“**OCULAR DISEASE RECOGNITION USING DEEP LEARNING**”

A Project Report submitted to

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR.

In Partial Fulfillment of the Requirements for the Award of the degree of

BACHELOR OF TECHNOLOGY
IN
COMPUTER SCIENCE AND ENGINEERING
BY

E. VENKATA RAM SAI	**20121A3120**
K. SONIKA	**20121A3126**
L. GOURI PRIYANKA	**20121A3132**
S. VINISH RANGANATH	**20121A3156**

Under the Guidance of

Mr. N. BALAKRISHNA, M.Tech,(Ph.D.)
Assistant professor,
Dept of CSE, SVEC

Department of Computer Science and Engineering
SREE VIDYANIKETHAN ENGINEERING COLLEGE
(Affiliated to JNTUA, Anantapuramu)
Sree Sainath Nagar, Tirupathi – 517 102

2020-2024

SREE VIDYANIKETHAN ENGINEERING COLLEGE

(Affiliated to Jawaharlal Nehru Technological University Anantapur)
Sree Sainath Nagar, A. Rangampet, Tirupati – 517 102, Chittoor Dist., A.P.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the Project Work entitled

"Ocular Disease Recognition Using Deep Learning"

is the bonafide work done by

E. VENKATA RAM SAI	**20121A3120**
K. SONIKA	**20121A3126**
L. GOURI PRIYANKA	**20121A3132**
S. VINISH RANGANATH	**20121A3156**

In the Department of Computer Science and Engineering, Sree Vidyanikethan Engineering College, A. Rangampet. is affiliated to JNTUA, Anantapuramu in partial fulfillment of the requirements for the award of Bachelor of Technology in Computer Science and Engineering during 2020-2024.

This is work has been carried out under my guidance and supervision.

The results embodied in this Project report have not been submitted in any University or Organization for the award of any degree or diploma.

Internal Guide

Mr. N. Balakrishna
Assistant Professor
Dept of CSE
Sree Vidyanikethan Engineering College
Tirupathi

Head

Dr. B. Narendra Kumar Rao
Prof & Head
Dept of CSE
Sree Vidyanikethan Engineering College
Tirupathi

INTERNAL EXAMINER

EXTERNAL EXAMINER

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

VISION AND MISSION

VISION

To become a Centre of Excellence in Computer Science and Engineering by imparting high quality education through teaching, training and research.

MISSION

The Department of Computer Science and Engineering is established to provide undergraduate and graduate education in the field of Computer Science and Engineering to students with diverse background in foundations of software and hardware through a broad curriculum and strongly focused on developing advanced knowledge to become future leaders.

Create knowledge of advanced concepts, innovative technologies and develop research aptitude for contributing to the needs of industry and society.

Develop professional and soft skills for improved knowledge and employability of students.

Encourage students to engage in life-long learning to create awareness of the contemporary developments in computer science and engineering to become outstanding professionals.

Develop attitude for ethical and social responsibilities in professional practice at regional, National and International levels.

Program Educational Objectives (PEO's)

1. Pursuing higher studies in Computer Science and Engineering and related disciplines

2. Employed in reputed Computer and I.T organizations and Government or have established startup companies.

3. Able to demonstrate effective communication, engage in team work, exhibit leadership skills, ethical attitude, and achieve professional advancement through continuing education.

Program Specific Outcomes (PSO’s)

1. Demonstrate knowledge in Data structures and Algorithms, Operating Systems, Database Systems, Software Engineering, Programming Languages, Digital systems, Theoretical Computer Science, and Computer Networks. (PO1)

2. Analyze complex engineering problems and identify algorithms for providing solutions (PO2)

3. Provide solutions for complex engineering problems by analysis, interpretation of data, and development of algorithms to meet the desired needs of industry and society. (PO3, PO4)

4. Select and Apply appropriate techniques and tools to complex engineering problems in the domain of computer software and computer based systems (PO5)

Program Outcomes (PO's)

1. Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems (**Engineering knowledge**).

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

3. 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 (**Design/development of solutions**).

4. Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions (**Conduct investigations of complex problems**).

5. Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations (**Modern tool usage**)

6. 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 (**The engineer and society**)

7. Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of,

and need for sustainable development (**Environment and sustainability**).

8. Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice (**Ethics**).

9. Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings (**Individual and team work**).

10. 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 (**Communication**).

11. 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 (**Project management and finance**).

12. 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 (**Life-long learning**).

Course Outcomes

CO1. Knowledge on the project topic (PO1)

CO2. Analytical ability exercised in the project work.(PO2)

CO3. Design skills applied on the project topic. (PO3)

CO4. Ability to investigate and solve complex engineering problems faced during the project work. (PO4)

CO5. Ability to apply tools and techniques to complex engineering activities with an understanding of limitations in the project work. (PO5)

CO6. Ability to provide solutions as per societal needs with consideration to health, safety, legal and cultural issues considered in the project work. (PO6)

CO7. Understanding of the impact of the professional engineering solutions in environmental context and need for sustainable development experienced during the project work. (PO7)

CO8. Ability to apply ethics and norms of the engineering practice as applied in the project work.(PO8)

CO9. Ability to function effectively as an individual as experienced during the project work. (PO9)

CO10. Ability to present views cogently and precisely on the project work. (PO10)

CO11. Project management skills as applied in the project work. (PO11)

CO12. Ability to engage in life-long leaning as experience during the project work. (PO12)

CO-PO Mapping

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3	PSO4
CO1	3												3			
CO2		3												3		
CO3			3												3	
CO4				3											3	
CO5					3											3
CO6						3										
CO7							3									
CO8								3								
CO9									3							
CO10										3						
CO11											3					
CO12												3				

(Note: 3-High, 2-Medium, 1-Low)

DECLARATION

We hereby declare that this project report titled **"Ocular Disease Recognition Using Deep Learning"** is a genuine project work carried out by us, in **B.Tech *(Computer Science and Engineering - Artificial Intelligence)*** degree course of **Jawaharlal Nehru Technological University Anantapur** and has not been submitted to any other course or University for the award of any degree by us.

Signature of the student

1. E. Venkata Ram Sai
2. K. Sonika
3. L. Gouri Priyanka
4. S. Vinish Ranganath

ACKNOWLEDGEMENT

We are extremely thankful to our beloved Chairman and founder **Dr. M. Mohan Babu** who took keen interest to provide us the infrastructural facilities for carrying out the project work.

We are highly indebted to **Dr. B. M. Satish**, Principal of Sree Vidyanikethan Engineering College for his valuable support and guidance in all academic matters.

We are very much obliged to **Dr. B. Narendra Kumar Rao,** Professor & Head, Department of CSE, for providing us the guidance and encouragement in completion of this project.

We would like to express our indebtedness to the project coordinator, **Ms. K. Ghamya**, Assistant Professor, Department of CSE, for her valuable guidance during the course of project work.

We would like to express our deep sense of gratitude to **Mr. N. Balakrishna**, Assistant professor, Department CSE, for the constant support and invaluable guidance provided for the successful completion of the project.

We are also thankful to all the faculty members of CSE Department, who have cooperated in carrying out our project. We would like to thank our parents and friends who have extended their help and encouragement either directly or indirectly in completion of our project work.

ABSTRACT

Ocular diseases present a significant public health concern globally, warranting precise and efficient diagnostic methodologies. This research aims to develop an approach for identifying ocular diseases by classifying eight different eye conditions using fundus images. Leveraging transfer learning with EfficientNet, a powerful convolutional neural network architecture which is employed to learn discriminative features from the images, facilitating precise classification of various ocular diseases. Performance evaluations on the Ocular disease dataset ODIR highlight the practicality and effectiveness of our scheme. In addition to diagnostics, this combination has the potential for application in the fields of telemedicine and telehealth, enabling rapid and reliable diagnosis of eye diseases, especially in restricted or remote areas. The combination of Transfer Learning and CNN not only improves the accuracy of diagnosis but also simplifies the analysis process, helping to increase the efficiency of treatment and improve the patient's outcome effect.

Index Terms — Convolutional Neural Network (CNN), Transfer Learning, Ocular Diseases, Efficiency, Medical Imaging.

Table of content

CHAPTER - 1
INTRODUCTION

Ocular diseases pose significant challenges in the realm of medical diagnosis due to their intricate nature and potential impact on vision health [9]. The most prevalent eye conditions are myopia, cataracts, diabetic retinopathy, and various others. Timely and accurate recognition of these conditions is important for efficient care as well as the preservation of visual function [12]. In current years, advancements in deep learning, a subset of artificial intelligence (AI), have sparked a transformative wave in the area of medical imaging analysis, offering promising solutions for ocular disease detection and diagnosis [10]. The complexity of ocular pathology, spanning a spectrum of conditions like glaucoma, diabetic retinopathy, AMD, and more demands precision in analysis. Traditional diagnostic methods relying solely on manual examination or standard imaging techniques often face limitations in early detection and precise classification of these diseases. However, the existence of deep learning models, particularly convolutional neural networks (CNN), has opened avenues to revolutionize the way ocular diseases are detected and diagnosed. A convolutional neural network[7] (CNN) can do it capture the pattern at any point across the retina by allowing it to do so filter to browse the entire image and make a pattern suitable. The filter moves across the image in steps which decides how much the filter must shift in response image pattern. The entire network operates under a single, continuous function, transforming raw pixel values from the input image directly into class probabilities at the output. The CNN architecture takes advantage of the fact that the inputs are images, enabling the encoding of certain properties architecture.

CNNs, adept at learning hierarchical representations from data, have shown remarkable capabilities in processing and extracting intricate features from medical images, including those obtained from various ocular imaging[11] modalities like fluorescein angiography, fundus photography, and optical coherence tomography (OCT). The utilization of CNNs allows for automated analysis and pattern recognition within these images, enabling the identification of subtle disease indicators that might escape human observation.

Moreover, the combination of deep learning techniques with transfer learning methodologies such as EfficientNet, presents an innovative approach for model building. This process can potentially provide clearer and more detailed representations, aiding in the accurate characterization and classification of ocular diseases[8]. In this research, we classify diverse eye disorders, facilitating advancements in diagnostic methods for these conditions.

