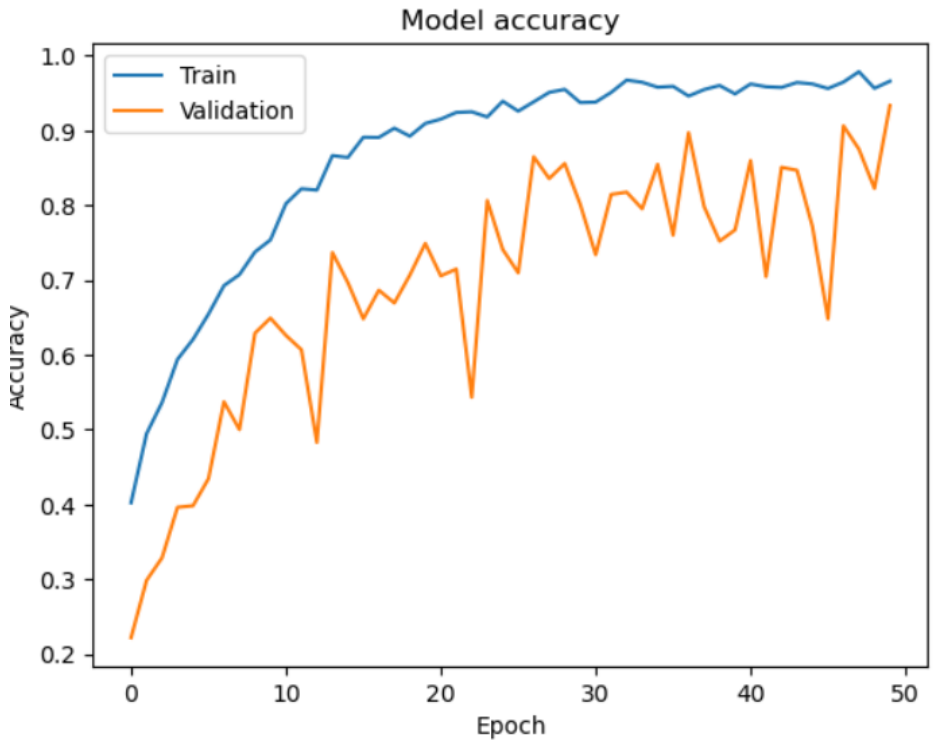


Figure 21: Model accuracy for real-time dataset

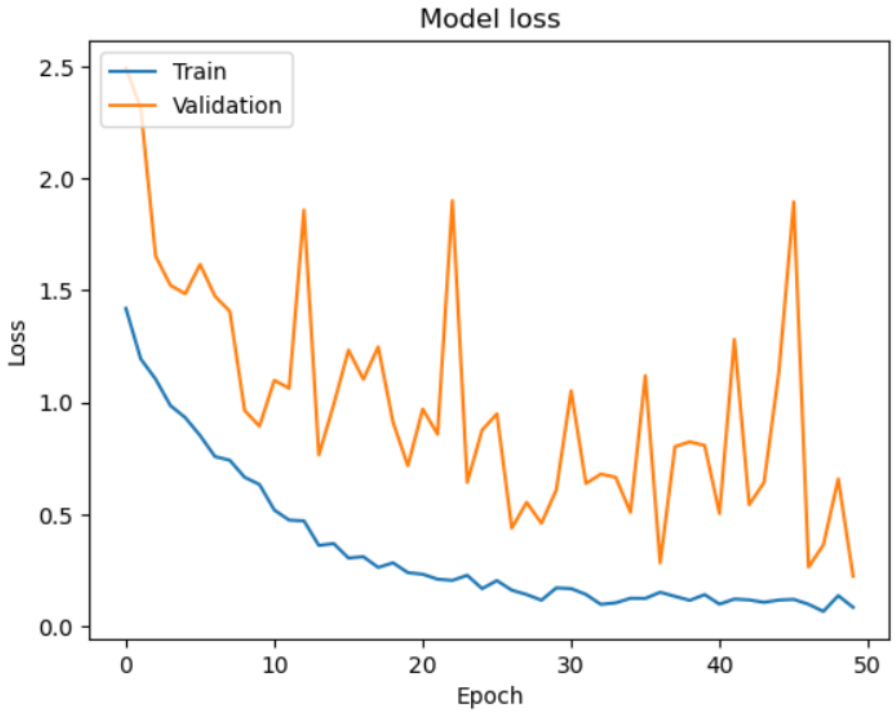


Figure 22: Model loss for real-time dataset

```
Final Training Accuracy: 90.62%
Final Validation Accuracy: 75.00%
Found 500 images belonging to 5 classes.
16/16 ━━━━━━━━━━━━━━━━━━━━ 16s 966ms/step - accuracy: 0.3604 - loss: 1.8117
Test Accuracy: 46.40%
```

