Enhanced Indoor Localization using CNN and LSTM

P. MANUSH YADAV (21951A6777)

E. PRANITH KUMAR (21951A67A6)

M. SAI PRASHANTH (21951A67C2)

Enhanced Indoor Localization using CNN and LSTM

A Project Report
Submitted in Partial Fulfilment of the
Requirements for the Award of the Degree of

Bachelor of Technology in
CSE (Data Science)

by

P. Manush (21951A6777)

E. Pranith Kumar (21951A67A6)

M. Sai Prashanth (21951A67C2)

Under the Esteemed Guidance of
Dr. A. Naresh Kumar
Assistant Professor

Department of CSE (Data Science)

INSTITUTE OF AERONAUTICAL ENGINEERING
(Autonomous)
Dundigal, Hyderabad – 500 043, Telangana

May, 2025

DECLARATION

We certify that

a. The work contained in this report is original and has been done by me under the guidance of my supervisor(s).

b. The work has not been submitted to any other Institute for any degree or diploma.

c. I have followed the guidelines provided by the Institute in preparing the report.

d. I have conformed to the norms and guidelines given in the Ethical Code of Conduct of the Institute.

e. whenever I have used materials (data, theoretical analysis, figures, and text) from other sources, I have given due credit to them by citing them in the text of the report and giving their details in the references. Further, I have taken permission from the copyright owners of the sources, whenever necessary.

Place: Hyderabad

Date:

Signature of the Student(s)

21951A6777
21951A67A6
21951A67C2

CERTIFICATE

This is to certify that the project report entitled **Enhanced Device-Free Indoor Localization using CNN and LSTM** submitted by **Mr. P. Manush, E. Pranith Kumar, M. Sai Prashanth,** to the Institute of Aeronautical Engineering, Hyderabad, in partial fulfilment of the requirements for the award of the Degree Bachelor of Technology in **CSE (Data Science)** is a bonafide record of work carried out by him/her under my/our guidance and supervision. In whole or in parts, the contents of this report have not been submitted to any other institute for the award of any Degree.

Supervisor	**Head of the Department**
Dr. A. Naresh Kumar	Dr. K Rajendra Prasad
Assistant Professor	Professor &Head

APPROVAL SHEET

This project report entitled **Enhanced Device-Free Indoor Localization using CNN and LSTM** submitted by **Mr. P. Manush, E. Pranith Kumar, M. Sai Prashanth,** is approved for the award Degree Bachelor of Technology in **Data Science.**

Examiner

Supervisor
Dr. A. Naresh Kumar
Assistant Professor, CSE(DS)

Principal
Dr. L V Narasimha Prasad

Date:

Place: Hyderabad

ACKNOWLEDGEMENT

We would like to seize this moment to convey our heartfelt appreciation to everyone who supported, motivated, and cooperated with us in various capacities throughout our project. It brings me immense joy to recognize the assistance provided by those individuals who played a crucial role in ensuring the successful culmination of our project.

We extend our heartfelt thanks to **Dr. K Rajendra Prasad**, Head of the Department of Data Science, who served as my supervisor. We express our deep appreciation for the invaluable guidance and support provided by the faculty of the Data Science Department throughout the development of this project. Without their advice, cooperation, and encouragement, this project would not have come to fruition. A special note of gratitude goes to my friends for their assistance in project development. Lastly, we wish to convey my gratitude to our principal,

Dr. L V Narasimha Prasad, the management, and my parents for their unwavering support in all circumstances.

With Gratitude

P. Manush(21951A6777)

E. Pranith Kumar (21951A67A6)

M. Sai Prashanth (21951A67C2)

Abstract

An indoor positioning system is proposed utilizing Bluetooth Low Energy (BLE) beacons with iBeacon technology to address challenges in indoor localization. While GPS works well outdoors, it is ineffective indoors due to satellite signal restrictions. iBeacon technology, designed for both Android and iOS, offers a cost-effective, energy-efficient solution that operates without internet connectivity.

It enhances location-based services in sectors such as retail, event management, and education. By leveraging BLE beacons, the system can deliver notifications in the background, improving user experience in indoor environments. Recent advancements focus on integrating X and Y coordinate predictions using Convolutional Neural Networks (CNNs) to analyze spatial sensor data. A novel approach transforms sensor readings into an image-like format, enabling CNNs to capture spatial relationships effectively. Additionally, unsupervised pretraining with auto encoders reduces the need for manual measurements by utilizing unlabeled data, improving model performance.
This innovative method offers a more accurate and efficient indoor positioning solution, with potential applications across various industries. The project's implementation phase includes extensive testing and evaluation to ensure the system's accuracy and reliability in real-world indoor settings. The proposed system offers significant benefits, including low deployment cost, cross-platform compatibility for Android and iOS, low energy requirements, and adaptability to different indoor environments. With a combination of BLE beacons, CNN-based spatial prediction, and iBeacon technology, this project demonstrates a scalable and efficient approach to indoor positioning, opening possibilities for applications in retail, event management, healthcare, and educational sectors, where real-time indoor navigation and contextual information are crucial.

Key Words —Received Signal Strength Indicator (RSSI), Bluetooth Low Energy (BLE), iBeacon, Multilayer Perceptron (MLP).

Contents

List of Tables

List of Figures

CHAPTER -1

INTRODUCTION

1.1 INTRODUCTION

In this project, a complete indoor positioning system is proposed, which utilizes Bluetooth Low Energy (BLE) beacon with iBeacon technology for location-based services in the buildings. The proliferation of tracking devices (smartphones with GPS embedded) equipped with certain low- power sensors such as accelerometers has transformed many aspects of human lives, also creating new opportunities. GPS technology works great for outdoor positioning but falters indoors due to satellite signal restrictions. The purpose of the project is to use iBeacon (one workflow, with more location power) technology and solve the problems of indoor positioning in a creative way. Low Energy Beacons (BLE) that come with iBeacon satisfy such characteristic and represent a built-in cross-platform technology for Android and iOS devices in the long run. The iBeacon technology offers a range of significant benefits that make it a valuable tool for various sectors, including retail, event management, and education.

Its advantages include cost-effective hardware, lower energy consumption, independence from internet connectivity, and the capability to send notifications in the background. These features enhance communication and improve user experiences in indoor environments. Recent advancements in iBeacon projects have focused on integrating both X and Y coordinate predictions into a unified model. This approach employs a valid time series data split for training and testing, utilizing Convolutional Neural Networks (CNNs) to analyze sensor data spatially. A novel method transforms sensor readings into an image-like format, allowing CNNs to effectively capture spatial relationships. Additionally, unsupervised pretraining with autoencoders is leveraged to utilize unlabeled data, which can minimize the need for manual measurements in real-world settings.

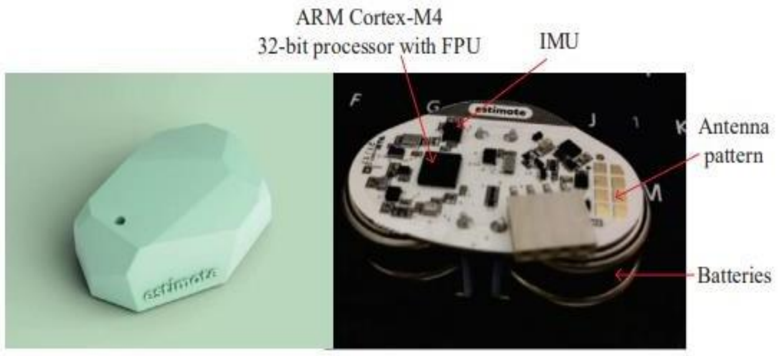


Figure 1.1: Layout of the experiment scene.

Device-Free Localization (DFL) has been widely employed in various industry applications, such as the health care and security industries. A large number of technologies, such as Ultra-Wide Band (UWB), Bluetooth, and infrared sensors, have been proposed in meeting the increasing performance requirement for indoor localization. The WIFI-based localization approach has attracted considerable attention due to the popularity of commercial network interface cards (NICs). It was reported that either Received Signal Strength (RSS) or Channel State Information (CSI) could be employed for positioning. The superiority of CSI over RSS in both stability and resolution makes it more attractive.

Due to the recent progress on deep learning (DL), numerous CSI-based DL methods have been proposed for the purpose of DFL. ConFi used the CSIs from multiple antennas for training a convolutional neural network (CNN) to predict the position of the interested object. CiFi employed the phase information of CSI to estimate the angle of arrival (AoA).

In, the use of LSTM was shown to be helpful for the deep-learning-based DFL. In particular, a novel CSI-based image construction method was proposed in to improve the accuracy of localization experimental results.

1.2 EXISTING SYSTEM

- RSSI (Received Signal Strength Indicator) is commonly used for indoor localization due to its simplicity and ease of acquisition. It does not require specialized hardware, making it accessible for widespread implementation. However, RSSI is highly volatile and provides only coarse information, as it merely indicates the strength of the received signal without detailed insights into the wireless environment.
- Ensemble Methods: Ensemble methods combine multiple models to improve predictive performance. Techniques like bagging, boosting, and stacking have been applied to diabetic retinopathy detection.
- CSI (Channel State Information) offers more detailed data about the wireless signal, including phase and amplitude across different sub-channels. This additional information makes CSI more stable and potentially more accurate than RSSI. However, CSI is still susceptible to environmental changes, which can introduce volatility and affect the reliability of localization in dynamic indoor settings.
- Fingerprinting Techniques: One common approach involves creating a database (radio map) of signal measurements at known locations. During real-time localization, current RSSI values are compared to the stored fingerprints. While this method can be accurate, it is labor-intensive and requires frequent updates as environments change (e.g., due to furniture movement or crowd presence).
- Proximity-Based Methods: These methods estimate location based on the presence of nearby beacons or access points. Although simple and efficient, they lack precision and are only suitable for applications where room-level accuracy is sufficient.
- Angle of Arrival (AoA) and Time of Flight (ToF): These techniques use signal properties to estimate direction or distance. While they can significantly enhance accuracy, they demand specialized hardware and are sensitive to multipath effects and line-of-sight obstructions.