The most eight prevalent eye conditions that are detected in the Ocular Disease detection are:

- Glaucoma
- Diabetic Retinopathy
- Age-related Macular Degeneration(AMD)
- Myopia
- Hypertension
- Cataract
- Normal
- Other abnormalities/diseases

1.1. STATEMENT OF PROBLEM

The problem addressed is to classify the ocular diseases accurately with greater accuracy in determining the true symptoms of the person. The implementation of this can be achieved using the combined approach of Transfer Learning and CNN. The Transfer Learning aims to decrease model training time, improve performance and then retrain. It improves the accuracy of diagnosis but also simplifies the analysis process, helping to increase the efficiency of treatment and improve the patient's outcome effect.

1.2. OBJECTIVES

- To predict an eye disease among various fundus images using deep learning.
- To predict whether the ocular disease occurs or not given various aspects of human life
- To predict which type of disease has occurred in the eye.

1.3. SCOPE

- Used to predict the Ocular Disease, Future prediction based on the attributes.
- News journal will able to predict Ocular Disease forecasting using these models.
- It may take preventive measure by recognizing, forecasting of the Ocular disease.

1.4. APPLICATIONS

- Useful in Automated screening & early detection

  Deep learning aids eye exams, flagging diseases for earlier treatment.
- Useful for improvement in diagnostic accuracy

  Deep learning trumps experts in spotting eye disease from images.
- Useful in Remote eye care & telehealth

  Deep learning aids remote eye disease diagnosis in underserved areas.

1.5. LIMITATIONS

- **Large dataset**

  To train the Deep Learning(DL) model, we must use the larger datasets, which make the process of training slower.

- **Time Consuming**

  Since, this is a process involving a lot of data which changes regularly over time it becomes a time-consuming process to predict results.

- **Low efficiency**

  Like most of the process involving Deep Learning, efficiency to be very good takes a lot of time to train and predict the results. The same is the case in this prediction.

- **Expensive on Machines**

  Since, all the Deep Learning(DL) models use a lot of data to work with it often requires expensive resources to perform predictions.

- **High Complexities**

  Since, the DL model mostly works on data it requires complex data, models, algorithms to work with.

CHAPTER - 2

LITERATURE SURVEY

The field of automated fundus disease recognition has witnessed significant progress, marked by diverse methodologies and the utilization of deep learning techniques.

According to J. Wang, L. Yang, Z. Huo, W. He and J. Luo, "Multi-Label Classification of Fundus Images with EfficientNet," in IEEE Access, vol. 8, pp. 212499- 212508, 2020, *doi: 10.1109/ACCESS.2020.3040275* aims to identify and detect multiple eye diseases in fundus images, using information of both eyes for improved accuracy. It utilizes a deep learning approach for feature extraction from the images and combines those features in a new way. While claiming high success compared to other methods, the researchers acknowledge areas for improvement, such as further explaining how they combine the information from both eyes and providing more detailed analysis of their results. They also recognize the need to compare their approach to newer methods in the field.

According to I, N., Li, T., Hu, C., Wang, K., Kang, H. (2021). A Benchmark of Ocular Disease Intelligent Recognition: One Shot for Multi-disease Detection. In: Wolf, F., Gao, W. (eds) Benchmarking, Measuring, and Optimizing. Bench 2020. Lecture Notes in Computer Science(), vol 12614. Springer, Cham. *https://doi.org/10.1007/978-3-030-71058-3_11* presents a benchmark for ocular disease intelligent recognition, emphasizing multi-disease detection. It introduces a standardized evaluation framework to assess algorithm performance, aiming to advance automated diagnosis and treatment of eye conditions.

Published in Springer's Lecture Notes in Computer Science series, the study contributes to healthcare optimization efforts. The benchmark provides valuable insights for researchers and practitioners, facilitating advancements in machine learning-based eye disease diagnosis and enhancing patient care outcomes.

According to Muchuchuti . S, Viriri, "Retinal Disease Detection Using Deep Learning Techniques: A Comprehensive Review", J. Imaging 2023, 9, 84. *https://doi.org/10.3390/jimaging9040084* This paper examines using deep learning (CNNs, RNNs, GANs) to detect retinal diseases like myopia and AMD. It covers data, preprocessing, and model evaluation. Deep learning's potential for feature selection and limited data is acknowledged. Further exploration of data augmentation, detailed result analysis, and comparisons with newer, superior methods on the ODIR dataset are recommended

According to Alzubaidi, L.; Al-Amidie, M.; Al-Asadi, A.; Humaidi, A.J.; Al-Shamma, O.; Fadhel, M.A.; Zhang, J.; Santamaría, J.; Duan, Y. Novel Transfer Learning Approach for Medical Imaging with Limited Labeled Data. Cancers 2021, 13, 1590. *https://doi.org/10.3390/cancers13071590***,** This paper proposes a transfer learning approach for medical image analysis with limited labeled data. It utilizes a pre-trained model on vast unlabeled data and fine-tunes it with a smaller set of labeled medical images. Furthermore, it introduces a new DCNN architecture. While achieving improved performance on breast and skin cancer tasks compared to existing methods, the paper could benefit from exploring data augmentation techniques, deeper result analysis, and comparisons with recent advancements in the field.

CHAPTER - 3

ANALYSIS

Existing model:

Deep learning models are showing promise in ocular disease detection. Existing models use convolutional neural networks (CNNs) trained on large datasets of labeled retinal images. This allows them to identify subtle signs of diseases like glaucoma or diabetic retinopathy. While achieving high accuracy, challenges include limited data availability, interpretability of the models, and ensuring generalizability to unseen cases.

Proposed System:

In this proposed method, a deep learning model named *EfficientNetB7* is mainly used in this model, but for comparison we also used *InceptionV3, ResNet50, VGG-16*. Among these models, *EfficientNet* gave better and higher results with more accuracy for the prediction of disease. In this project, transfer learning techniques were employed using popular convolutional neural network architecture called *EfficientNet* to classify eye diseases. In the context of image classification, transfer learning involves leveraging a pre-trained CNN model on a large dataset (like ImageNet) and fine-tuning it on the Ocular 5K dataset. This approach allows the model to learn relevant features from the large dataset and adapt them to the specific classification task at hand

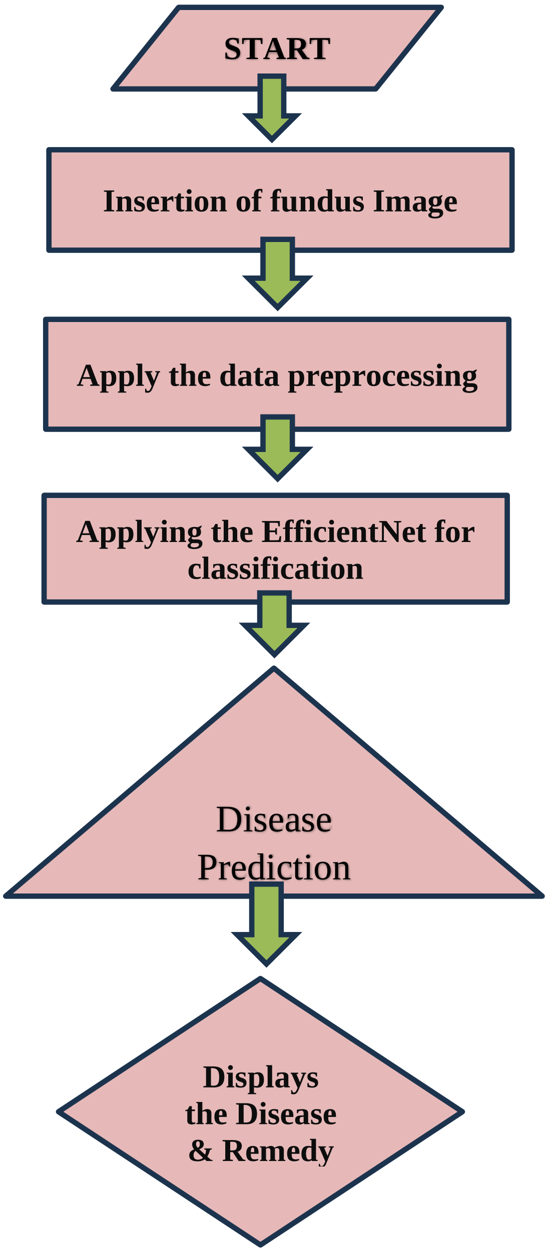
Problem Statement:

The problem addressed is to classify the ocular diseases accurately with greater accuracy in determining the true symptoms of the person. The implementation of this can be achieved using the combined approach of Transfer Learning and CNN. The Transfer Learning aims to decrease model training time, improve performance, and then retrain. It improves the accuracy of diagnosis but also simplifies the analysis process, helping to increase the efficiency of treatment and improve the patient's outcome effect.

Objectives

- To predict an eye disease among various fundus images using deep learning.
- To predict whether the ocular disease occurs or not given various aspects of human life
- To predict which type of disease has occurred in the eye.

Flow Diagram:



Figure(3.1): Flow of Implementation

CHAPTER – 4
DESIGN

4.1. UML DIAGRAMS:

UML stands for Unified Modelling Language. UML is a standardized general-purpose modelling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object-oriented computer software. In its current form UML is comprised of two major components: A Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modelling Language is a standard language for specifying, Visualization, Constructing and documenting the artefacts of software system, as well as for business modelling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modelling of large and complex systems. The UML is a very important part of developing objects-oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

4.2. GOALS:

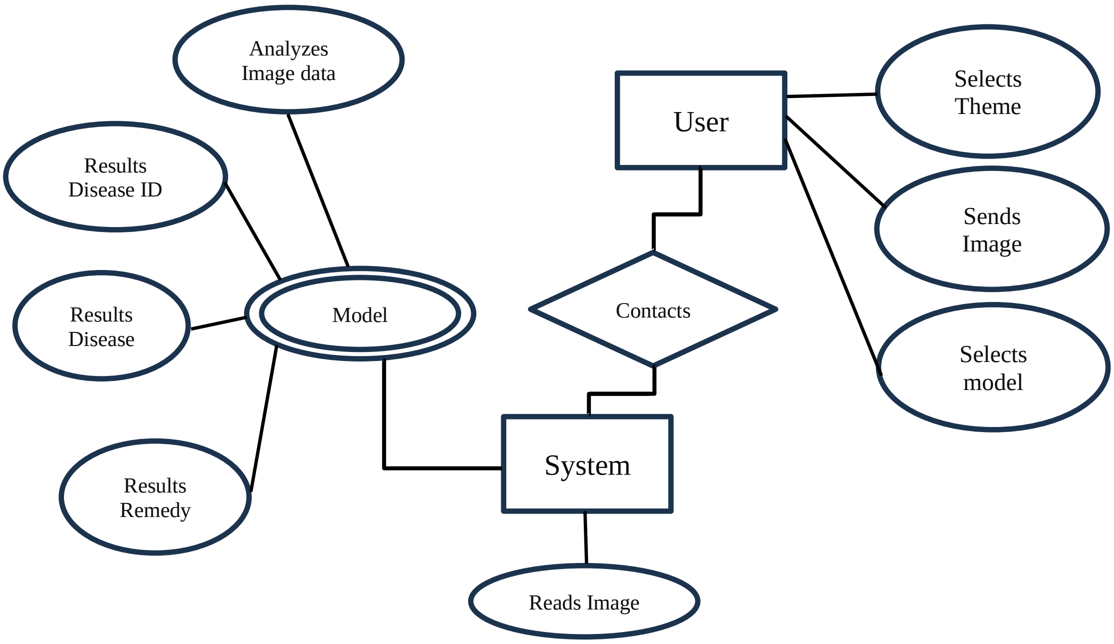
The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modelling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of programming languages and development process.