Figure 23: MobileNet model for kaggle dataset

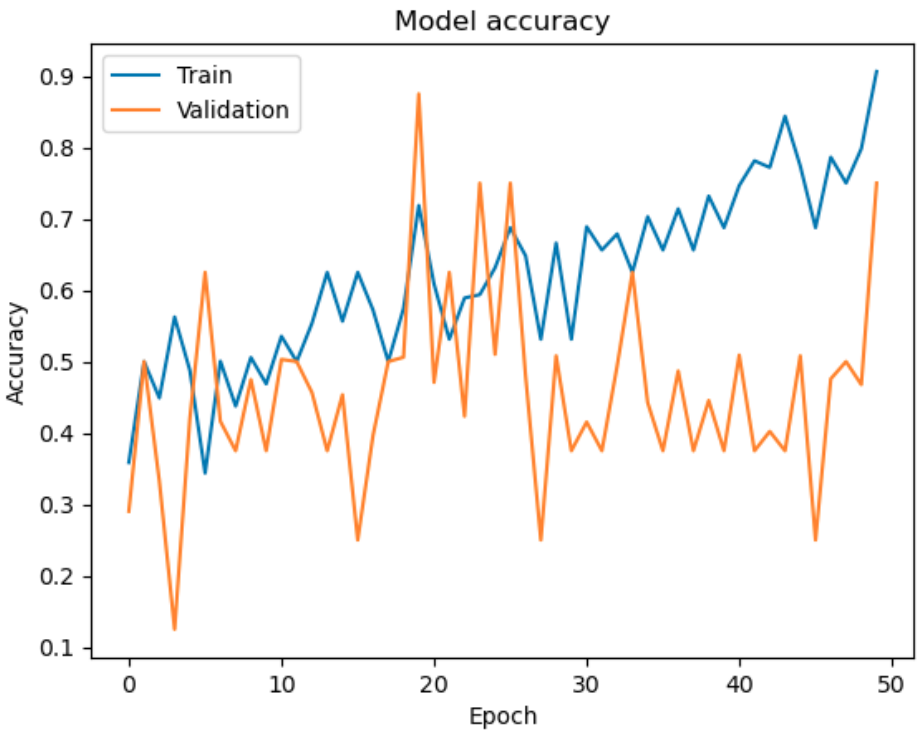


Figure 24: Model loss for kaggle dataset

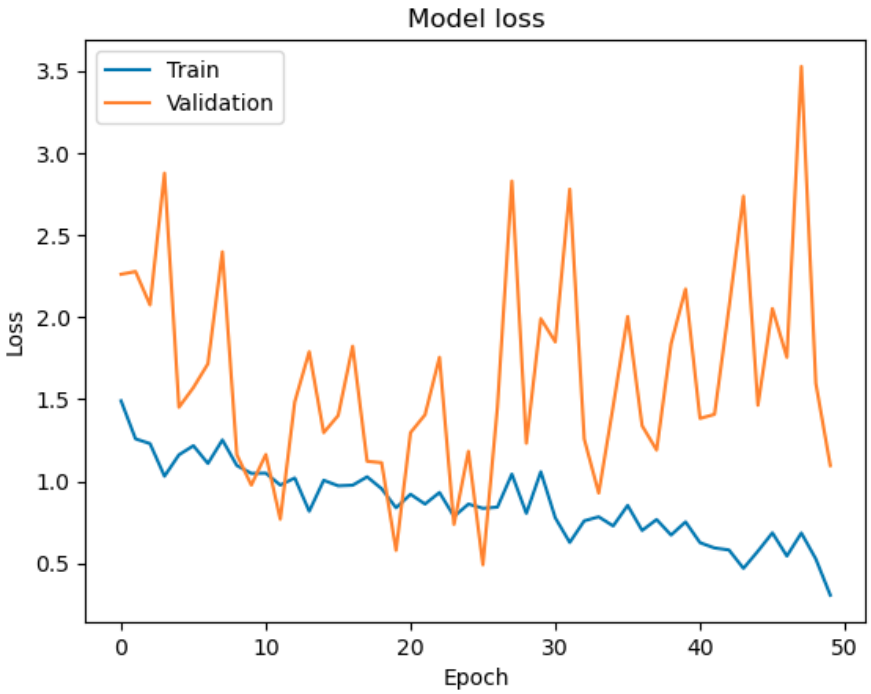


Figure 25: Model loss for kaggle dataset

```
Final Training Accuracy: 74.13%
Final Validation Accuracy: 40.93%
Found 500 images belonging to 5 classes.
16/16 [==============================] - 28s 2s/step - loss: 2.4470 - accuracy: 0.4000
Test Accuracy: 40.00%
```