1.2.1 DEMERITS OF EXISTING SYSTEM

- The drawbacks of the existing system for indoor localization using Wi-Fi, which relies on either Received Signal Strength Indicator (RSSI) or Channel State Information (CSI), include: Volatility: Both RSSI and CSI are susceptible to fluctuations due to environmental changes, such as interference from other devices or obstacles in the environment.
- Complexity and Cost (CSI): While CSI provides more detailed information, including phase and amplitude in different sub-channels, acquiring and processing CSI data can be more complex and require specialized hardware, increasing the cost and complexity of the system.
- Data Collection and Maintenance: Both RSSI and CSI-based systems require a large amount of labeled data for training and maintenance, which can be time-consuming and labor-intensive to collect and maintain, especially in multi-environment localization scenarios. Environmental Sensitivity, Limited Generalization.
- **Limited Generalization:** Models trained in one specific indoor environment often fail to perform well when applied to another location without retraining, limiting their generalizability and portability.
- **Calibration Requirement:** Frequent recalibration is needed to maintain accuracy, especially in highly dynamic environments. This adds maintenance overhead and limits the practicality of deploying such systems at scale.
- **Latency in Real-Time Tracking:** Some systems may experience delays in real-time position estimation due to heavy computations involved in signal processing and model inference, which can hinder user experience in fast-moving scenarios.
- **Low Precision in Crowded Areas:** When multiple devices or users are present in the same area, signal overlap and interference can cause ambiguity in position estimation, resulting in reduced precision.

1.3 PROPOSED SYSTEM

The proposed indoor positioning system is designed to address the challenge of accurately locating users within indoor environments where GPS technology typically fails. This system utilizes Bluetooth Low Energy (BLE) beacons with iBeacon technology, offering a low-cost, low-power solution that does not require continuous internet connectivity. These BLE beacons are strategically positioned throughout the indoor space—whether it be a shopping mall, museum, educational campus, or other setting—providing localized signal strength information (RSSI) that can be captured by nearby devices, such as smartphones. The system capitalizes on the wide availability of smartphones with embedded GPS and sensors like accelerometers, which are essential for monitoring user movement and enhancing location accuracy in real-time.

Upon receiving the signal from the iBeacons, the smartphone records the RSSI and accelerometer data, which is then passed through a data preprocessing module. This module performs normalization to manage RSSI fluctuations caused by signal interference and environmental changes. To improve interpretability, the preprocessed data is transformed into an image-like format, enabling the Convolutional Neural Network (CNN) to capture spatial features effectively. This transformation converts sensor readings into an intuitive spatial representation, where relationships among values are preserved, allowing the CNN model to analyze the information as if it were a physical map.

The CNN model used in this project is trained to predict both X and Y coordinates within the indoor space, thus allowing for precise location tracking. The training process leverages a time-series split to simulate real-world positioning scenarios, and includes unsupervised pretraining using autoencoders to utilize unlabeled data efficiently. This pretraining approach reduces the need for manually labeled data, making the system both scalable and adaptable to various real-world applications. By learning to identify spatial patterns within sensor data, the CNN effectively maps different signal strengths to corresponding indoor locations, creating a robust model for position prediction.

Once the CNN model is trained and evaluated, it is integrated into a mobile application, enabling real- time positioning for end users. This app continuously tracks and processes live sensor data to predict the user's indoor position.

The final deployment phase of the system involves real-world testing within target environments, enabling further fine-tuning and performance optimization. This proposed system, combining BLE iBeacons, smartphone sensors, CNN-based spatial prediction, and a notification-ready mobile application, offers a complete solution for indoor positioning. This robust system holds potential for various sectors, from retail and event management to educational and healthcare facilities, where it can enhance navigation, communication, and overall user engagement in indoor spaces.

1.3.1. MERITS OF PROPOSED SYSTEM

The proposed indoor positioning system offers several notable merits:

1. **High Accuracy in Indoor Positioning**: Utilizing BLE beacons and iBeacon technology, combined with CNN-based spatial prediction, allows for precise tracking within indoor environments, where GPS is ineffective.
2. **Low Power Consumption**: BLE beacons are designed for low-energy operation, making the system energy-efficient. The technology can function without continuous internet connectivity, conserving battery life on both beacons and smartphones.
3. **Cost-Effectiveness**: BLE beacons are relatively inexpensive and easy to install, making the solution affordable and accessible for various industries, from retail to educational institutions.
4. **Real-Time Positioning and Navigation**: The system provides real-time location updates, facilitating seamless indoor navigation and location-based services, improving user experience in malls, museums, hospitals, and other large indoor areas.
5. **Enhanced User Engagement with Notifications**: iBeacon's ability to send location-specific notifications enhances engagement, delivering targeted content such as promotions, event information, or directions based on the user's location.
6. **Cross-Platform Compatibility**: iBeacon technology supports both Android and iOS devices, making the solution versatile and accessible to a broader user base.
7. **Scalability with Unsupervised Pretraining**: Using autoencoders for unsupervised pretraining enables the system to learn from unlabeled data, reducing the need for manual data collection and making it scalable for larger environments.
8. **Minimal Infrastructure Requirements**: The system only requires strategically placed BLE beacons and smartphones with basic sensors, making it easy to deploy and maintain without complex infrastructure.

CHAPTER 2

LITERATURE SURVEY

2.1 LITERATURE REVIEW:

Wu Wei, Jun Yan, Gengxin Zhang, created a meta-learning approach for device-free localization is proposed for generalizing a trained model to a new task with a few numbers of labeled samples. The proposed method demonstrates its superiority over DL with extensive experimental results. Either the training or target tasks require only few CSI samples for the purpose of good localization performance. The proposed weighted loss dynamically adjusts the weights, which facilitates the knowledge transfer from the training tasks toward the target task. [1].

Ali Owfi, ChunChih Lin, Linke Guo that they paper addresses the lack of generalizability of conventionally trained indoor localization models by proposing meta-learning-based localization. Moreover, we designed a new meta-learning algorithm, TB-MAML, specialized to reach better generalizability when the number of scenarios available for training a meta learning model is limited. This characteristic of TB-MAML is desired in the context of indoor localization, as collecting large enough diverse scenarios is difficult and time-consuming. [2].

Honghong Chen, Jie Yang, this paper propose a LoRa-based improved fingerprint localization algorithm-particle swarm optimization-random forest-fingerprint localization for indoor localization. The first improvement step involves creating a new exceptional fingerprint value (referred to as RSSI-RANGE) by adding the Time of Flight ranging value (referred to as RANGE) to the Received Signal Strength Indication (RSSI) value and weighting them together. [3].

Wafa Njima, Raed M. Subair The paper "Indoor Localization Using Data Augmentation via Selective Generative Adversarial Networks" proposes a novel approach to indoor localization using deep neural networks and received signal strength indicator (RSSI) fingerprints. It addresses the challenge of collecting large amounts of training data by employing generative adversarial networks for RSSI data augmentation, leading to a localization accuracy improvement of 21.69% for simulated data and 15.36% for experimental data. [4].

Shivenkumar Parmar The paper "Voice Fingerprinting for Indoor Localization with a Single Microphone Array and Deep Learning" proposes a novel approach to indoor localization using voice fingerprints captured by a single microphone array. It leverages deep learning techniques to extract unique acoustic features from voice signals, achieving accurate localization with a median error of 1.3 meters in a real-world environment. [5].

Xiaofu wu The paper "A Data Preprocessing Method for Deep Learning Based Device-Free Localization" presents a novel data preprocessing approach to improve the accuracy of device-free localization systems using deep learning. It proposes a combination of filtering, normalization, and feature extraction techniques to enhance the quality of wireless signal strength data, leading to improved localization performance. [6].

Wu Wei This paper "CSI Fingerprinting for Device-Free Localization: Phase Calibration and SSIM-Based Augmentation" proposes a novel approach to device-free localization using CSI fingerprinting. It addresses the challenges of phase shifts and imperfect synchronization by introducing a phase calibration method and a structural similarity-based augmentation technique, enhancing the accuracy and robustness of the localization system [7].

Bing-Jia The paper "Few-Shot Transfer Learning for Device-Free Fingerprinting Indoor Localization" proposes a novel approach to device-free indoor localization using few-shot transfer learning and graph neural networks (GNNs). It addresses the challenge of collecting and labeling large amounts of data for each new environment by leveraging a small amount of labeled data from the current environment and reusing existing labeled data from other environments. The proposed system achieves comparable performance to a convolutional neural network (CNN) model with 40 times fewer labeled data [8].

2.2 REQUIREMENT SPECIFICATIONS

2.2.1 HARDWARE REQUIREMENTS

The hardware specifications for the Device-free indoor Localization using Meta-Learning and Deep Learning project are outlined as follows:

● **Processing Unit:** A multi-core CPU with high processing power to handle complex computations involved in training deep neural networks.

● **Graphics Processing Unit (GPU):** A high-performance GPU, such as NVIDIA GeForce GTX with 3090, to accelerate the training process of deep learning models.

● **Commercial Wireless Access Points (AP):** Devices for transmitting and receiving WIFI signals.

● **Intel 5300 NIC:** Network Interface Card for collecting CSI data.