4. Provide a formal basis for understanding the modelling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns, and components.
7. Integrate best practices.

4.3. ER DIAGRAM:

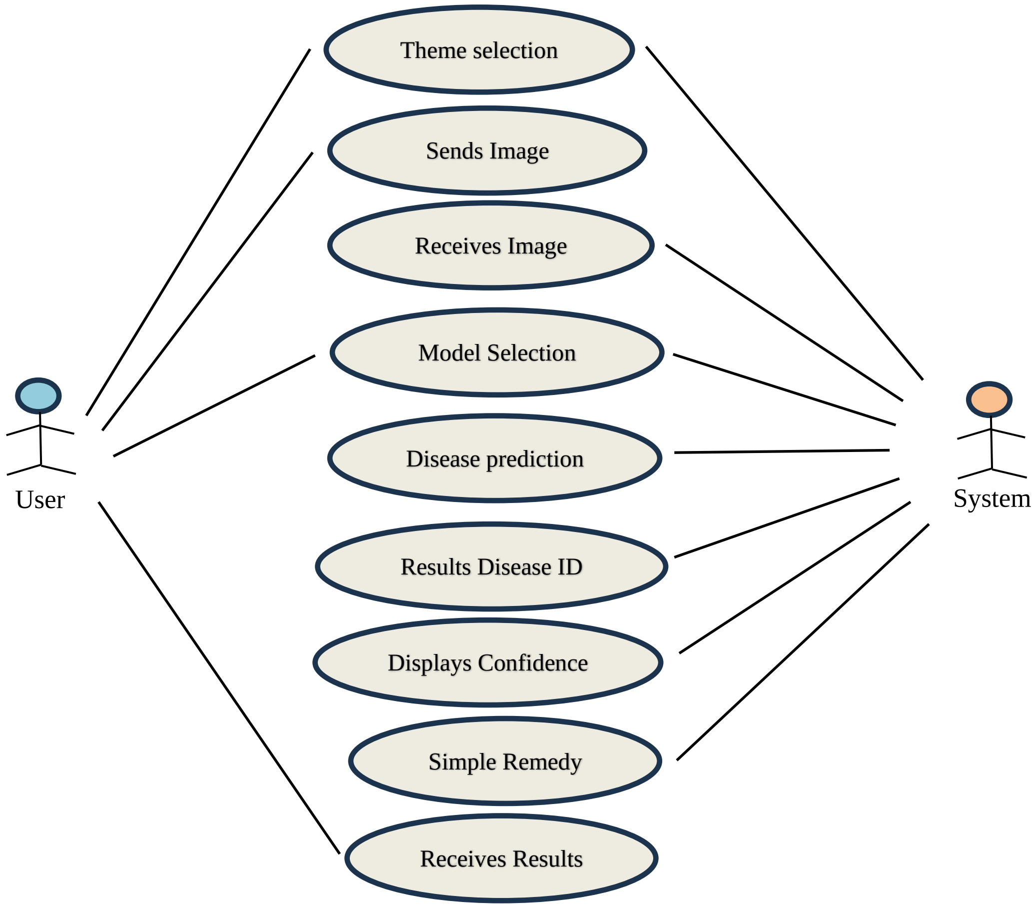
An Entity–relationship model (ER model) describes the structure of a database with the help of a diagram, which is known as Entity Relationship Diagram (ER Diagram). An ER model is a design or blueprint of a database that can later be implemented as a database. The main components of E-R model are: entity set and relationship set. An ER diagram shows the relationship among entity sets. An entity set is a group of similar entities and these entities can have attributes. In terms of DBMS, an entity is a table or attribute of a table in database, so by showing relationship among tables and their attributes, ER diagram shows the complete logical structure of a database. Let's have a look at a simple ER diagram to understand this concept.



Figure(4.1): ER Diagram

4.4. USE CASE DIAGRAM:

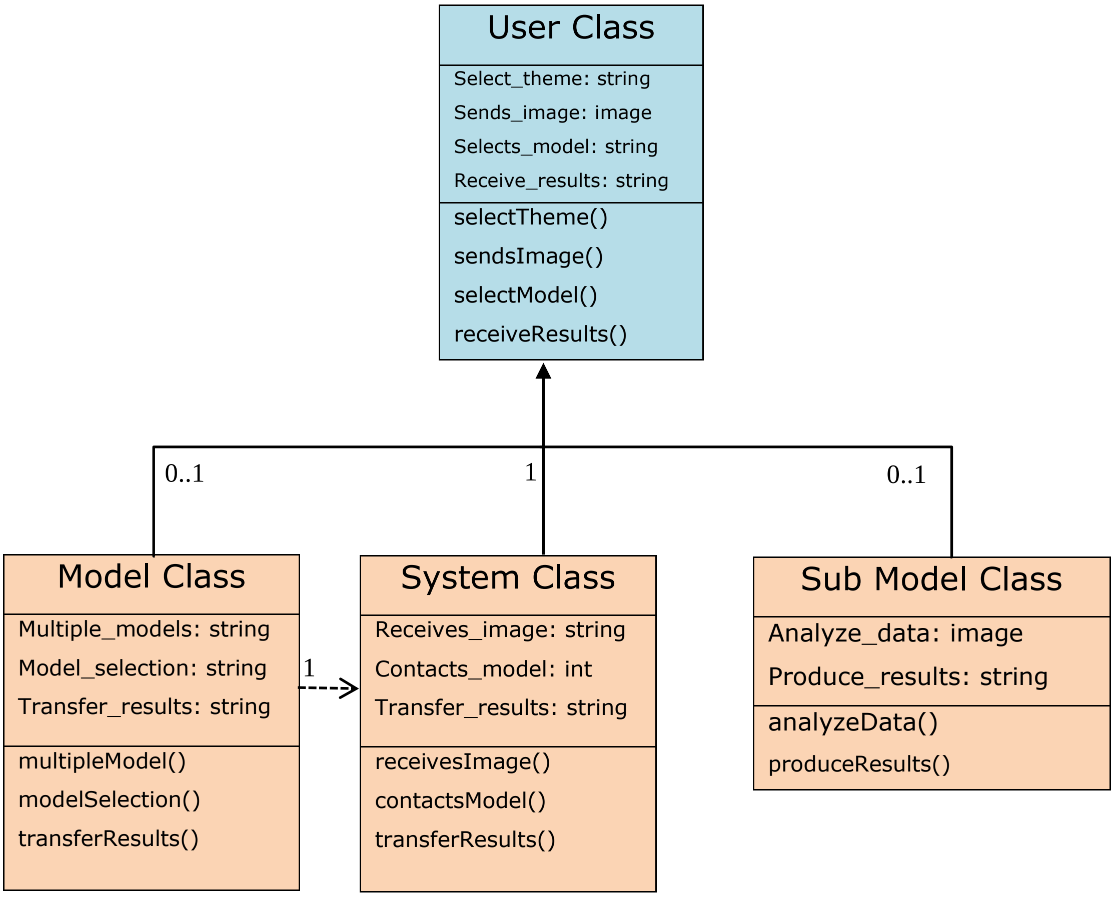
A use case diagram in the Unified Modelling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.



Figure(4.2): Use – Case Diagram

4.5. CLASS DIAGRAM:

In software engineering, a class diagram in the Unified Modelling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

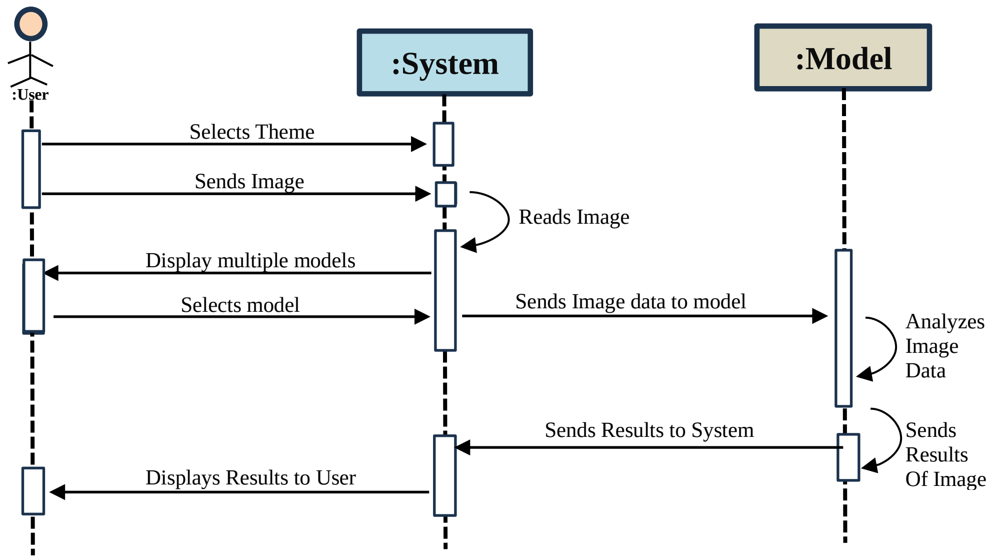


Figure(4.3): Class Diagram

The above class diagram represents the user, which is main class of the UML diagram. This class will communicate with the *system class*, which has the connection with *model class.* The *model class* has the connection with *Sub Model Class,* which consists of 4 different types of models.