Figure 26: DenseNet model for merged dataset

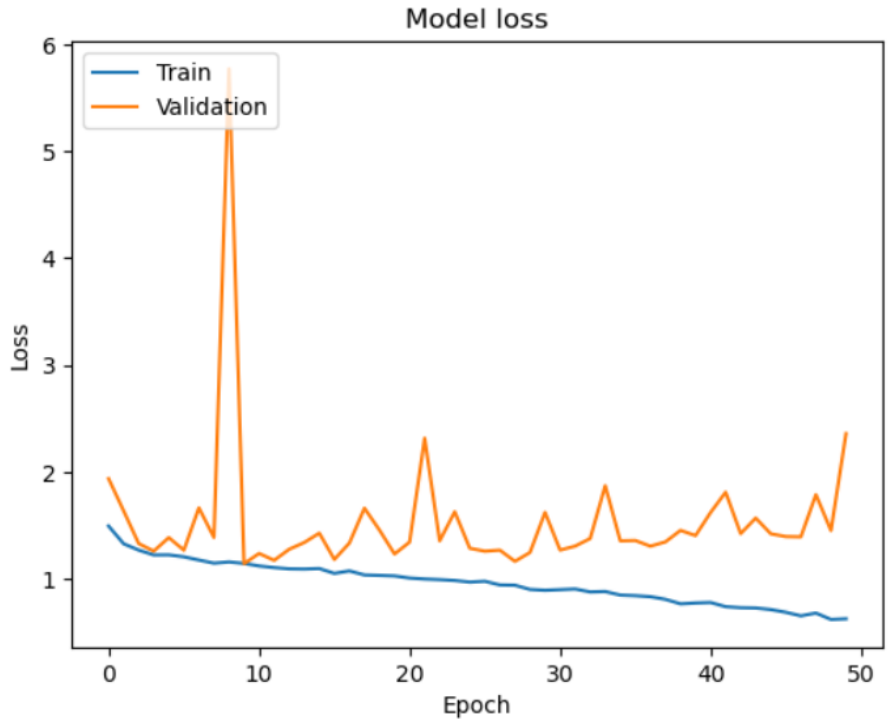


Figure 27: Model loss for merged dataset

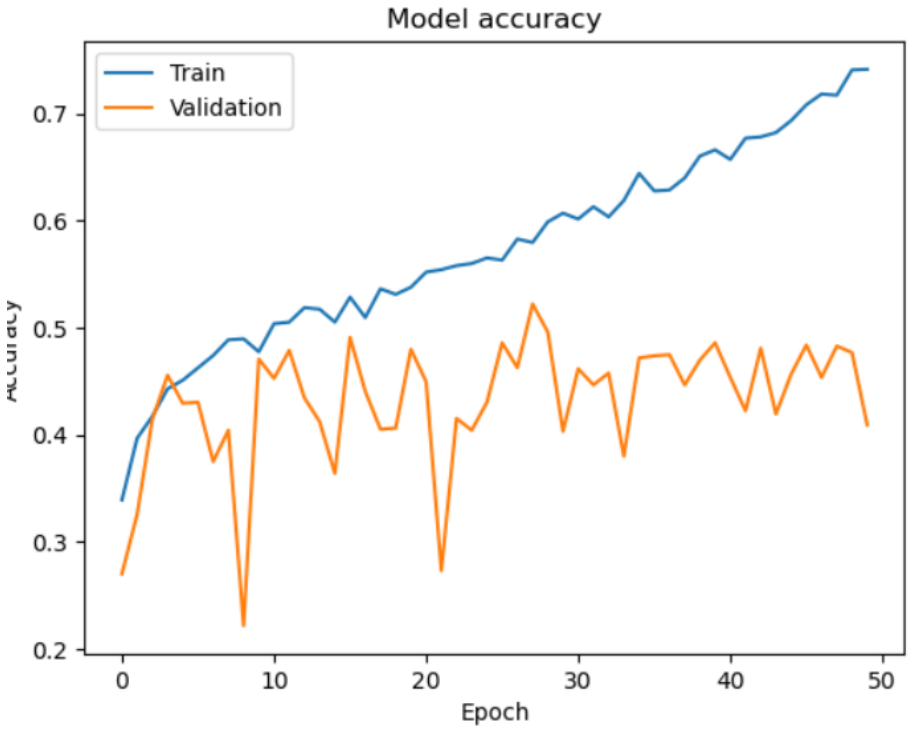


Figure 28: Model accuracy for merged dataset

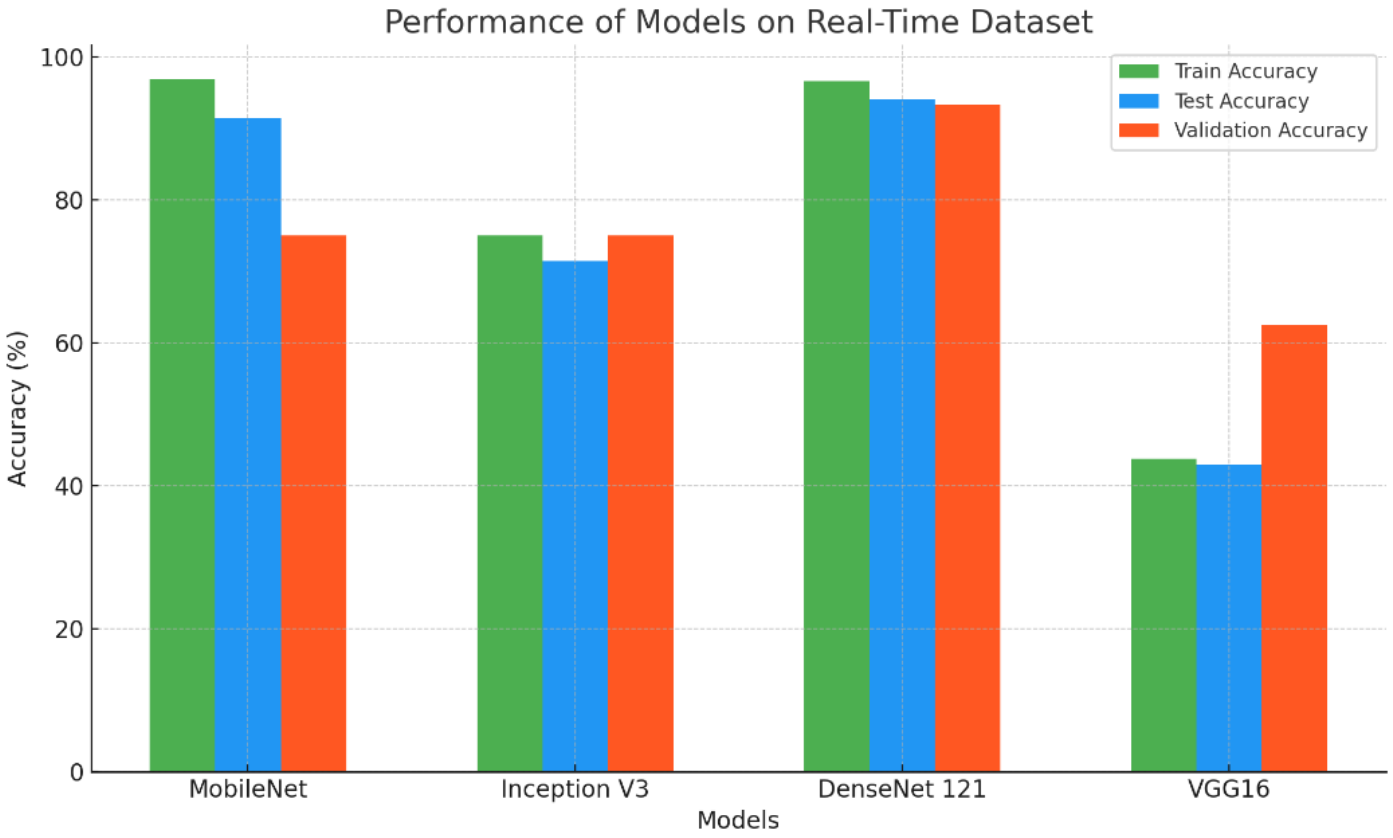


Figure 29: Bar graph for Real-Time dataset Models performance

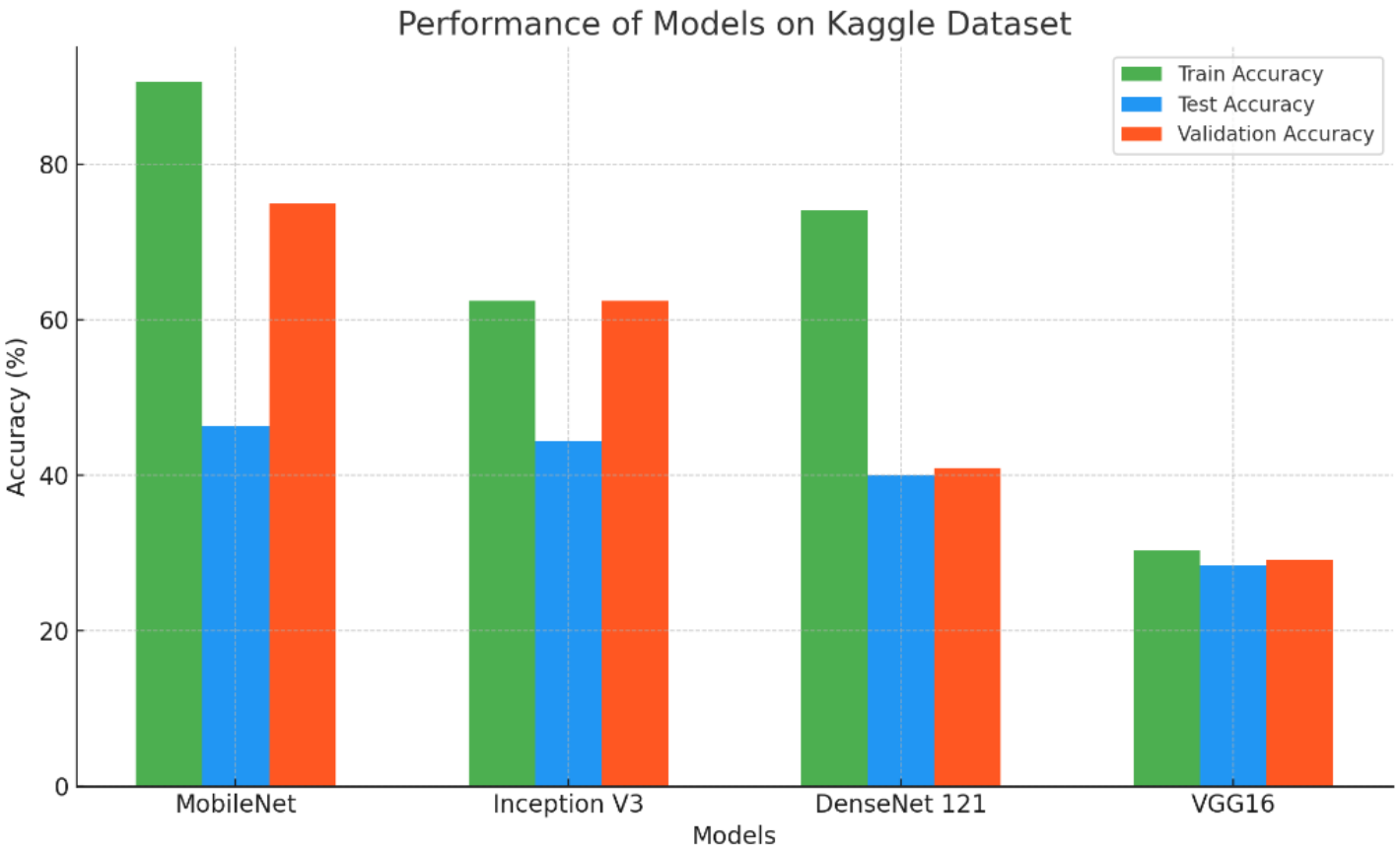


Figure 30: Bar graph for Kaggle dataset Models performance

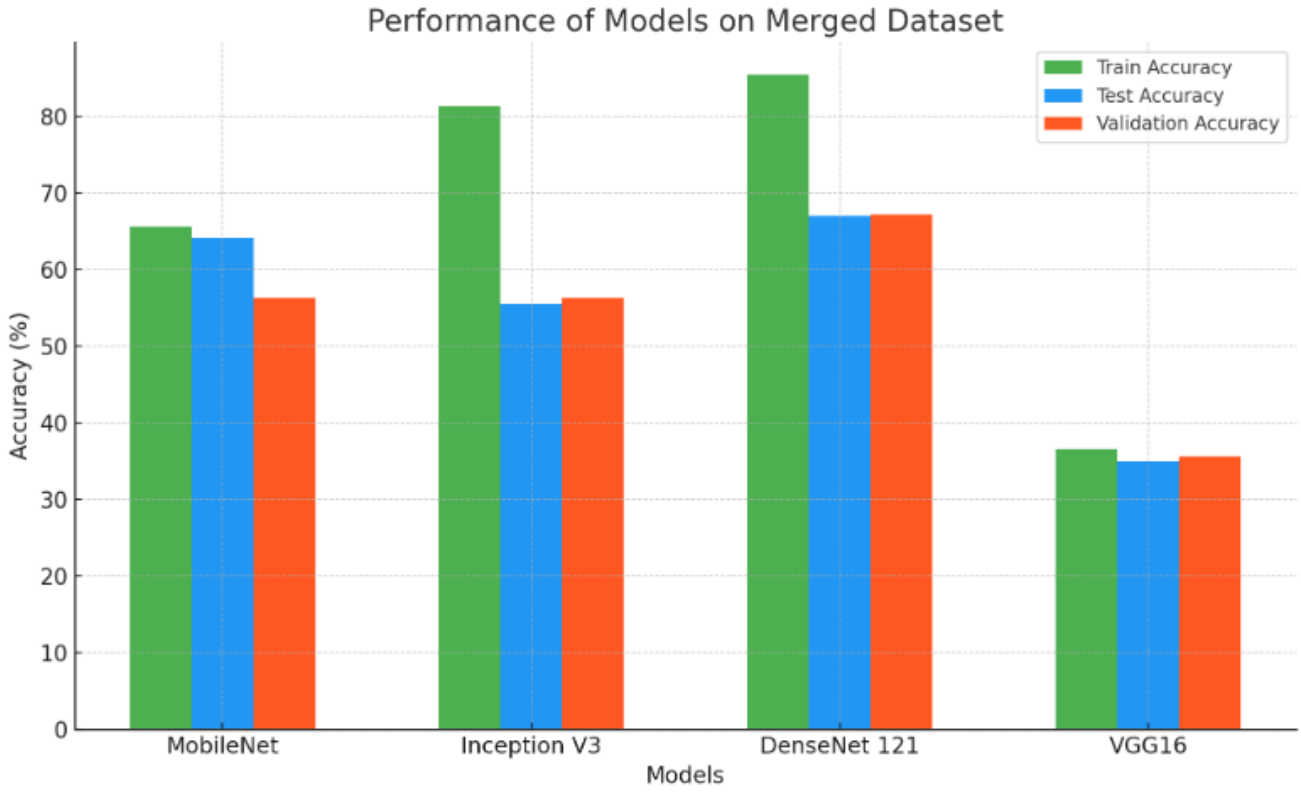