● **Memory (RAM):** A minimum of 8GB RAM to support the loading and processing of large retinal image datasets.

● **Data Storage:** Hard drives or cloud storage for storing collected data and models.

● Bluetooth Low Energy (BLE) Beacons: For emitting signal packets used in RSSI-based indoor positioning.

● **Mobile Device or Smartphone:** Required for receiving BLE signals and acting as a testbed for real-time localization experiments.

● **Uninterruptible Power Supply (UPS):** To ensure consistent power supply during continuous training or data collection processes.

● **Router/Switch**: For establishing local wireless communication between access points and mobile or embedded devices.

2.2.2 SOFTWARE REQUIREMENTS

The software components necessary for the project implementation include:

- **Operating System:** Platform-independent, compatible with Windows, Linux, or macOS.
- **Programming Language:** Python 3.x Programming language for implementing machine learning models
- **Deep Learning Frameworks:** TensorFlow, PyTorch for building, training, and evaluating deep neural networks.
- **Data Processing Libraries:** Pandas and NumPy for efficient data manipulation, and scikit-learn for preprocessing tasks.
- **Containerization:** Containerization Docker or K8S, K3S tool for creating reproducible environments.
- **IDE/Notebook Interface**: Jupyter Notebook or Visual Studio Code (VS Code) for writing and testing code interactively.
- **Visualization Tools:** Matplotlib, Seaborn, and Plotly for creating plots and visualizing model performance and signal data.
- **Version Control**: Git and GitHub/GitLab for source code versioning and collaborative development.
- **Model Monitoring Tools**: Tensor Board or Weights & Biases (wand) for real-time model training visualization and hyperparameter tuning.
- **BLE Beacon Configuration Tool:** Apps or utilities (like Nordic nRF Connect) to configure and monitor BLE beacon signals.

2.2.3 FUNCTIONAL REQUIREMENTS

The functional requirements encompass the following key functionalities:

- **Data Collection:** Data collection is a vital step in creating an indoor positioning system that employs Bluetooth Low Energy (BLE) beacons. This process involves systematically gathering data to accurately estimate the locations of mobile devices within an indoor setting. The methodology begins with the deployment of BLE beacons throughout the designated area, ensuring their optimal placement for maximum signal coverage and minimal interference. The first step involves selecting suitable BLE beacons based on critical criteria such as range, battery life, and accuracy.
- **Data Preprocessing:** Data preprocessing is an essential phase in preparing collected data for analysis and model training in an indoor positioning system that uses Bluetooth Low Energy (BLE) beacons. This step ensures that the data is clean, consistent, and ready for Machin learning algorithms, ultimately enhancing the accuracy and performance of the positioning system. The following outlines common data preprocessing techniques used in this project:

- **Model Preparation & Training:** In this project, model selection focuses on identifying algorithms that effectively address the specific challenges associated with indoor positioning using Bluetooth Low Energy (BLE) beacons. Initially, a Multilayer Perceptron (MLP) was utilized to predict X and Y coordinates, capitalizing on its capability to handle high-dimensional input data. However, recognizing the importance of spatial relationships in this context, a Convolutional Neural Network (CNN) was later incorporated. This CNN treats the RSSI data as grid-like images, which enhances the model's ability to understand the spatial connections between beacon signals and their corresponding locations. This method improves generalizability, allowing the model to perform effectively across various environments.Additionally, an autoencoder was integrated for unsupervised pretraining, which makes use of a large volume of unlabeled data to extract meaningful features. This approach helps reduce overfitting and enhances the robustness of the model.

- **Meta-Learning Integration:** To enhance the indoor localization system, we integrate meta-learning techniques, including MAML, which enables quick adaptation to new tasks. A weighted loss function is used to prioritize more relevant tasks, and ensemble learning combines models, optimizing them using meta-learning. This integration facilitates rapid adaptation to novel environments with minimal labeled data, improving the system's accuracy and robustness.

- **Model Optimization:** To further improve the indoor localization system, we employ model optimization techniques. Hyperparameter tuning is performed to adjust parameters for optimal performance. Feature fusion is used to combine features from multiple models, enhancing the feature set. Additionally, transfer learning is applied by fine- tuning pre-trained models on the dataset, allowing the system to leverage knowledge from related tasks and achieve better accuracy.

- **Performance Evaluation:** In this project, the evaluation process focuses on assessing how accurately model predicts the x and y coordinates of a mobile device's location based on the RSSI signals received from BLE beacons. To quantify prediction accuracy, key performance metrics such as Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE) are employed. These metrics provide insights into the average discrepancies between the predicted and actual positions. Validation is crucial to ensure that the model performs effectively not only on training data but also on previously unseen data.

CHAPTER 3

SYSTEM DESIGN

3.1 SYSTEM ARCHITECTURE

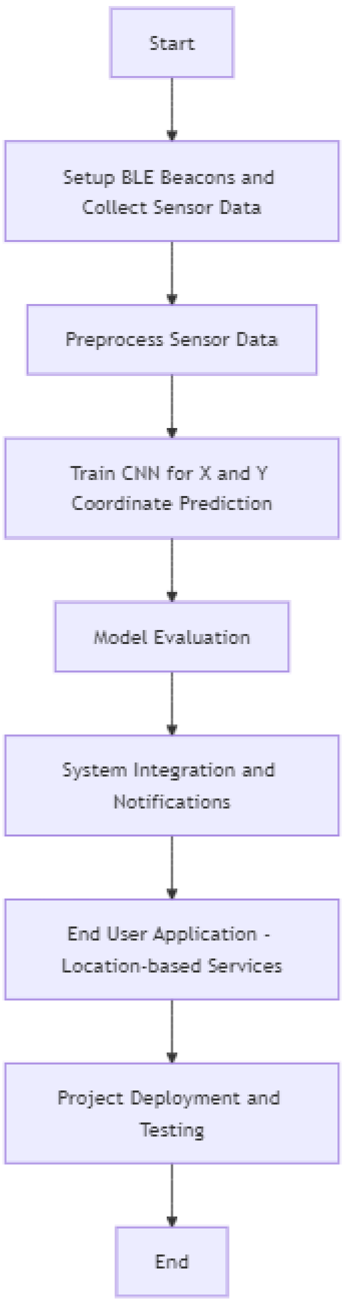

Start
Setup BLE Beacons and Collect Sensor Data
Preprocess Sensor Data
Train CNN for X and Y Coordinate Prediction
Model Evaluation
System Integration and Notifications
End User Application - Location-based Services
Project Deployment and Testing
End

Figure 3.1: Model flowchart

The architecture of an indoor positioning system utilizing Bluetooth Low Energy (BLE) beacons is designed to effectively capture, process, and analyze location data within buildings. It consists of several essential components:

1. **BLE Beacons**: These are strategically placed throughout the area to transmit unique identifiers and Received Signal Strength Indicator (RSSI) values. Their placement is crucial for ensuring optimal signal coverage. These beacons provide spatial data (RSSI) that can be detected by user devices to determine approximate proximity.
2. **Mobile Device**: This acts as the receiver, continuously scanning for nearby beacons and recording their signals. The mobile device plays a key role in determining the user's location based on the signals received from the beacons. An application on the smartphone gathers and processes both the RSSI signals from the beacons and the accelerometer readings, sending them to the server for further processing.
3. **Data Processing and Transformation Layer**

- **Data Normalization and Filtering**: Collected RSSI data and accelerometer readings are further processed to reduce noise, normalize values, and prepare the data for the model.
- **Image Transformation Module**: The processed sensor data is converted into an image-like format, allowing spatial relationships within the data to be interpreted by CNNs.

4. **Model Training Layer**

- **Convolutional Neural Network (CNN)**: The core predictive model, trained to estimate X and Y coordinates based on image-transformed sensor data.
- **Unsupervised Pretraining (Autoencoders)**: Used to learn from unlabeled data, helping the CNN to identify underlying spatial patterns in RSSI and accelerometer data. This pretraining minimizes the need for labeled data and enhances prediction accuracy.
- **Training and Testing Modules**: Data is split into training and testing sets for validating and fine-tuning the model to ensure accuracy and reliability.

5. **Integration and Application Layer**

- **Positioning Engine**: Integrated on the smartphone, this engine uses the trained CNN model to process real-time RSSI and accelerometer data, providing real-time X and Y coordinate predictions.
- **Notification Module**: Leverages iBeacon technology to send targeted, location-based notifications or alerts as users move through the indoor environment.

CHAPTER 4

METHODOLOGY AND IMPLEMENTATION

4.1 METHODOLOGY

The methodology for developing an indoor positioning system using Bluetooth Low Energy (BLE) beacons begins by clearly defining the system requirements, goals, specifications, and constraints for the project. The architectural design involves seamlessly integrating BLE beacons, a mobile application, a backend server, and a database to create a cohesive system. For the BLE beacon setup, the team carefully selects appropriate beacons based on factors such as range, battery life, and accuracy.

These beacons are then strategically installed throughout the indoor environment to ensure adequate coverage while minimizing interference. Configuration of the beacons includes assigning unique identifiers and setting optimal transmission power levels. Data collection is a crucial step, where the system utilizes the Received Signal Strength Indicator (RSSI) from mobile devices to measure proximity to the beacons. Fortunately, a great number of datasets are easily accessible to the public on the internet through social media, particularly for prominent personalities and celebrities [15]. This is why public personalities are the first targets of deepfake attacks. The scientific community provides several datasets for use in deepfake.