4.6. SEQUENCE DIAGRAM:

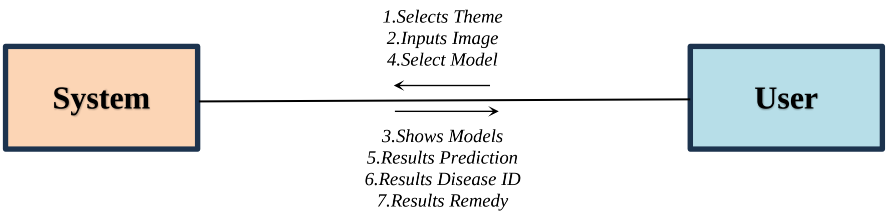
A sequence diagram in Unified Modelling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.



Figure(4.4): Sequence diagram

4.7. COLLABORATION DIAGRAM:

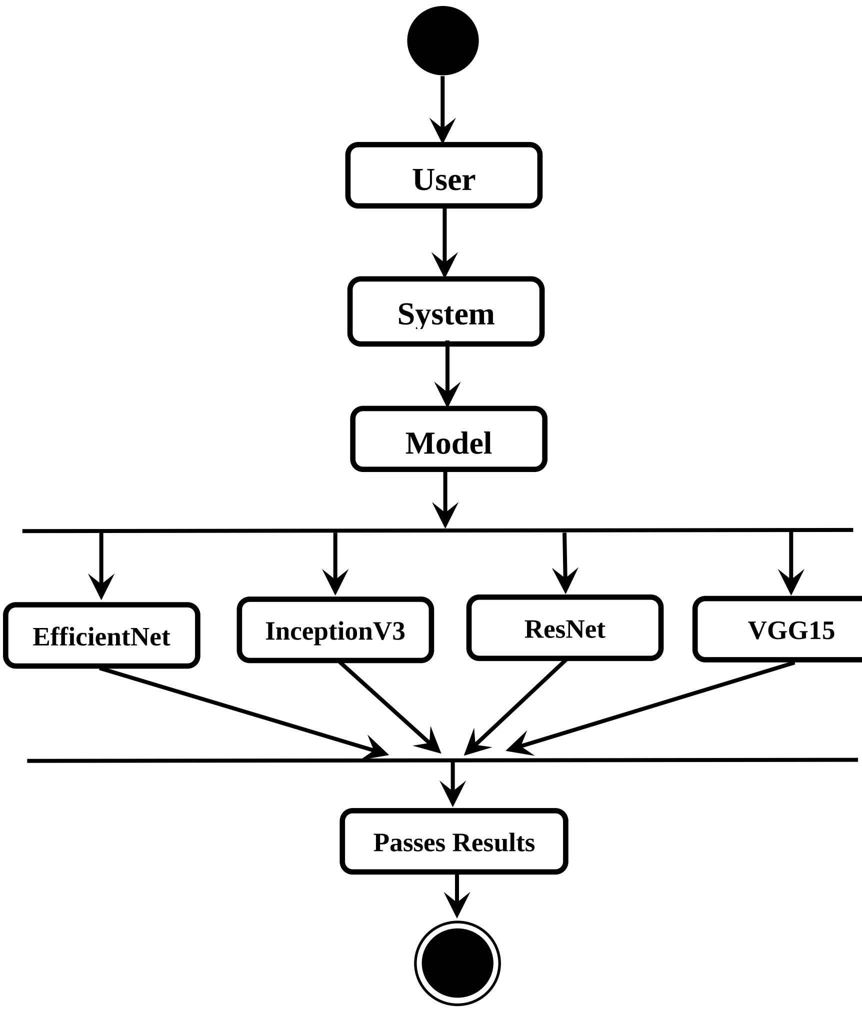
In collaboration diagram the method call sequence is indicated by some numbering technique as shown below. The number indicates how the methods are called one after another. We have taken the same order management system to describe the collaboration diagram. The method calls are similar to that of a sequence diagram. But the difference is that the sequence diagram does not describe the object organization whereas the collaboration diagram shows the object organization.



Figure(4.5): Collaboration Diagram

4.8. ACTIVITY DIAGRAM:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modelling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.



Figure(4.6): Activity Diagram

CHAPTER - 5
IMPLEMENTATION

SYSTEM REQUIREMENTS:

System requirements gives the idea about what are the necessary things that are needed for proposed system, which plays a very important role in the development of any system. This deals with what are hardware and software components that are needed for system.

- *Visual Studio*(VS Code) is the platform to run the project.

Essential Libraries:

- *Python (3.6 or later):* The core programming language for building the deep learning model. Download from https://www.python.org/downloads/.
- *NumPy:* Provides numerical computing capabilities for data manipulation. Install using pip install NumPy.
  - *NumPy Arrays*: Fundamental data structure in deep learning. They store large collections of numerical values (usually floats) in a multidimensional grid format, ideal for representing image pixels. Imagine a 3D grid where each element holds a pixel intensity value.
- *Pandas:* Offers data structures (DataFrames) and data analysis tools. Install using pip install pandas.
  - *Pandas DataFrames*: Two-dimensional labeled data structures combining columns (like image paths, labels) and rows (representing individual data points). Useful for organizing and manipulating your dataset.

Deep Learning Libraries:

- *TensorFlow/Keras:* Popular deep learning frameworks for building and training the Convolutional Neural Network (CNN) model. Choose one:

- *TensorFlow:* Install using pip install tensorflow.
- *Keras:* Install using pip install keras. You might also need TensorFlow as a backend for Keras.

IMPLEMENTATION:

Implementation is the important stage of the project where the proposed system Design turned out into a working system. Thus, it can be the most crucial stage in achieving a successful new system and gives the user the confidence that the new system will work and be effective.

The Implementation stage involves:

- DATA PREPARATION AND PREPROCESSING
- DATA AUGUMENTATION AND SPLITTING
- TRANSFER LEARNING
- MODEL TRAINING
- LOSS FUNCTION

A. DATA PREPARATION AND PREPROCESSING:

Initially, we import necessary libraries which includes pandas for data manipulation, Tensorflow's Keras API for building model and various layers from Keras to define neural network architecture. We read the dataset from a CSV file which contains information like image labels, path, and class labels. The relevant data is organized into dictionary and then converted into DataFrame for further processing. During this stage observed data which is solved using Data Augmentation.

B. DATA AUGUMENTATION AND SPLITTING

During the second phase of model development, to address the challenge posed by data imbalance we use a technique called Data Augmentation. Using Keras Image Data Generator, we rescaled pixel values to a range of 0 to 1, applied shear transformations, introduced zooming, horizontal and vertical flipping. This approach aimed to diversify our dataset, enhance robustness, generalization capabilities mitigate overfitting. The Image Data Generator then employed to generate batches of images for these subsets. With a target size of (224, 224) and batch size of 32, generator operated in categorical mode. A validation subset was created constituting 20% of entire dataset, ensuring a robust evaluation during training. To seamlessly integrate our data generators into the TensorFlow deep learning pipeline, we constructed TensorFlow datasets, namely train_ds and val_ds. These datasets were generated from the respective data generators, ensuring compatibility with the subsequent model training process.

C. TRANSFER LEARNING

Transfer learning involves utilizing a pre-existing model as a starting point to solve a related new problem. Instead of training a deep neural network from scratch we use models which are already trained on ImageNet dataset. Using this reduces training time, improve performance, generalization of model and prevents our model from overfitting. We use this technique in our model.

D. MODEL TRAINING

Choosing EfficientNetB7[15], a convolutional neural network architecture as the base model available in the TensorFlow Keras applications module, which uniformly scales model depth, width, and resolution. The decision was made to utilize the Adam optimizers[16] for adjusting weights during the training process, coupled with the selection of categorical cross-entropy as the chosen loss function. This is called fine-tuning which involves changing the model's weights and parameters. The training dataset, encapsulated in train_ds, was fed through the model with a specified number of steps per epoch, as per the batch size. The training process was designed to progressively fine-tune the model's parameters through iterative adjustments and optimize its ability to capture intricate patterns for accurate ocular disease classification.

E. LOSS FUNCTION

After detail analysis of weighting loss functions, we decided to utilize the following categorical cross entropy as the loss function. The Equation(1) defines it, with representing y_i as actual label and y_i^ as predicted label,

$$CE = - \sum_{i=1}^{i=N} y\_true_i \,.\, \log(y\_pred_i)$$

$$CE = - \sum_{i=1}^{i=N} y_i \,.\, \log(\hat{y}_i)$$

Equation (1)

After loss function , using Adam as the optimization function to optimize the learning parameters in the model. Adam converges well when compared to others due to the gradient which is updated every time. Similar to momentum, adam maintains exponential decay average of the previous gradient which improves training speed. Adam is defined in the Equation (2)

$$m_t = \beta_1 * m_{t-1} + (1 - \beta_1) * g_t$$

$$v_t = \beta_2 * v_{t-1} + (1 - \beta_2) * g_t^2$$

$$\theta = \theta - (\alpha * m_t / \sqrt{(v_t + \varepsilon)})$$

Equation (2)

where β_1 and β_2 are hyper-parameters, α is Step size for optimization, g_t is the gradient, m_t and v_t are the *first* and *second* momentum. The complete data flow diagram is shown in Figure(5.2).

Models:

EfficientNet:

EfficientNet is a convolutional neural network (CNN) architecture designed for deep learning tasks. It balances three key aspects to achieve high accuracy:

1. **Depth:** It scales the network depth efficiently, allowing for learning complex patterns.
2. **Width:** It scales the network width proportionally to depth, maintaining representational power.
3. **Resolution:** It adjusts image resolution based on the model size, optimizing resource usage.

This balancing act lets EfficientNet achieve high accuracy while being faster and more resource-friendly than traditional CNNs.

Applications:

EfficientNetB7, a powerful deep learning architecture, tackles demanding computer vision tasks like image classification, object detection, and semantic segmentation. Its efficiency makes it suitable for resource-constrained environments or large datasets.

VGG-16:

VGG-16 is a convolutional neural network (CNN) architecture known for its depth. It uses repeated stacks of 3x3 convolutional layers with small filters, followed by pooling layers, to extract features from images. While effective, its depth (19 layers) can be computationally expensive. VGG-16 is pre-trained on a massive ImageNet dataset, allowing it to be fine-tuned for various image recognition tasks. However, newer architectures like EfficientNet achieve similar or better accuracy with less computational resources.