Figure 31: Bar graph for Merged dataset Models performance

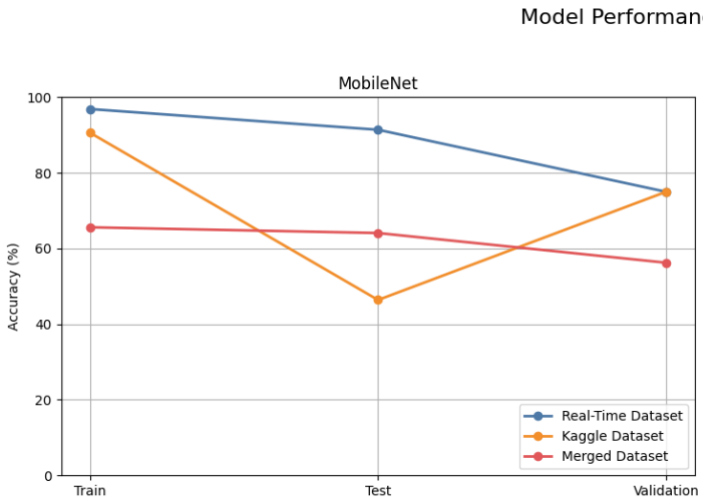


Figure 32: MobileNet Performance

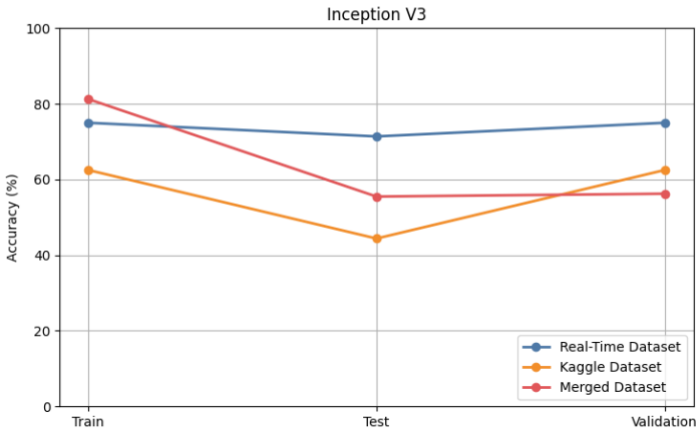


Figure 33: Inception V3 Performance

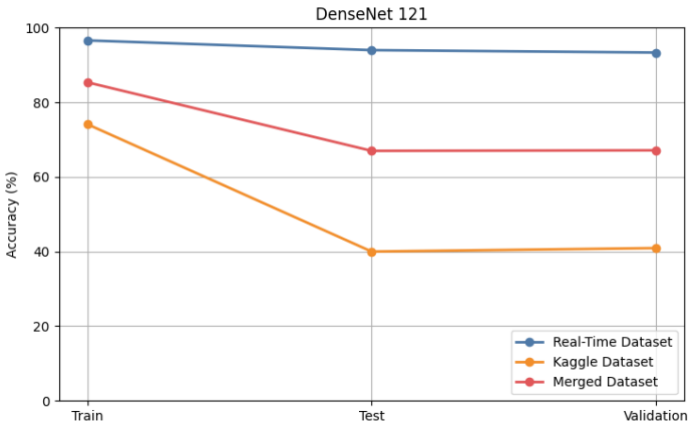


Figure 34: DenseNet 121 Performance

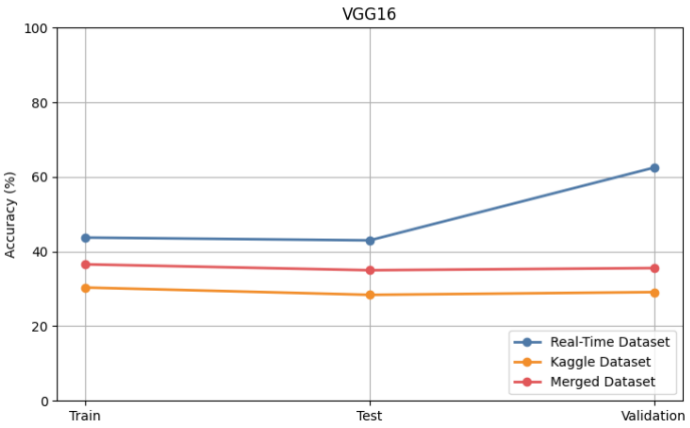


Figure 35: VGG 16 Performance

CHAPTER 8
CONCLUSION

A condition known as diabetic retinopathy affects people with diabetes and eventually leads to blindness in middle-aged people. The diabetes causes damage to retina over time. In the early stages the symptoms are hard to detect but by the time it gets detected much damage has been occurred by the time it first gets diagnosedUsually, an ophthalmologist assists in the diagnosing method by means of the use of the retina's fundus image. There is an pressing want for an automated technique to determine the phases of diabetic retinopathy with a purpose to facilitate early-level identification. The system accepts any fundus image from one of five classifications as input.

The image used as input is processed using various image transformation techniques. A dense internet version used for fine turning has been used for function extraction and classification. It enables faster training and better performance and there is a slight scope for probing other pre trained models. The predominant challenges with this paper are collection of datasets and choice of model to categorise the ranges of diabetic retinopathy with maximum accuracy. The project is approached in three phases classification using real time data, classification using the Kaggle dataset and merging actual real time and Kaggle data to verify accuracy and efficiency. We will finally compare how the models will perform with all types of datasets.