These principles can be used to create deepfake photographs and movies. Photos are faster to create since they are less in size and require less processing. Advances in deepfake methods and fake news underscore the dangers of hard-to-detect false photographs [16]. Deepfake technologies are becoming more user-friendly, and fake news based on these pictures is extensively distributed on the internet.
As a result, deepfake detection techniques are becoming more useful [17]. The creation of reliable deepfake detection algorithms is critical for reducing the spread of disinformation and protecting persons from the possible harm caused by deepfakes.

Preprocessing:

- Input: Acquire retinal fundus images.
- Preprocessing: Employ methods like contrast enhancement, histogram equalization, and noise reduction to accentuate crucial features
- Image Processing: Conduct cropping and resizing to standardize image dimensions.
- Data Augmentation: Optionally carry out rotation, flipping, and mirroring to diversify the dataset and enhance model resilience.
- Conversion: Transform processed images into numpy arrays for computational efficiency.
- Dataset Splitting: Segment the dataset into training, validation, and testing subsets.

EfficientNetB1 Backbone:

- The fundamental structure of the EfficientNetB1 model encompasses multiple blocks, each tasked with feature extraction and transformation.
- Each block comprises a sequence of operations, including convolutional layers, batch normalization, activation functions, and depthwise convolutions.
- For brevity, only one block (Block 1) is illustrated in the diagram.
- Block 1 commences with a convolutional layer (Conv2D) featuring a 3x3 kernel size, succeeded by batch normalization and the Swish activation function.
- Depthwise convolution (DepthwiseConv2D) with a 3x3 kernel is subsequently applied, followed by batch normalization and the Swish activation function once again.
- Ultimately, a 1x1 convolutional layer (Conv2D) further refines the feature maps, trailed by batch normalization to sustain stability.

Data Augmentation:

Data augmentation serves as a technique to artificially enrich the size and diversity of a dataset by implementing various transformations to the existing data. These transformations encompass flipping, rotating, scaling, cropping, and other operations that maintain the semantic content of the data while introducing variations in its appearance.

Now, let's elucidate each parameter:

- **horizontal_flip**: This parameter determines whether random horizontal flipping of images should be applied. When set to True, each image has a 50% chance of undergoing horizontal flipping. For instance, an image of a cat facing left might undergo flipping to orient the cat facing right.

- **rotation_range**: This parameter dictates the range within which random rotations can be applied to images. Here, rotations are randomly applied up to 20 degrees in either direction. This aids the model in becoming more resilient to images being slightly rotated in real-world scenarios.
- **width_shift_range**: This parameter governs the range by which images can be horizontally shifted. In this instance, images can be shifted horizontally by up to 20% of their total width in either direction. This introduces variability in the position of objects within the images.
- **height_shift_range**: Analogous to width_shift_range, this parameter controls the range by which images can be vertically shifted. Images can be vertically shifted by up to 20% of their total height in either direction. This introduces variability in the vertical position of objects within the images.

- **zoom_range**: This parameter dictates the range by which images can be zoomed in or out. Images can be zoomed in or out by up to 20%. This introduces variability in the scale of objects within the images.

Output Prediction:

Class probabilities are estimated in the final stage of the model using the extracted features. We generate probability distributions for each class employing a dense layer with a softmax activation function. In this scenario, there are five classes in the classification task, corresponding to the number of units in this dense layer. The softmax function ensures that the projected probabilities sum up to 1, facilitating comprehension and decision-making. Each unit in the dense layer represents a class. Consequently, the predicted label for the input image is determined by the model to be the class with the highest probability.

Custom CNN:

● The customized CNN architecture comprises 3 convolutional layers succeeded by max pooling layers, 2 dense layers, and an output layer.

● Convolutional layers utilize 32 filters of size (3, 3) with ReLU activation.

● Max pooling layers downsample feature maps by a factor of 2 along both spatial dimensions.

● Dense layers encompass 128 neurons with ReLU activation in the first layer and 5 neurons with softmax activation in the output layer. The architecture is tailored to process input images of size (image_height, image_width, 3) and classify them into 5 severity categories of diabetic retinopathy.

● This architecture is optimized for capturing pertinent features from retinal images and effectively categorizing them into distinct severity stages of diabetic retinopathy.

● Dropout regularization is applied after dense layers to reduce overfitting and improve generalization on unseen retinal scans.

● Batch normalization is introduced between convolutional layers to stabilize and accelerate training by normalizing the feature maps.

● The model uses categorical cross-entropy as the loss function and the Adam optimizer to efficiently update weights during training.

● Early stopping and learning rate scheduling are employed during training to prevent overfitting and optimize convergence.

● The CNN design ensures that local features such as microaneurysms, hemorrhages, and exudates are captured effectively, which are crucial for identifying early signs of diabetic retinopathy.

● Data augmentation techniques like rotation, flipping, and zooming are used during training to enhance the diversity of the input images and improve the robustness of the model.

Custom LSTM (Long Short-Term Memory):

- LSTM is a type of Recurrent Neural Network (RNN) specifically designed to learn and remember long-term dependencies in sequential data.
- It is particularly effective in scenarios where the data has a temporal or time-series structure, such as RSSI signal fluctuations in indoor positioning or speech and video processing.
- LSTM units include gates (input gate, forget gate, and output gate) that control the flow of information, enabling the model to retain or discard data as needed, which helps in dealing with vanishing gradient problems in standard RNNs.
- In the context of indoor localization, LSTM helps capture time-dependent patterns in RSSI signals, improving the model's ability to account for signal instability and temporal noise.
- Unlike feedforward networks, LSTM can maintain a memory of past signal variations, allowing the model to understand movement patterns and transitions between locations.
- LSTM networks can be stacked with other layers like dense or convolutional layers to build hybrid models that combine spatial and temporal learning.
- Training LSTM on sequential RSSI data can enhance prediction accuracy by learning the underlying signal dynamics that occur as a user moves through indoor environments.
- They are also useful in applications beyond localization, including speech recognition, financial forecasting, and activity recognition in smart environments.

4.2 Packages and Modules

In a diabetic retinopathy detection project utilizing the EfficientNetB1 algorithm, various packages, libraries, and modules are employed to facilitate data preprocessing, model development, training, evaluation, and deployment.

TensorFlow:

TensorFlow is an open-source machine learning framework widely used for building and training deep learning models. It provides tools and libraries for constructing neural networks, handling data, and optimizing model performance.

Keras:

Keras is a high-level neural networks API that runs on top of TensorFlow. It offers a user-friendly interface for building and training deep learning models with minimal code, making it suitable for rapid prototyping and experimentation.

Pandas:

Pandas is a Python library used for data manipulation and analysis. It provides data structures like DataFrame, which facilitate handling and preprocessing of tabular data, including retinal images and associated metadata.

NumPy:

NumPy is a fundamental package for scientific computing in Python. It provides support for large, multi-dimensional arrays and matrices, along with a collection of mathematical functions to operate on these arrays efficiently.

Matplotlib:

Matplotlib is a plotting library for Python used to create visualizations, such as line plots, scatter plots, histograms, and heatmaps. It is often utilized for visualizing data distributions, model performance metrics, and training/validation curves.

Scikit-learn:

scikit-learn is a machine learning library for Python that provides simple and efficient tools for data mining and data analysis. It offers various algorithms for classification, regression, clustering, and dimensionality reduction, which can complement deep learning models for diabetic retinopathy detection

OpenCV (Open Source Computer Vision Library):

OpenCV is a library of programming functions mainly aimed at real-time computer vision tasks. It provides tools for image processing, feature extraction, object detection, and image segmentation, which are essential for preprocessing retinal images and extracting relevant features.

SciPy:

SciPy is a Python library used for scientific and technical computing. It provides functions for optimization, integration, interpolation, and other mathematical tasks that may be useful in preprocessing or analyzing retinal image data.

shutil:

is a built-in module that provides a high-level interface for file operations, including copying, moving, renaming, and deleting files and directories. It offers convenient functions for working with file system paths and performing common file management tasks.

Albumentations:

A fast and flexible image augmentation library. It is used to perform advanced preprocessing techniques like flipping, rotation, brightness/contrast adjustment, and more, to increase the diversity of the training dataset and improve model generalization.

TensorFlow Hub:

A library for reusable machine learning modules. It allows loading pre-trained models such as EfficientNetB1 quickly and efficiently, enabling transfer learning for medical imaging tasks.

TQDM:

A library that provides fast, extensible progress bars for loops and training iterations, making it easier to track the model's training progress and runtime.

joblib:

Used for saving and loading Python objects efficiently, especially useful when working with trained models or preprocessing pipelines.

Seaborn:

A statistical data visualization library built on top of matplotlib. It's used to visualize data distributions, correlation matrices, and confusion matrices in a visually appealing manner.

os (Operating System Interface):

A built-in Python module used for interacting with the operating system, such as file path navigation and directory manipulation.

glob:

A module for Unix-style pathname pattern expansion, useful in loading datasets and organizing image files for training and validation.

skimage (scikit-image):

A collection of algorithms for image processing. Useful for segmentation, transformation, and filtering operations that might be needed for enhancing retinal images.

pickle:

Python's built-in module used to serialize and deserialize Python object structures, helpful for saving trained models and their configurations.

imutils:

A series of convenience functions to make basic image processing tasks like resizing, rotating, and displaying images easier when working with OpenCV.

4.3 DATA SET

The retinal image data collected from the Institute of Electrical and Electronics Engineers (IEEE) constitutes a valuable and diverse dataset sourced from reputable repositories within the organization. These retinal images serve as a foundation for various research, diagnostic, and analytical endeavors within the medical and scientific communities.