Application:

VGG-16 can identify objects in pictures after being trained on a large dataset.

ResNet50:

ResNet, short for Residual Network, is a type of convolutional neural network (CNN) architecture designed to address the vanishing gradient problem that can hinder training of very deep networks. It utilizes skip connections that bypass some layers, allowing the network to learn the identity function (simply copying the input) in addition to more complex transformations. This enables ResNet to train much deeper networks compared to traditional CNNs, leading to improved performance on various image recognition tasks. However, ResNets can still be computationally expensive due to their depth.

Application:

ResNet50 can recognize objects in images even with complex backgrounds.

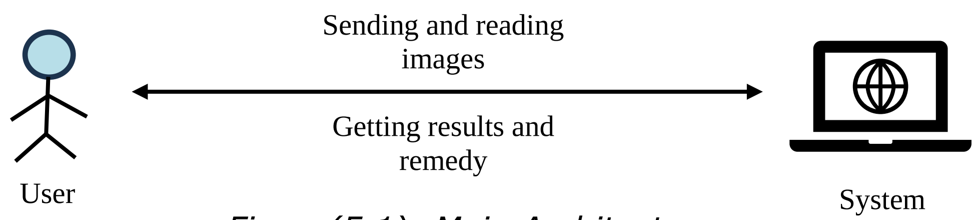
InceptionV3:

InceptionV3 is a convolutional neural network (CNN) architecture known for its efficiency and strong performance in image recognition tasks. It utilizes inception modules, which combine filters of various sizes (1x1, 3x3, 5x5) within a single block. This allows the network to capture features at different scales simultaneously. Additionally, InceptionV3 employs techniques like dimensionality reduction and grid-based inception to optimize resource usage. While pre-trained on a massive ImageNet dataset, InceptionV3 can be fine-tuned for various tasks, including ocular disease detection by analyzing retinal images.

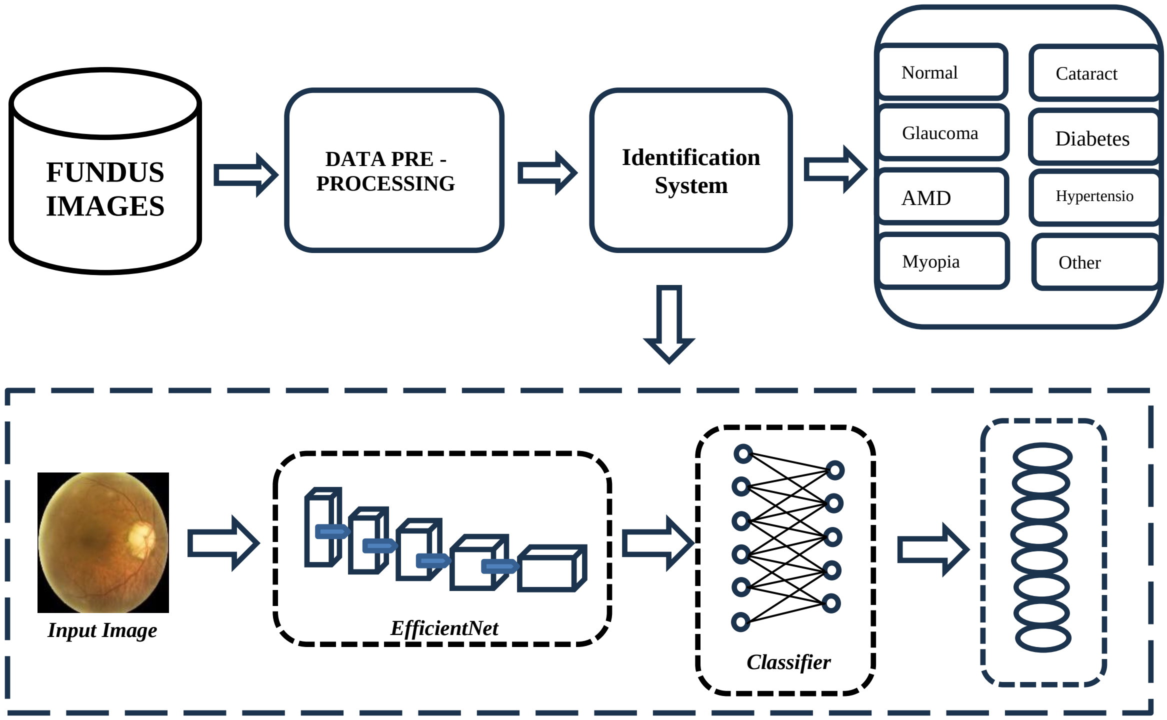
Application:

InceptionV3 tackles image recognition tasks efficiently, from spotting objects in pictures to identifying diseases in medical scans like retinal images.

Architecture:



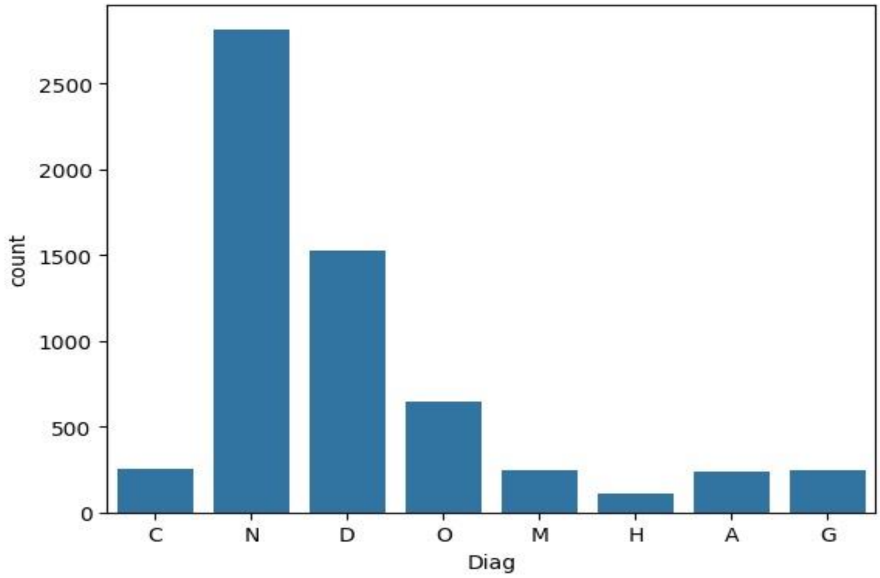
Figure(5.1): Main Architecture



Figure(5.2): Model architecture

DATASET:

The Ocular Disease Intelligent Recognition (ODIR) dataset is "real" patient data gathered by Shang gong Medical Technology from various healthcare facilities and hospitals across China. The researchers categorized the patient data into 8 groups: normal(N), cataract(C), glaucoma (G), diabetes (D), hypertension (H), pathological myopia(M), age-related macular degeneration (A), and other abnormalities or diseases. The visual representation of samples within the dataset is given below figure.



Figure(5.3): The Count of images in the dataset

Procedure:

This procedure given below will explains the process that has done in each step of the model architecture.

1. Fundus Images

Fundus images are photographs of the inside of the eye[8], specifically the back of the eye where the retina is located. The retina is a light-sensitive layer containing millions of photoreceptor cells that play a crucial role in vision. Fundus imaging[5] is a non-invasive and painless procedure that allows ophthalmologists to examine the retina and other structures within the eye for signs of disease.

Here's how fundus images are instrumental in ocular disease recognition:

- **Visualization of Internal Structures:** Fundus images provide a detailed view of the retina, macula (responsible for central vision), optic nerve, and blood vessels. This allows ophthalmologists to detect abnormalities that might be indicative of various eye diseases.

- **Early Detection of Conditions:** Many eye diseases, such as diabetic retinopathy, glaucoma, and age-related macular degeneration, can develop without noticeable symptoms in the early stages[6]. Fundus images can reveal subtle changes in the retina that might signal the beginning of these diseases, enabling early intervention and potentially preventing vision loss.
- **Monitoring Disease Progression:** Fundus imaging allows ophthalmologists to track the progression of existing eye diseases over time. By comparing images taken at different visits, they can assess the effectiveness of treatment and make necessary adjustments.
- **Identification of Risk Factors:** Certain characteristics observed in fundus images, like blood vessel narrowing or hemorrhages, can indicate risk factors for specific eye diseases or systemic conditions like diabetes or hypertension.

In conclusion, fundus images are a vital tool in ocular disease recognition. They offer a window into the inner workings of the eye, enabling ophthalmologists to detect, diagnose, monitor, and potentially prevent vision loss from various eye conditions.

2. Data pre-processing

Data preprocessing is the crucial first step in any deep learning project. It involves cleaning and manipulating raw data to prepare it for use in a deep learning model. Imagine a deep learning model as an algorithm that learns from data to make predictions or classifications. Just like a student needs clear and concise instructions to learn effectively, a deep learning model needs clean and formatted data to train on.

Here is a breakdown of the common steps involved in data preprocessing:

1. **Data Acquisition:** This involves collecting data from various sources relevant to the deep learning problem you're trying to solve.
2. **Data Cleaning:** Real-world data often contains inconsistencies, missing values, or errors. Data cleaning involves fixing these issues to ensure the data is accurate and consistent.
3. **Data Integration:** If your data comes from multiple sources, you might need to combine them into a single format for analysis.
4. **Data Transformation:** This step involves scaling or normalizing the data to ensure all features are on a similar scale. It might also involve encoding categorical variables into numerical values that the deep learning model can understand.
5. **Handling Missing Values:** Data can have missing entries for various reasons. Data preprocessing involves deciding how to handle these missing values, such as removing rows with missing data, imputing missing values with estimates, or using statistical methods to account for them.

3. Identification System

An identification system refers to a general application that uses a trained model to classify or categorize new, unseen data points. These systems can be applied to various tasks, including:

- **Image recognition:** Classifying objects or scenes in images, like identifying different dog breeds or distinguishing between healthy and diseased tissue in medical scans.
- **Spam filtering:** Classifying emails as spam or not spam based on their content.
- **Sentiment analysis:** Identifying the sentiment (positive, negative, neutral) expressed in text data like social media posts or customer reviews.
- **Fraud detection:** Classifying financial transactions as fraudulent or legitimate.