CHAPTER 9
REFERENCES

[1] Mishra S, Hanchate, S. and Z., Saquib "2020, Diabetic retinopathy detection using deep learning; In IEEE, p515-20" cited December 6th."International conference on Smart Technologies in Computing, Electrical, and Electronics.2020"

[2] Ramchandre S,B. P, S Pharand, K javalipande, H Diabatic retinopathedy detection with Deep transfer approach. " in conference of "INICE (INOCON); Published date: February 29" 2020.".p1-6";cite December 7th DOI#: 10.1109/INOCON.2020.1

[3] Jayakumari, C., Lavanya, V., & Sumesh, E. P. Automated Diabetic Retinopathy Detection and Classification Using ImageNet Convolution Neural Network Using Fundus Images, Proceedings of the International Conference on Smart Electronics and Communication (ICOSEC), Article No. 577577 (2020), 577-582. DOI: 10.1109/ICOSEC.2020.1

[4] Kalyani, G., Janakiramaiah, B., Karuna, A., & Prasad, L. V. N. Diabetic Retinopathy Detection and Classification Using Capsule Networks, Complex & Intelligent Systems (2023) 2651–2664,

[5] Abidalkareem, A. J., Abd, M. A., Ibrahim, A. K., Zhuang, H., Altaher, A. S., & Ali, A. M. Diabetic retinopathy severity level classification using multimodel convolutional neural networks. In Proceedings of the IEEE International Conference on Machine Learning and Applications (ICMLA), pp. 1404-1408. IEEE.

[6] Sallam, M. S., Asnawi, A. L., & Olanrewaju, R. F. Diabetic Retinopathy Grading Using ResNet Convolutional Neural Network. In Proceedings of the IEEE Conference on Big Data and Analytics(ICBDA)(pp.73-78).IEEE.

[7] Sathwik, A. S., Agarwal, R., Jubilson, A., & Basa, S. S. Diabetic Retinopathy Classification Using Deep Learning. EAI Endorsed Transactions on Pervasive Health and Technology

[8] Karki, S. S., & Kulkarni, P. Diabetic Retinopathy Classification Using a Combination of EfficientNets. Proceedings of the 2021 International Conference on Emerging Smart Computing and Informatics, Pune, India, 7-11 December 2021. IEEE

[9] Ghosh R, Ghosh K, and Maitra S: Automatic detection and classification of diabetic retinopathy stages using CNN. Proceedings of the 2017 4th International Conference on Signal Processing and Integrated Networks, Noida, India, March 1-2, 2017. IEEE.

[10] Wang, X., Lu, Y., Wang, Y., & Chen, W.-B. "Diabetic Retinopathy Stage Classification Using Convolutional Neural Networks." Proceedings of 2018 IEEE International

Conference on Information Reuse and Integration for Data Science (IRI). Piscataway, NJ: IEEE, 2018. DOI: 10.1109/IRI.2018.00074 .

[11] Supriya Mishra, Seema Hanchate , “Diabetic Retinopathy Detection using Deep Learning,” IEEE Trans. Instrum. Meas., vol. 58, no. 4, pp. 1170 – 1175, April 2009.

[12] Satwik Ramchandre, Bhargav Patil, Shardul Pharande, Karan Javali, Himangi Pande, “ A Deep Learning Approach for Diabetic Retinopathy detection using Transfer Learning ,” 2020 IEEE International Conference for Innovation in Technology (INOCON) Bengaluru, India. Nov 6-8, 2020. [13] Jayakumari.C, Vidhya Lavanya, Sumesh E P, Automated Diabetic Retinopathy Detection and classification using ImageNet Convolution Neural Network using Fundus Images , Proceedings of the International Conference on Smart Electronics and Communication (ICOSEC 2020) IEEE Xplore Part

[14] G. Kalyani, B. Janakiramaiah, A. Karuna ,L. V. Narasimha Prasad4, Diabetic retinopathy detection and classification using capsule networks, Complex Intelligent Systems (2023) 9:2651–2664 https://doi.org/10.1007/s40747-021-00318-9.

[15] Abidalkareem, Moaed A. Abd, Ali K. Ibrahim, Hanqi Zhuang1, Diabetic Retinopathy (DR) Severity Level Classification Using Multimodel Convolutional Neural Networks, Jama, vol. 316, no. 22, pp. 2402–2410, 2016.