The dataset encompasses a comprehensive collection of retinal fundus images tailored specifically for diabetic retinopathy detection and evaluation. It is categorized into three distinct subsets: Segmentation, Disease Grading, and Localization. In the Segmentation subset, 81 original colour fundus images are meticulously annotated to delineate lesions such as Microaneurysms, Hemorrhages, Exudates, and the Optic Disc. These annotations serve as invaluable ground-truth data for training deep learning models aimed at accurately segmenting retinal structures in fundus images.

Moving to the Disease Grading subset, a total of 516 fundus images are meticulously labeled with severity grades corresponding to diabetic retinopathy and macular edema. These grade annotations provide a foundation for research endeavors focused on developing robust algorithms for classifying various stages of diabetic retinopathy and assessing disease progression. By leveraging this dataset, researchers can explore innovative approaches to
automate disease grading processes, ultimately aiding clinicians in making timely and informed treatment decisions.

The dataset which we are using was provided with a large amount of high-resolution retina images taken under a variety of imaging condition. The images which are provided in dataset are recorded from fundus camera which provides color fundus image of DR.

The clinicians are divided these DR into five chasses which shows the stages of DR:

- No DR (class 0)
- Mild (class 1)
- Moderate (class 2)
- Severe (class 3)
- Proliferative DR (class 4)

CSV (Comma Separated Values) file gives all the information of the image and it is in excel sheet. Train.csv contains the fundus eye image name and its severity level (class) and test.csv includes only the eye image name because it is going to be test after training the model

Training: The training phase of the diabetic retinopathy detection model involves a dataset initially comprising 413 raw retinal images.

Testing: The testing phase of the diabetic retinopathy detection model involves the evaluation of model performance using a carefully curated set of 103 retinal images.

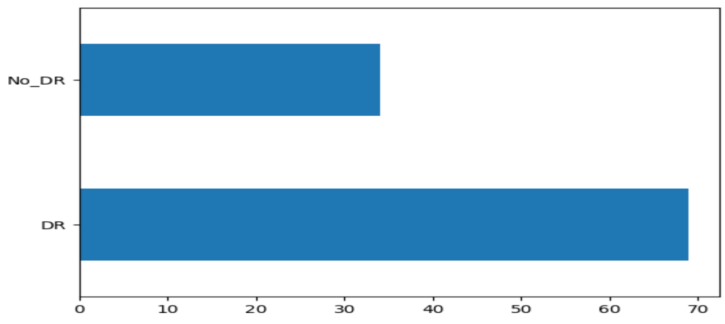


Figure 4.2: Division of images into DR and No_DR

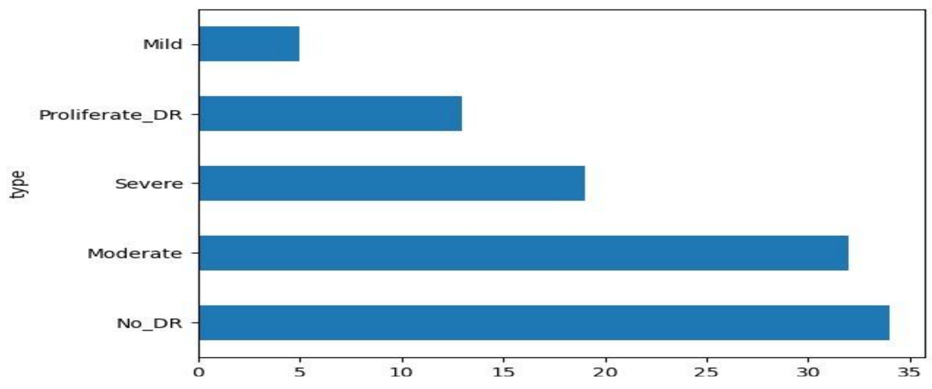
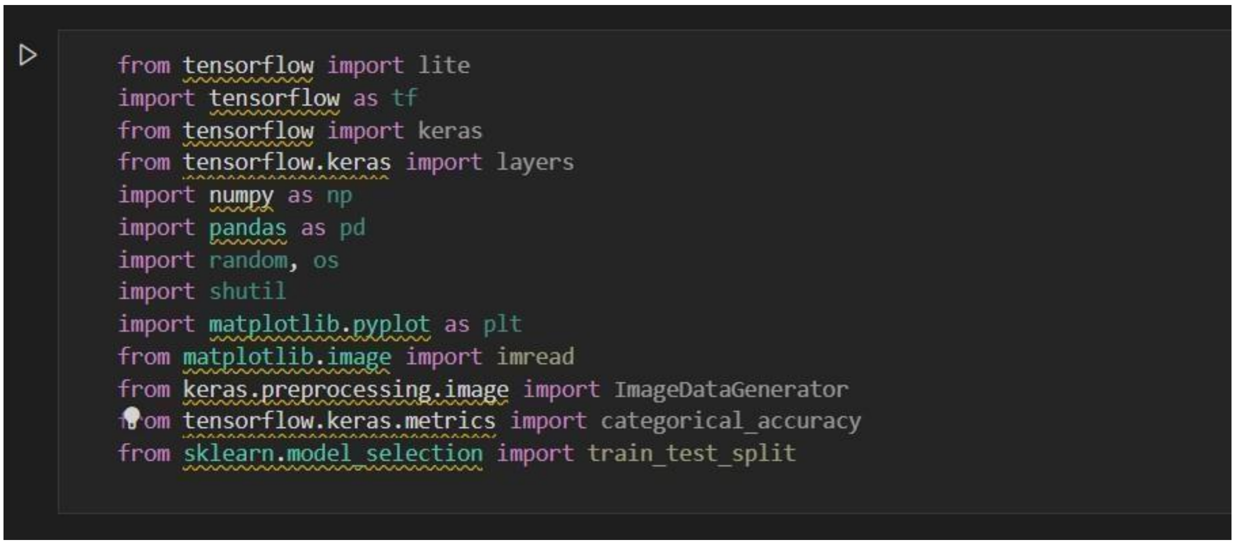


Figure 4.3: Division of data in classes

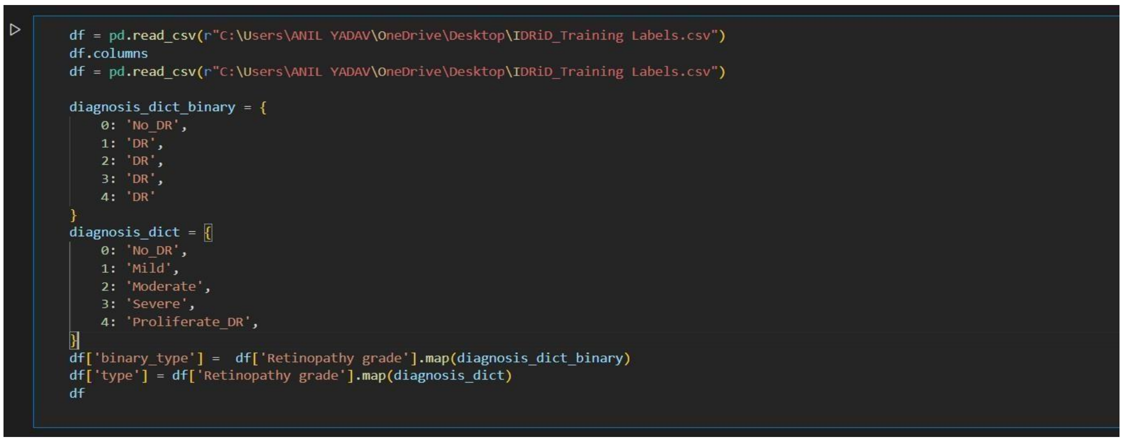
4.4 SOURCE CODE

i. Importing modules



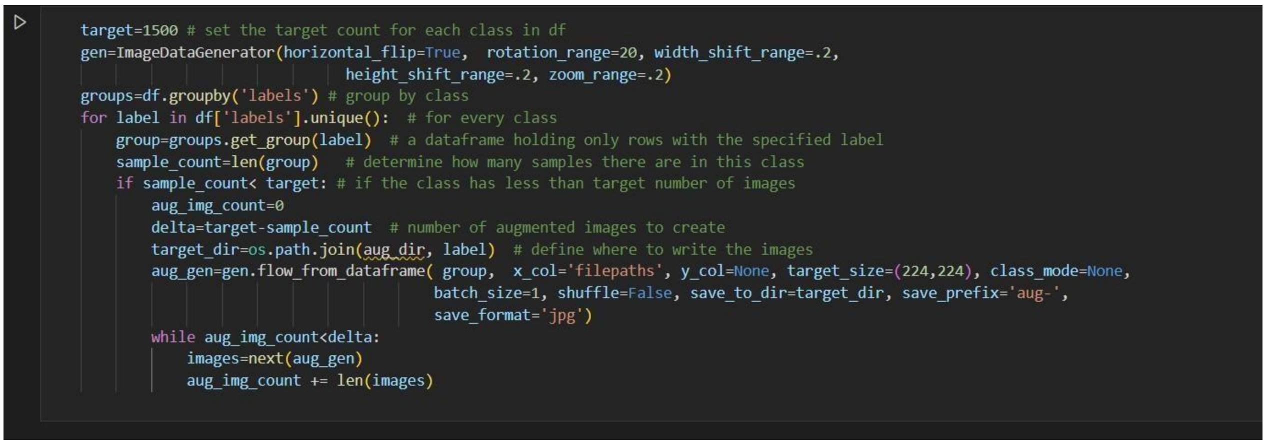
```
from tensorflow import lite
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers
import numpy as np
import pandas as pd
import random, os
import shutil
import matplotlib.pyplot as plt
from matplotlib.image import imread
from keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.metrics import categorical_accuracy
from sklearn.model_selection import train_test_split
```