Here's a breakdown of how an identification system typically works:

1. **Model Training:** The system is trained on a large dataset[4] of labeled examples. Each example consists of an input (e.g., an image, a piece of text, a set of features) and a corresponding label (e.g., the object in the image, the sentiment of the text, the category of the transaction). During training, the model learns to identify patterns and relationships between the input data and the labels.
2. **Identification:** Once trained, the system can be used to identify new, unseen data points. The system takes an unseen input, processes it through the trained model, and outputs a label prediction based on the learned patterns.

4. EfficientNet

EfficientNet is a family of convolutional neural network (CNN) architectures specifically designed to achieve high accuracy while being computationally efficient. This translates to faster training times and the ability to run on devices with lower processing power.

In the realm of ocular disease recognition, EfficientNet shows promise as a powerful tool for building accurate identification systems.

Here's how EfficientNet can be advantageous in ocular disease recognition:

- **High Accuracy:** EfficientNet architectures have achieved state-of-the-art performance on various image recognition tasks. This translates to potentially high accuracy in classifying ocular diseases from retinal fundus images.
- **Computational Efficiency:** Compared to traditional CNN architectures, EfficientNet models achieve similar or better accuracy with fewer parameters and FLOPs (Floating-point Operations). This efficiency makes them suitable for deployment on devices with limited resources, potentially enabling the development of mobile applications for eye disease screening.
- **Faster Training:** The reduced complexity of EfficientNet models translates to faster training times compared to bulkier CNNs. This can be crucial for researchers and developers working on ocular disease recognition systems, allowing for quicker iteration and improvement.

Here are some examples of how EfficientNet is being applied in ocular disease recognition:

- **Research Studies:** Several studies have explored the use of EfficientNet for classifying various eye diseases from fundus images. These studies demonstrate promising results, with EfficientNet achieving competitive or even surpassing performance compared to other CNN architectures.
- **Potential Applications:** The efficiency of EfficientNet makes it suitable for developing mobile applications for eye disease

screening. Such applications could be used in telemedicine settings or by healthcare workers in remote areas to provide preliminary assessments for eye conditions.

Overall, EfficientNet offers an exciting advancement in the field of ocular disease recognition. Its ability to balance accuracy with efficiency paves the way for developing new tools and applications that can potentially improve access to eye care and early detection of eye diseases.

5. Classifier

In ocular disease recognition, a classifier is the heart of the system, acting like a decision-maker that analyzes retinal images and determines the presence or absence of a specific eye disease. It's essentially an algorithm within the overall system that performs the classification task.

Here's a breakdown of how classifiers function in ocular disease recognition:

1. **Model Training:** The classifier is trained on a large dataset of labeled retinal images. Each image is labeled with the corresponding disease it represents (e.g., healthy, glaucoma, diabetic retinopathy). During training, the classifier learns to identify patterns and features within the images that are indicative of different eye diseases.
2. **Feature Extraction:** Classifiers, particularly those based on deep learning architectures like convolutional neural networks (CNNs), excel at extracting features from the images. These features can be intricate details like blood

vessel patterns, optic nerve characteristics, or the presence of lesions in the retina.

3. **Classification:** Once trained, the classifier can analyze a new, unseen retinal image. It extracts features from the image and compares them to the patterns learned during training. Based on this comparison, the classifier outputs a prediction about the presence or absence of a specific disease. The prediction can be a simple binary classification (healthy vs. diseased) or a multi-class classification (healthy, glaucoma, diabetic retinopathy, etc.).

Common Classifier Types in Ocular Disease Recognition:

- Convolutional Neural Networks (CNNs):
- Support Vector Machines (SVMs)
- Random Forests

CHAPTER - 6
EXECUTION PROCEDURE AND TESTING

STEPS FOR EXECUTION:

1. Import all the Libraries/packages.
2. Load the dataset.
3. Perform exploratory data analysis.
4. Pre-process the datasets.
5. Remove the null values.
6. Check the unbalanced data.
7. Split the dataset.
8. Train the dataset using Transfer Learning with different models.
9. The models are:
    a. EfficientNetB7
    b. InceptionV3
    c. VGG – 16
    d. ResNet50
10. The model is used for predictions of performance from the data.

- Note: Only the *EfficientNet* is the main model, remaining are only the comparison models.

TESTING:

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of tests. Each test type addresses a specific testing requirement.

TYPES OF TESTS:

Unit testing:

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration testing:

Integration tests are designed to test integrated software components to determine if they run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Functional test:

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:
Valid Input : identified classes of valid input must be accepted.
Invalid Input : identified classes of invalid input must be rejected.
Functions : identified functions must be exercised.
Output : identified classes of application outputs must be exercised.
Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

System test:

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

White Box Testing:

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

Black Box Testing:

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box. you cannot "see" into it. The test provides inputs and responds to outputs without considering how the software works.

Unit Testing:

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test Objectives:

- Image
- Dataset must be balanced. (If it is unbalanced)
- The results must contain high accuracy.
- In the entire dataset, training is done on 80% and remaining 20% of dataset should be used for testing.

CHAPTER – 7
RESULTS AND PERFORMANCE EVALUATION

RESULTS:

In this application, we have successfully created a project for the prediction of Ocular disease using Deep Learning techniques. The module consists of system and user. The system captures the data and gives the results for each model. The performance evaluation can be done based on the four metrics

- Accuracy
- Loss
- F1 Score
- Precision

1.Accuracy:

Accuracy reflects how close a measurement is to a known or accepted value.

The calculation is carried out using below formula:

Accuracy = (True positives + True Negatives) / (True positives + True negatives + False positives + False negatives)

2.Loss:

Model loss in deep learning quantifies the disparity between predicted and true values during training, facilitating optimization by backpropagation. It serves as a crucial metric for evaluating and refining model performance.

3. F1 Score:

F1 score is an alternative deep learning evaluation metric that assesses the predictive skill of a model by elaborating on its class-wise performance rather than an overall performance as done by accuracy. F1 score combines two competing metrics- precision and recall scores of a model.

Precision=TP/(TP+FP)

Formula of F1 Score:

*F1 Score= (2*Precision*Recall)/(Precision + Recall)*

4.Precision:

Precision evaluates the fraction of correctly classified instances or samples among the ones classified as positives. Thus, the formula to calculate the precision is given by:

Precision = True positives/ (True positives + False positives) = TP/(TP + FP)

After the complete model training, the progression of training and validation and their loss over multiple epochs of the architectures are shown in Figure(1) and Figure(2). The F1 Score and Precision graphs are depicted in the Figure(3) and Figure(4).

PERFORMANCE EVALUATION:

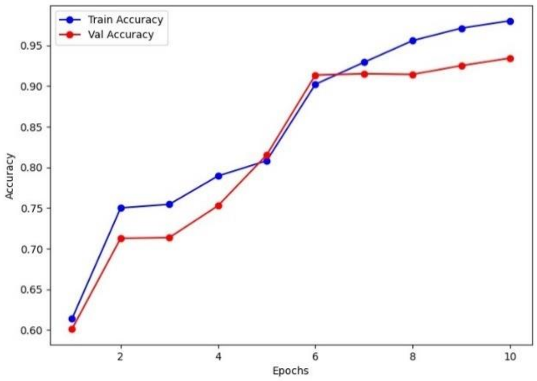


Fig1: Progression of training and validation over multiple epochs

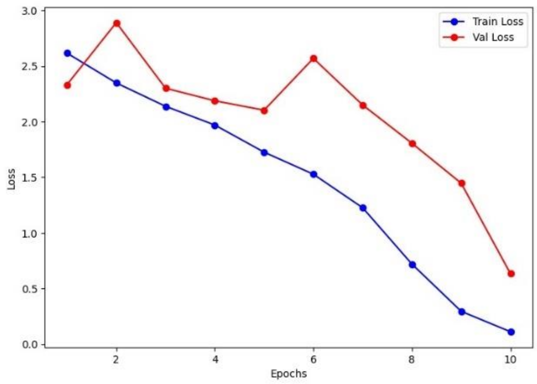


Fig2: Progression of training and losses over multiple epochs

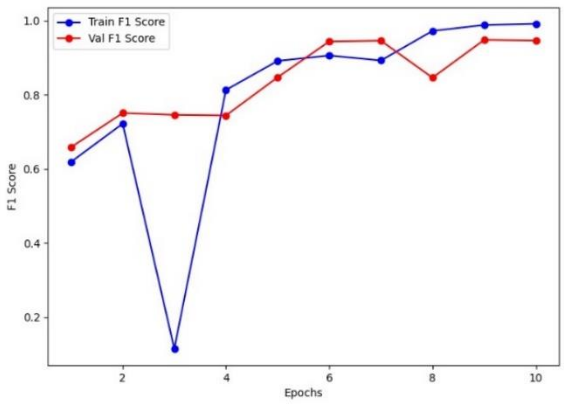


Fig3: Progression of training and validation F1 score over multiple epochs

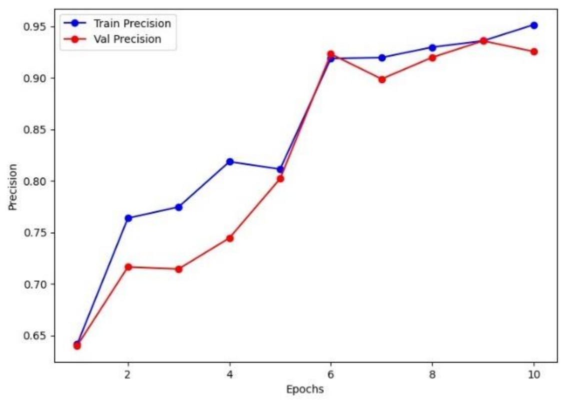


Fig4: Progression of training and validation precision over multiple epochs

CHAPTER – 8
CONCLUSION AND FUTURE WORKS

CONCLUSION:

The main purpose of our research is to analyze the efficiency of deep learning architectures in ocular disease classification. We introduce a novel classification method for ocular diseases, combining a fusion of Transfer Learning and CNN. Our goal was to achieve superior results through this innovative approach and we delivered promising results by delivering 92% accuracy which outperformed the existing methods and techniques. Upon examining the results, it becomes evident that when a more extensive dataset is accessible, the implementation of deep neural networks in healthcare is preferable. However, despite promising results, practical implementation remains a significant challenge. Within the ODIR-2019 dataset, the 'O' label, denoting 'Other Diseases,' encompasses a diverse range of infrequent fundus diseases. Several challenges remain in the field, including the limitations of available datasets, class imbalance issues, and ensuring model can adapt to new situations.