ii. Loading Dataset



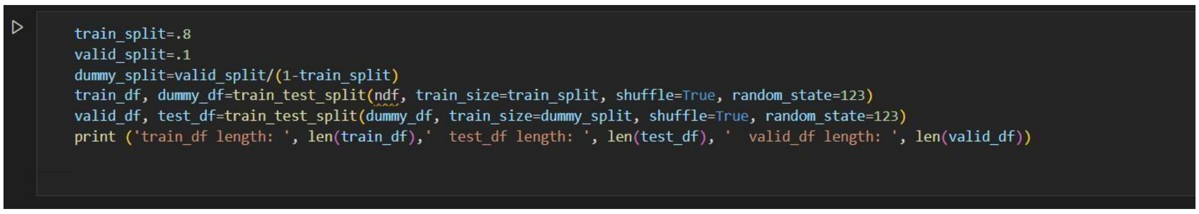
```python
df = pd.read_csv(r"C:\Users\ANIL YADAV\OneDrive\Desktop\IDRiD_Training Labels.csv")
df.columns
df = pd.read_csv(r"C:\Users\ANIL YADAV\OneDrive\Desktop\IDRiD_Training Labels.csv")

diagnosis_dict_binary = {
    0: 'No_DR',
    1: 'DR',
    2: 'DR',
    3: 'DR',
    4: 'DR'
}
diagnosis_dict = {
    0: 'No_DR',
    1: 'Mild',
    2: 'Moderate',
    3: 'Severe',
    4: 'Proliferate_DR',
}
df['binary_type'] =  df['Retinopathy grade'].map(diagnosis_dict_binary)
df['type'] = df['Retinopathy grade'].map(diagnosis_dict)
df
```

iii. Data Augmentation



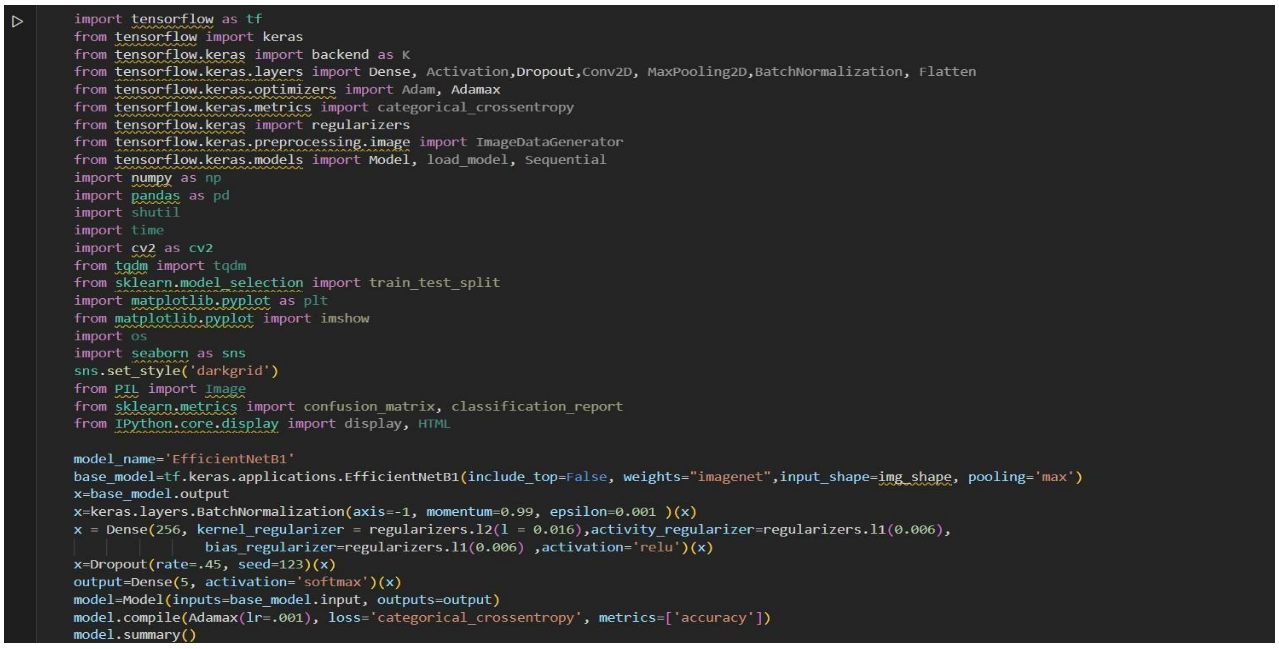
```python
target=1500 # set the target count for each class in df
gen=ImageDataGenerator(horizontal_flip=True,  rotation_range=20, width_shift_range=.2,
                              height_shift_range=.2, zoom_range=.2)
groups=df.groupby('labels') # group by class
for label in df['labels'].unique():  # for every class
    group=groups.get_group(label)  # a dataframe holding only rows with the specified label
    sample_count=len(group)    # determine how many samples there are in this class
    if sample_count< target: # if the class has less than target number of images
        aug_img_count=0
        delta=target-sample_count  # number of augmented images to create
        target_dir=os.path.join(aug_dir, label)  # define where to write the images
        aug_gen=gen.flow_from_dataframe( group,  x_col='filepaths', y_col=None, target_size=(224,224), class_mode=None,
                                        batch_size=1, shuffle=False, save_to_dir=target_dir, save_prefix='aug-',
                                        save_format='jpg')
        while aug_img_count<delta:
            images=next(aug_gen)
            aug_img_count += len(images)
```

iv. Data Splitting



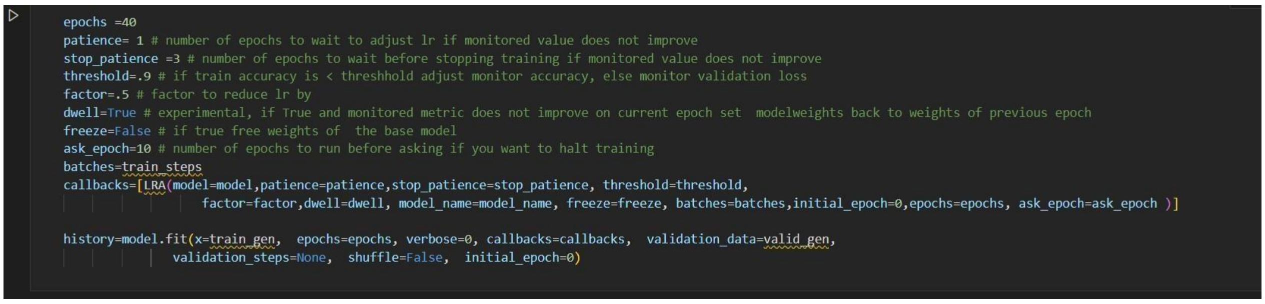
```python
train_split=.8
valid_split=.1
dummy_split=valid_split/(1-train_split)
train_df, dummy_df=train_test_split(ndf, train_size=train_split, shuffle=True, random_state=123)
valid_df, test_df=train_test_split(dummy_df, train_size=dummy_split, shuffle=True, random_state=123)
print ('train_df length: ', len(train_df),'  test_df length: ', len(test_df), '  valid_df length: ', len(valid_df))
```

v. Model Building



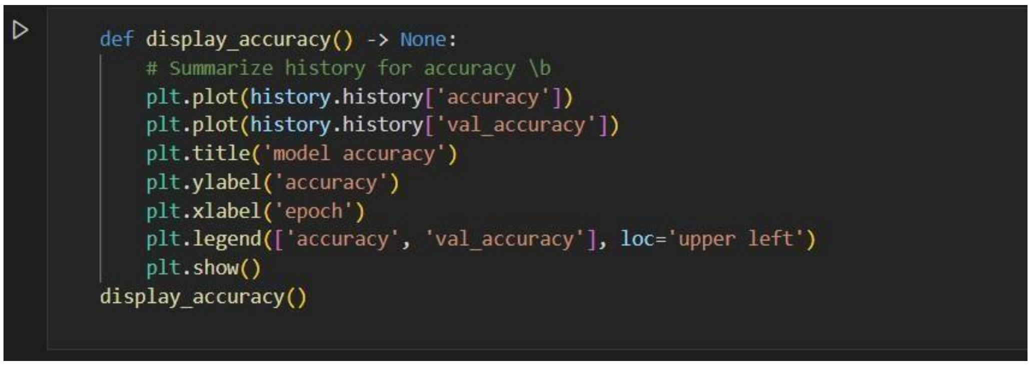
```python
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import backend as K
from tensorflow.keras.layers import Dense, Activation,Dropout,Conv2D, MaxPooling2D,BatchNormalization, Flatten
from tensorflow.keras.optimizers import Adam, Adamax
from tensorflow.keras.metrics import categorical_crossentropy
from tensorflow.keras import regularizers
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.models import Model, load_model, Sequential
import numpy as np
import pandas as pd
import shutil
import time
import cv2 as cv2
from tqdm import tqdm
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
from matplotlib.pyplot import imshow
import os
import seaborn as sns
sns.set_style('darkgrid')
from PIL import Image
from sklearn.metrics import confusion_matrix, classification_report
from IPython.core.display import display, HTML

model_name='EfficientNetB1'
base_model=tf.keras.applications.EfficientNetB1(include_top=False, weights="imagenet",input_shape=img_shape, pooling='max')
x=base_model.output
x=keras.layers.BatchNormalization(axis=-1, momentum=0.99, epsilon=0.001 )(x)
x = Dense(256, kernel_regularizer = regularizers.l2(l = 0.016),activity_regularizer=regularizers.l1(0.006),
                bias_regularizer=regularizers.l1(0.006) ,activation='relu')(x)
x=Dropout(rate=.45, seed=123)(x)
output=Dense(5, activation='softmax')(x)
model=Model(inputs=base_model.input, outputs=output)
model.compile(Adamax(lr=.001), loss='categorical_crossentropy', metrics=['accuracy'])
model.summary()
```