FUTURE WORK:

We can collect proper class balanced and large datasets which can improve model performance and accuracy. Along with those abnormal diseases from the fundus images involves further research. In order to implement deep learning methods in medical field should consider various aspects and visualization is necessary.

APPENDIX

PROGRAM LISTING/CODE

```
import pandas as pd
import numpy as np
from tensorflow.keras.preprocessing.image import
ImageDataGenerator
from tensorflow.keras.layers import Dense, Conv2D, MaxPooling2D,
Flatten
from tensorflow.keras.models import Sequential
data = pd.read_csv(r'dataset/full_df.csv')
data.head()
data['filename'] = data['filename'].apply(lambda x:
f'dataset/preprocessed_images/{x}')
data['class'] = data['labels'].apply(lambda x: x[2])
main_data = {
    'filename' : list(data['filename']),
    'class' : list(data['class'])
}
main_df = pd.DataFrame(main_data)
main_df.head()
main_df['class'].unique()

import os
import tensorflow as tf

# Function to perform image augmentation
def augment_image(image):
    # Define augmentation operations
    image = tf.image.random_flip_left_right(image)
    image = tf.image.random_flip_up_down(image)
    image = tf.image.random_brightness(image, max_delta=0.2)
```

```
    image = tf.image.random_contrast(image, lower=0.5,
upper=1.5)
    image = tf.image.rot90(image, k=tf.random.uniform(shape=[],
minval=0, maxval=4, dtype=tf.int32))
    image = tf.image.random_translate(image,
translations=[tf.random.uniform(shape=[], minval=-50,
maxval=50, dtype=tf.int32),

tf.random.uniform(shape=[], minval=-50, maxval=50,
dtype=tf.int32)])
    image = tf.image.random_hue(image, max_delta=0.2)
    image = tf.image.random_saturation(image, lower=0.5,
upper=1.5)
    image = tf.clip_by_value(image, 0.0, 1.0)
    return image

# Function to generate augmented images and save them
def generate_augmented_images(input_dir, output_dir,
um_generated_images):
    # Create output directory if it doesn't exist
    if not os.path.exists(output_dir):
        os.makedirs(output_dir)

    # Get list of image files in the input directory
    image_files = [f for f in os.listdir(input_dir) if
os.path.isfile(os.path.join(input_dir, f))]

    for filename in image_files:
        # Load image
        image_path = os.path.join(input_dir, filename)
        image = tf.io.read_file(image_path)
        image = tf.image.decode_image(image, channels=3)
```

```
        image = tf.cast(image, tf.float32) / 255.0

        # Apply augmentation and save generated images
        for i in range(num_generated_images):
            augmented_image = augment_image(image)

            # Save augmented image
            output_filename = os.path.splitext(filename)[0] +
f'_augmented_{i}.jpg'
            output_path = os.path.join(output_dir, output_filename)
            augmented_image =
tf.image.convert_image_dtype(augmented_image, tf.uint8)
            augmented_image =
tf.image.encode_jpeg(augmented_image)
            tf.io.write_file(output_path, augmented_image)

input_directory = 'input_images'
output_directory = 'augmented_images'
num_generated_images = 5
generate_augmented_images(input_directory, output_directory,
num_generated_images)



dg = ImageDataGenerator(
    rescale = 1./255,
    zoom_range = 0.2,
    horizontal_flip = True,
    vertical_flip = True,
    validation_split = 0.2
)
train_dg = dg.flow_from_dataframe(
    main_df,
```

```
    target_size = (224, 224),
    batch_size = 32,
    subset = 'training',
)

val_dg = dg.flow_from_dataframe(
    main_df,
    target_size = (224, 224),
    batch_size = 32,
    subset='validation'
)
from tensorflow.keras.applications import EfficientNetB7
base_model4 = EfficientNetB7(
    weights='imagenet',
    input_shape = (224, 224, 3),
    include_top = False
)
for layer in base_model4.layers:
    layer.trainable = False
model4 = Sequential([
    base_model4,
    Flatten(),
    Dense(128, activation='relu'),
    Dense(32, activation='relu'),
    Dense(8, activation='softmax')
])
model4.compile(loss='categorical_crossentropy', optimizer='Adam',
metrics=['accuracy'])
history3 = model4.fit(train_dg, validation_data=val_dg,
epochs=10)
model4.save('efficientnet.h5')
```

```
import matplotlib.pyplot as plt
epochs = range(1, 11)
accuracy = [0.6137, 0.7501, 0.7545, 0.7894, 0.8081, 0.9022,
0.9293, 0.9560, 0.9713, 0.9804]
val_accuracy = [0.6011, 0.7128, 0.7136, 0.7528, 0.8152, 0.9136,
0.9152, 0.9144, 0.9252, 0.9244]
f1_score = [0.6188, 0.7221, 0.1143, 0.8131, 0.8914, 0.9062,
0.8926, 0.9725, 0.9885, 0.9919]
val_f1_score = [0.6584, 0.7508, 0.7459, 0.7442, 0.8472, 0.9440,
0.9461, 0.8462, 0.9483, 0.9463]
loss = [2.6161, 2.3480, 2.1381, 1.9697, 1.7257, 1.5275, 1.2263,
0.7198, 0.2949, 0.1116]
val_loss = [2.3329, 2.8902, 2.3006, 2.1876, 2.1032, 2.5690,
2.1499, 1.8062, 1.4467, 0.6338]
precision = [0.6413, 0.7638, 0.7747, 0.8186, 0.8114, 0.9188,
0.9197, 0.9298, 0.9359, 0.9516]
val_precision = [0.6400, 0.7164, 0.7145, 0.7445, 0.8021, 0.9232,
0.8989, 0.9199, 0.9359, 0.9255]

# Creating plots
plt.figure(figsize=(14, 10))

# Accuracy plot
plt.subplot(2, 2, 1)
plt.plot(epochs, accuracy, 'bo-', label='Train Accuracy')
plt.plot(epochs, val_accuracy, 'ro-', label='Val Accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()

# F1 Score plot
plt.subplot(2, 2, 2)
```

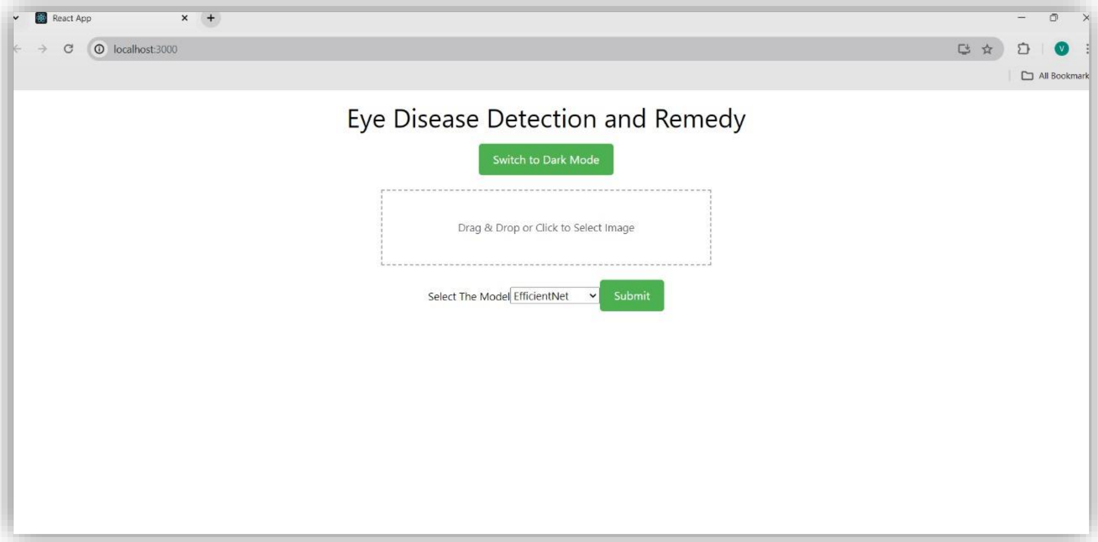
```
plt.plot(epochs, f1_score, 'bo-', label='Train F1 Score')
plt.plot(epochs, val_f1_score, 'ro-', label='Val F1 Score')
plt.xlabel('Epochs')
plt.ylabel('F1 Score')
plt.legend()

# Loss plot
plt.subplot(2, 2, 3)
plt.plot(epochs, loss, 'bo-', label='Train Loss')
plt.plot(epochs, val_loss, 'ro-', label='Val Loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()

# Precision plot
plt.subplot(2, 2, 4)
plt.plot(epochs, precision, 'bo-', label='Train Precision')
plt.plot(epochs, val_precision, 'ro-', label='Val Precision')
plt.xlabel('Epochs')
plt.ylabel('Precision')
plt.legend()

plt.tight_layout()
plt.show()
```

List of Abbreviations/Nomenclature

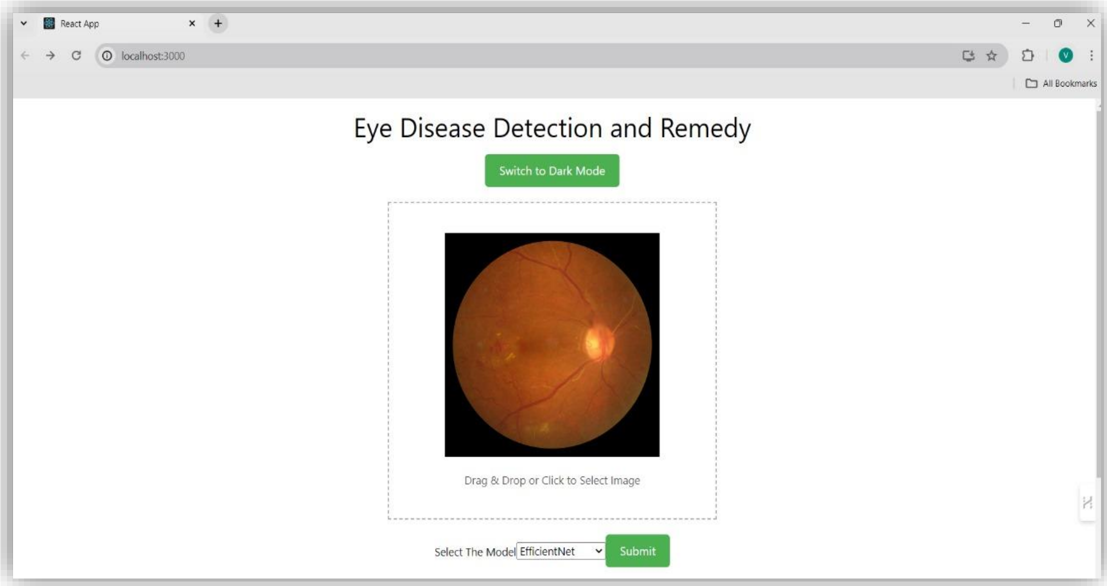
1.	DL:	Deep Learning
2.	CNN:	Convolution Neural network
3.	OCT:	Optical coherence tomography
4.	RNN:	Recurrent neural networks
5.	ResNet:	Residual network
6.	SVM:	Support Vector Machines