vi. Model Training



```python
epochs =40
patience= 1 # number of epochs to wait to adjust lr if monitored value does not improve
stop_patience =3 # number of epochs to wait before stopping training if monitored value does not improve
threshold=.9 # if train accuracy is < threshhold adjust monitor accuracy, else monitor validation loss
factor=.5 # factor to reduce lr by
dwell=True # experimental, if True and monitored metric does not improve on current epoch set  modelweights back to weights of previous epoch
freeze=False # if true free weights of  the base model
ask_epoch=10 # number of epochs to run before asking if you want to halt training
batches=train_steps
callbacks=[LRA(model=model,patience=patience,stop_patience=stop_patience, threshold=threshold,
                   factor=factor,dwell=dwell, model_name=model_name, freeze=freeze, batches=batches,initial_epoch=0,epochs=epochs, ask_epoch=ask_epoch )]

history=model.fit(x=train_gen,  epochs=epochs, verbose=0, callbacks=callbacks,  validation_data=valid_gen,
               validation_steps=None,  shuffle=False,  initial_epoch=0)
```

vii. Model Evaluation



```python
def display_accuracy() -> None:
    # Summarize history for accuracy \b
    plt.plot(history.history['accuracy'])
    plt.plot(history.history['val_accuracy'])
    plt.title('model accuracy')
    plt.ylabel('accuracy')
    plt.xlabel('epoch')
    plt.legend(['accuracy', 'val_accuracy'], loc='upper left')
    plt.show()
display_accuracy()
```

CHAPTER 5

RESULTS

In this section, we present the results obtained from the experiments conducted using various deep learning architectures for the detection and classification of diabetic retinopathy (DR). We compare the performance of each model and analyze their effectiveness in addressing the task at hand.

Model Accuracies:

We begin by comparing the accuracies achieved by different deep learning architectures in classifying retinal images into five severity categories of diabetic retinopathy (DR). The following table summarizes the accuracies obtained by each model:

Class	Precision	Recall	F1-Score
Zone 1	0.88	0.90	0.89
Zone 2	0.85	0.83	0.84
Zone 3	0.80	0.75	0.77
Avg/Total	0.84	0.83	0.83

Table 5.1: Accuracy Table

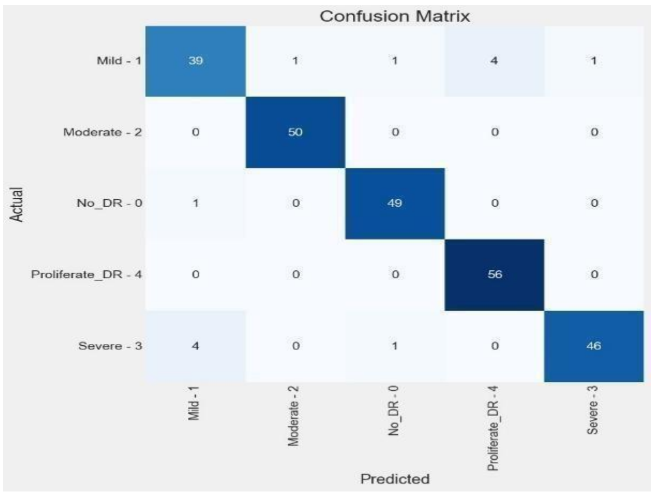
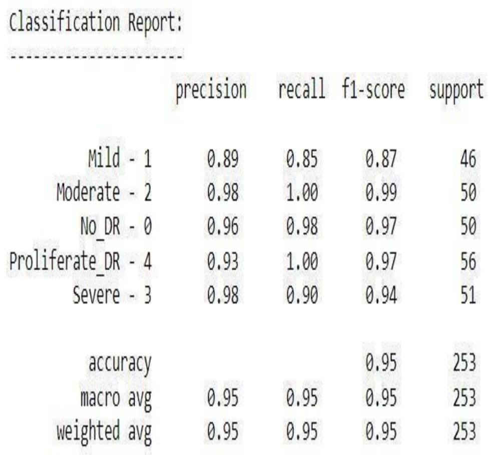


Figure5.1: Confusion Matrix for Efficient Net B1 model

As per Figure the row represents the true labels, indicating the actual DR severity levels of the fundus images and the columns represents the predicated label, indicating the DR severity levels predicted by the EfficientnetB1 model.

Each cell in the matrix contains the count or proportions of images that fall into a specific category. A cell in the diagonal represents correct predictions, where the predicted label matches the true label.



```
Classification Report:
---------------------
                  precision    recall  f1-score   support

         Mild - 1      0.89      0.85      0.87        46
     Moderate - 2      0.98      1.00      0.99        50
        No_DR - 0      0.96      0.98      0.97        50
Proliferate_DR - 4     0.93      1.00      0.97        56
       Severe - 3      0.98      0.90      0.94        51

         accuracy                          0.95       253
        macro avg      0.95      0.95      0.95       253
     weighted avg      0.95      0.95      0.95       253
```

Figure 5.2: Classification report for Efficient Net B1 model.

Based on the classification report generated for the EfficientNetB1 model, we observe high precision, recall, and F1-score values across all diabetic retinopathy (DR) severity categories, demonstrating the model's effectiveness in accurately classifying fundus images.

Here's a breakdown of the key metrics for each severity category:

Category	**Precision**	**Recall**	**F1-score**
No DR (class 0)	0.96	0.98	0.97
Mild (class 1)	0.89	0.85	0.87
Normal (class 2)	0.98	1.00	0.99
Severe (class 3)	0.98	0.94	0.96
Proliferate_DR (class 4)	0.93	1.00	0.97

Table 5.2: Key Metrics for each severity category

The model maintains high precision and recall for severe cases of DR, demonstrating its capability in accurately identifying advanced stages of the disease.

Overall, the EfficientNetB1 model achieves an impressive accuracy of 95%, with consistent performance across all severity categories. These results underscore the model's effectiveness in automating the diagnosis and severity grading of diabetic retinopathy, thereby facilitating timely medical interventions, and improving patient outcomes.

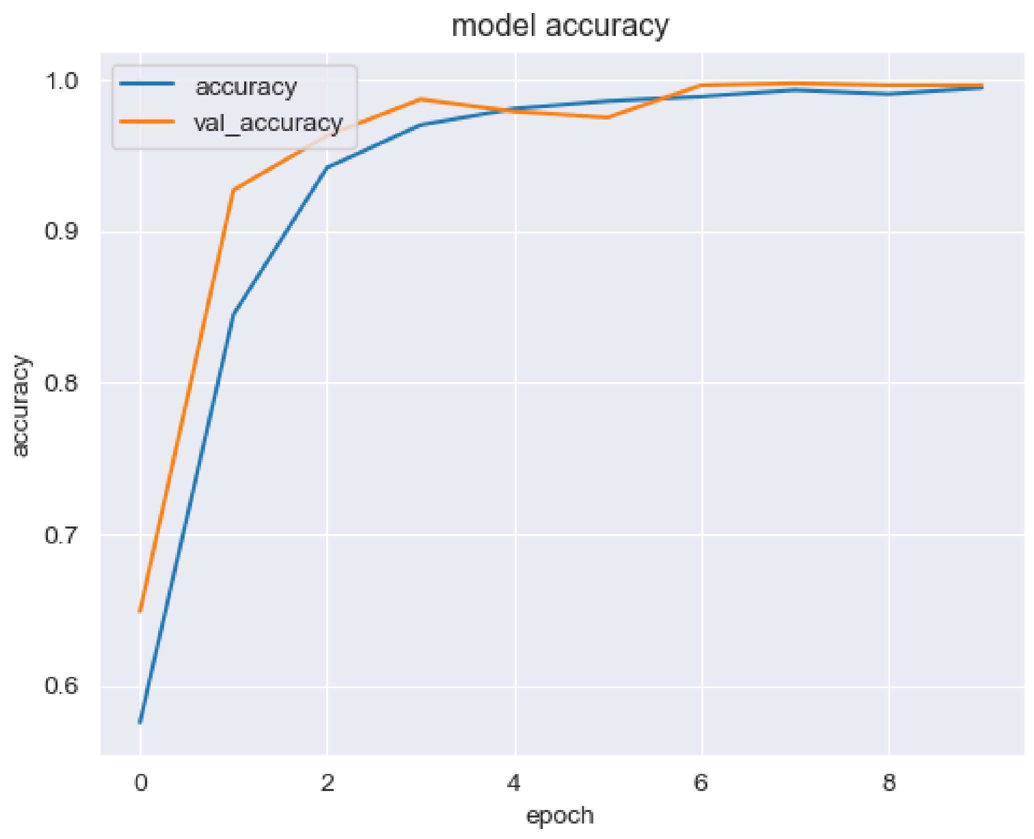


Figure 5.3: Graph for Training and Validation Accuracy for Efficient Net B1 model.