LIST OF FIGURES

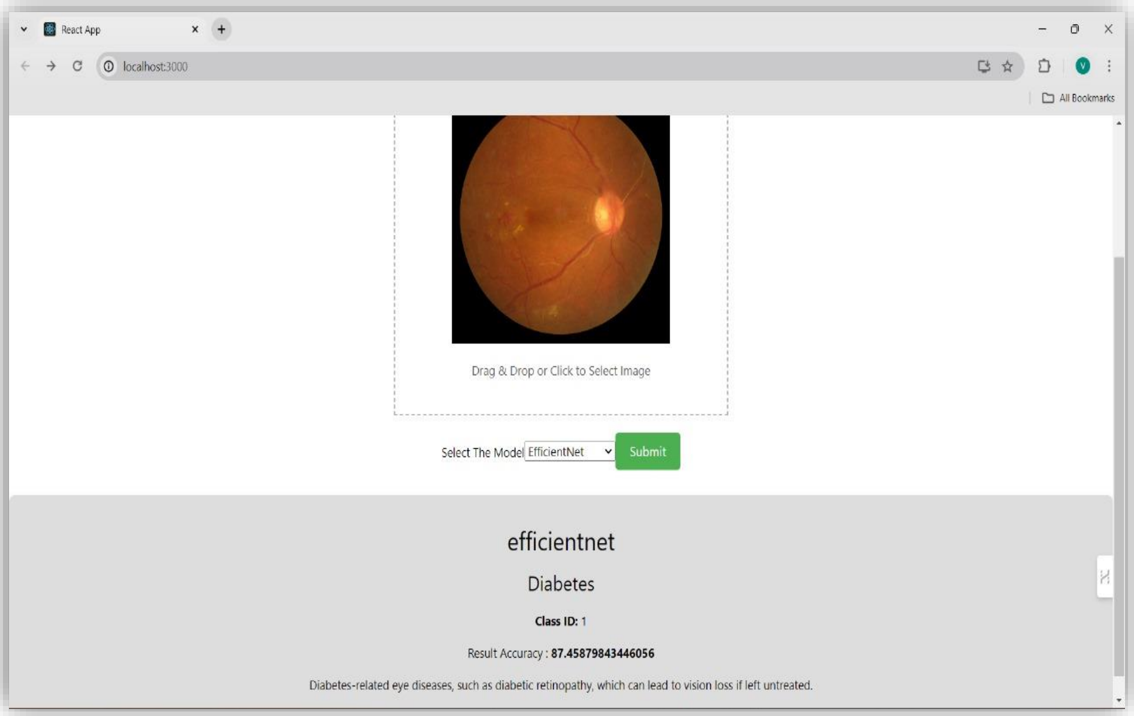
SCREENSHOTS



Above figure represents the opening interface of the project, where we can select the theme of the project(like dark mode or light mode), and we have to drop/select the image.



Above figure shows that, the user is allowed to insert the image and can select the model which has to be used to run the process.



The above figure shows the results (like used model, disease, disease ID, and confidence of disease) which are shown after the submission of the image and model.

REFERNCES

[1] J. Wang, L. Yang, Z. Huo, W. He and J. Luo, "Multi-Label Classification of Fundus Images with EfficientNet," in IEEE Access, vol. 8, pp. 212499-212508, 2020, doi: 10.1109/ACCESS.2020.3040275

[2] i, N., Li, T., Hu, C., Wang, K., Kang, H. (2021). A Benchmark of Ocular Disease Intelligent Recognition: One Shot for Multi-disease Detection. In: Wolf, F., Gao, W. (eds) Benchmarking, Measuring, and Optimizing. Bench 2020. Lecture Notes in Computer Science(), vol 12614. Springer, Cham. https://doi.org/10.1007/978-3-030-71058-3_11

[3] Muchuchuti . S, Viriri, "Retinal Disease Detection Using Deep Learning Techniques: A Comprehensive Review", J. Imaging 2023, 9, 84. https://doi.org/10.3390/jimaging9040084

[4] Md Shakib Khan, Nafisa Tafshir, Kazi Nabiul Alam, Abdur Rab Dhruba, Amani Abdulrahman Albraikan, and Faris A. Almalki, "Deep Learning for Ocular Disease Recognition: An Inner-Class Balance", https://doi.org/10.1155/2022/5007111

[5] O. Bernabé, E. Acevedo, A. Acevedo, R. Carreño and S. Gómez, "Classification of Eye Diseases in Fundus Images", in IEEE Access, vol. 9, pp. 101267-101276, 2021, doi: 10.1109/ACCESS.2021.3094649

[6] L. Jain, H. V. S. Murthy, C. Patel and D. Bansal, "Retinal Eye Disease Detection Using Deep Learning", 2018 Fourteenth International Conference on Information Processing (ICINPRO), Bangalore, India, 2018, pp. 1-6, doi: 10.1109/ICINPRO43533.2018.9096838

[7] Sushma K Sattigeri, Harshith N, Dhanush Gowda N, K A Ullas, Aditya M S, "EYE DISEASE IDENTIFICATION USING DEEP LEARNING" volume: 09

[8] R. Sarki, K. Ahmed, H. Wang and Y. Zhang, "Automatic Detection of Diabetic Eye Disease Through Deep Learning Using Fundus Images: A

Survey", in IEEE Access, vol. 8, pp. 151133-151149, 2020, doi: 10.1109/ACCESS.2020.3015258.

[9] K. Vayadande, V. Ingale, V. Verma, A. Yeole, S. Zawar and Z. Jamadar, "Ocular Disease Recognition using Deep Learning", 2022 International Conference on Signal and Information Processing (IConSIP), Pune, India, 2022, pp. 1-7, doi: 10.1109/ICoNSIP49665.2022.10007470.

[10] Gour, N., Tanveer, M. & Khanna, P. Challenges for ocular disease identification in the era of artificial intelligence. Neural Comput & Applic 35, 22887–22909 (2023). https://doi.org/10.1007/s00521-021-06770-5

[11] Puttagunta M, Ravi S. Medical image analysis based on deep learning approach. Multimed Tools Appl. 2021;80(16):24365-24398. doi: 10.1007/s11042-021-10707-4. Epub 2021 Apr 6. PMID: 33841033; PMCID: PMC8023554.

[12] Sarker, I.H. Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and Research Directions. SN COMPUT. SCI. 2, 420 (2021). https://doi.org/10.1007/s42979-021-00815-1

[13] Walton, O.B., Garoon, R.B., Weng, C.Y., Gross, J., Young, A.K., Camero, K.A., Jin, H.D., Carvounis, P.E., Coffee, R.E., & Chu, Y. (2016). Evaluation of Automated Teleretinal Screening Program for Diabetic Retinopathy. JAMA ophthalmology, 134 2, 204-9 .

[14] Alzubaidi, L.; Al-Amidie, M.; Al-Asadi, A.; Humaidi, A.J.; Al-Shamma, O.; Fadhel, M.A.; Zhang, J.; Santamaría, J.; Duan, Y. Novel Transfer Learning Approach for Medical Imaging with Limited Labeled Data. Cancers 2021, 13, 1590. https://doi.org/10.3390/cancers13071590

[15] V. -T. Hoang and K. -H. Jo, "Practical Analysis on Architecture of EfficientNet," 2021 14th International Conference on Human System Interaction (HSI), Gdańsk, Poland, 2021, pp. 1-4, doi:

10.1109/HSI52170.2021.9538782. keywords: {Computer vision; Analytical models; Costs; Computational modeling; Computer architecture; Real-time systems; Mobile handsets; efficient CNN architecture; efficientnet; SE module}.

[16] Z. Zhang, "Improved Adam Optimizer for Deep Neural Networks," 2018 IEEE/ACM 26th International Symposium on Quality of Service (IWQoS), Banff, AB, Canada, 2018, pp. 1-2, doi: 10.1109/IWQoS.2018.8624183. keywords: {Optimization; Training; Neural networks; Task analysis; Machine learning; Bridges; Conferences}.

ICAAIC

3rd International Conference on Applied Artificial Intelligence and Computing (ICAAIC-2024)

5-7, June 2024 | Salem, India

icaaic.contact@gmail.com | http://icaaic.com/2024/

Letter of Acceptance

Author Name: N. Balakrishna, M. B. Mukesh Krishnan, E. Venkata Ram Sai, K. Sonika, L. Gouri Priyanka, S. Vinish Ranganath

**Affiliation Details: Mohan Babus University, Tirupati, Andhra Pradesh, India.
SRMIST, Chennai, Tamilnadu, India.
Sree Vidyanikethan Engineering colleges, Tirupati, Andhra Pradesh, India**

Dear Author:

It is with great pleasure that we extend our warmest congratulations to you on the acceptance of the paper titled **"Ocular Disease Recognition Using Deep Learning "- PAPER ID: ICAAIC 684** for presentation at the 3rd International Conference on Applied Artificial Intelligence and Computing, scheduled to be held in R P Sarathy Institute of Technology, Salem, India from June 5th to June 7th, 2024.

Your submission was subjected to a rigorous review process, and the result that your paper has been selected for inclusion in our conference program. We believe that your contribution will greatly enrich the discussions and knowledge exchange at our event.

Your participation will undoubtedly contribute to the success of the 3rd International Conference on Applied Artificial Intelligence and Computing.

Once again, congratulations on your acceptance, and we anticipate your valuable contribution to our conference.

Yours Sincerely,

International Conference on Applied Artificial Intelligence and Computing
ICAAIC

Dr. Munusami Viswanathan,
Principal,
R P Sarathy Institute of Technology,
Salem, Tamil Nadu, India.

IEEE