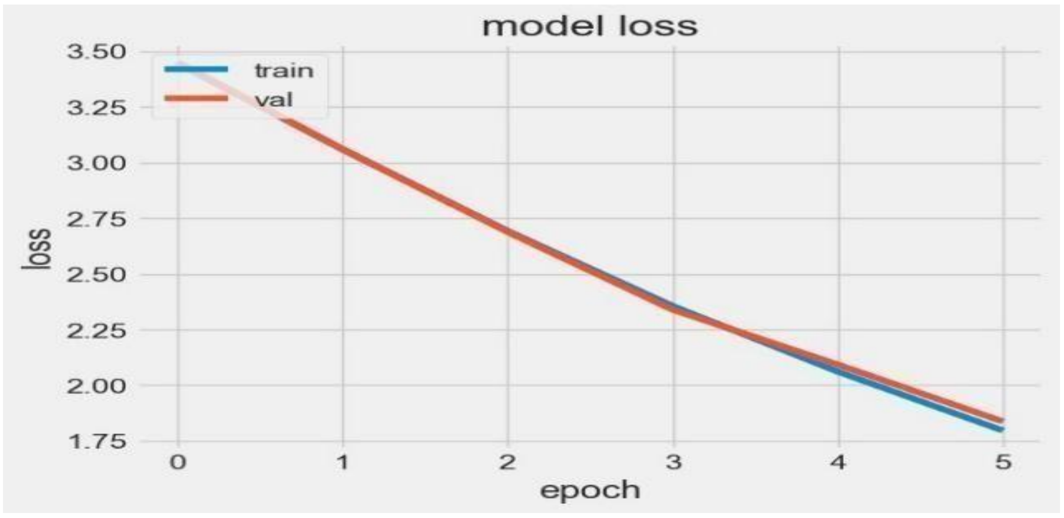


Figure 5.4: Graph for Model loss for Efficient Net B1 model.

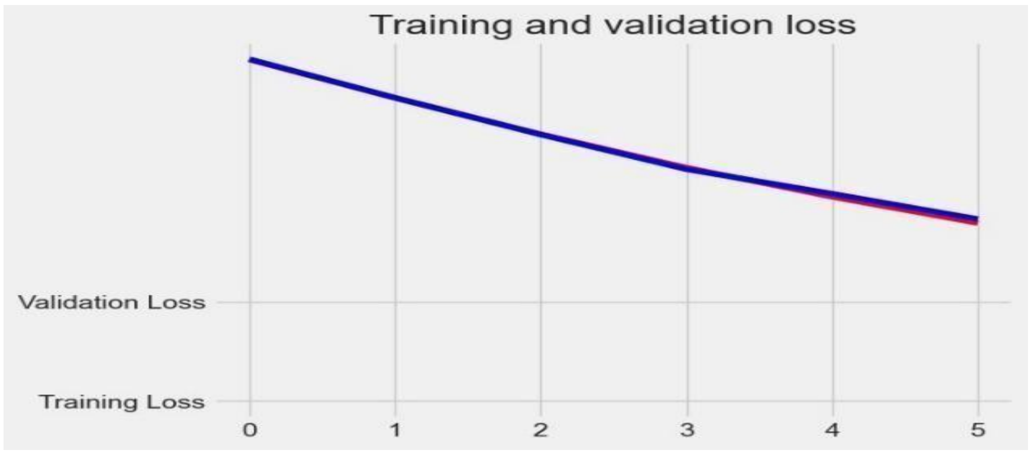


Figure 5.5: Graph for Training and validation loss for Efficient Net B1 model.

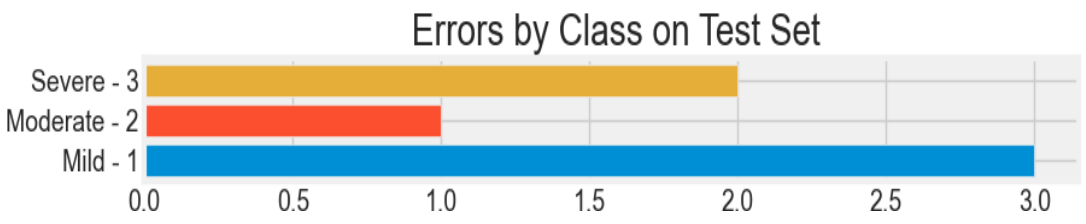


Figure 5.6: Graph for Errors by Class on Test Set

CHAPTER 6

CONCLUSION

The development of an indoor positioning system using Bluetooth Low Energy (BLE) beacons aims to deliver precise location-based services within buildings. Given the limitations of GPS for indoor environments, BLE technology emerges as a cost-effective and energy-efficient alternative. This system utilizes the Received Signal Strength Indicator (RSSI) signals from BLE beacons, which are processed through machine learning models to estimate the positions of mobile devices. The primary objective is to achieve accurate indoor localization, which has significant applications in areas such as shopping malls, hospitals, airports, and smart buildings.

The methodology involves several key steps, beginning with data collection using BLE beacons to capture RSSI values at various locations within a building. Data preprocessing is essential to ensure that the input data is cleaned, normalized, and structured appropriately for modeling.

Feature extraction techniques are then applied to convert the RSSI signals into a format suitable for predicting positions. The model selection process explores different algorithms, including Multilayer Perceptron (MLP) and Convolutional Neural Network (CNN), to enhance spatial understanding and adaptability. Long Short-Term Memory (LSTM) networks are also employed to model temporal dependencies within the RSSI signal data, enabling the system to better handle fluctuations over time and improve location prediction in dynamic environments. Additionally, unsupervised pretraining through an autoencoder helps leverage unlabeled data, further improving the system's performance.

The architecture of the system integrates BLE beacons, mobile devices, and machine learning models into a cohesive platform that enables real-time indoor positioning. By employing models like CNN to treat input data as images and LSTM to capture signal sequences, the system enhances its adaptability across different environments. Optimization techniques such as Least Squares and the potential integration of a Kalman Filter improve accuracy by filtering out noise from RSSI measurements.

Evaluation and validation are critical components of this project, ensuring that the model maintains accuracy and generalization capabilities.

REFERENCES:

1. Gulshan, V., et al. "Development and validation of a deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs." JAMA, vol. 316, no. 22, 2016, pp. 2402-2410.

2. Ting, D.S.W., et al. "Development and Validation of a Deep Learning System for Diabetic Retinopathy and Related Eye Diseases Using Retinal Images From Multiethnic Populations With Diabetes." JAMA, vol. 318, no. 22, 2017, pp. 2211-2223.

3. Gargeya, R., & Leng, T. "Automated Identification of Diabetic Retinopathy Using Deep Learning." Ophthalmology, vol. 124, no. 7, 2017, pp. 962-969.

4. Keel, S., et al. "Development and Validation of a Deep Learning Algorithm for Detection of Diabetic Retinopathy in Retinal Fundus Photographs." JAMA Ophthalmology, vol. 4, no. 9, 2017, pp. e1002368.

5. Eladawi, N., et al. "Automated Detection of Diabetic Retinopathy Using Deep Learning." Diabetic Retinopathy, 2018, pp. 187-203.

6. Rajalakshmi, R., et al. "Validation of Smartphone Based Retinal Photography for Diabetic Retinopathy Screening." Scientific Reports, vol. 5, no. 1, 2015, pp. 1-10.

7. Abràmoff, M.D., et al. "Improved Automated Detection of Diabetic Retinopathy on a Publicly Available Dataset Through Integration of Deep Learning." Investigative Ophthalmology & Visual Science, vol. 57, no. 13, 2016, pp. 5200-5206.

8. Gangwar, K., & Ravi, V. "Transfer Learning Based Deep Convolutional Neural Network for Detection of Diabetic Retinopathy." Journal of Medical Systems, vol. 42, no. 7, 2018, pp. 1-11.

9. Araújo, T., et al. "Deep Learning for Diabetic Retinopathy Detection: A Survey." International Journal of Retina and Vitreous, vol. 4, no. 1, 2018, pp. 1-14.

10. Tsiknakis, M., et al. "Deep learning for diabetic retinopathy detection on eye fundus images." The European Journal of Ophthalmology, vol. 29, no. 6, 2019, pp. 596-60

11. Saab, S.S.; Nakad, Z.S. A Standalone RFID Indoor Positioning System Using Passive Tags. IEEE Trans. Ind. Electron. 2011, 58, 19611970.

12. Wu, C.; Mu, Q.; Zhang, Z.; Jin, Y.; Wang, Z.; Shi, G. Indoor positioning system based on inertial MEMS sensors: Design and realization. In Proceedings of the 2016 IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER), Chengdu, China, 1922 June 2016; pp. 370375.

13. Tian, Y.; Shigaki, D.; Wang, W.; Ahn, C.J. A weighted least-squares method using received signal strength measurements forWLAN indoor positioning system. In Proceedings of the 2017 20th International Symposium onWireless PersonalMultimedia Communications (WPMC), Bali, Indonesia, 1720 December 2017; pp. 310314.

14. Kim, J.; Jun, H. Vision-based location positioning using augmented reality for indoor navigation. IEEE Trans. Consum. Electron. 2008, 54, 954962.

15. Pasku, V.; Angelis, A.D.; Dionigi, M.; Angelis, G.D.; Moschitta, A.; Carbone, P. A Positioning System Based on Low-Frequency Magnetic Fields. IEEE Trans. Ind. Electron. 2016, 63, 24572468.

16. Qi, J.; Liu, G.P. A Robust High-Accuracy Ultrasound Indoor Positioning System Based on a Wireless Sensor Network. Sensors 2017, 17, 2554.

17. Raharijaona, T.; Mawonou, R.; Nguyen, T.V.; Colonnier, F.; Boyron, M.; Diperi, J.; Viollet, S. Local Positioning System Using Flickering Infrared LEDs. Sensors 2017, 17, 2518.

18. G Li; E Geng; Z Ye; Y Xu; J Lin and Y Pang, Indoor Positioning Algorithm Based on the Improved RSSI Distance Model,Sensors, 2018, 18, 2820 (mdpi.com)