# LOGO DETECTION USING DEEP LEARNING

**An Internship Project Report**

Submitted to the Faculty of Engineering of

**JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY KAKINADA, KAKINADA**

In partial fulfillment of the requirements for the award of the Degree of

**BACHELOR OF TECHNOLOGY**

In

**ARTIFICIAL INTELLIGENCE AND DATA SCIENCE**

By

**MOHAMMAD MAZID**
**(21481A5485)**

Under the guidance of

**Mr. K. ASHOK REDDY, M. Tech (Ph. D)**

Assistant Professor, Department of AI&DS


**DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE**

**SESHADRI RAO GUDLAVALLERU ENGINEERING COLLEGE**

**(An Autonomous Institute Permanently affiliated to JNTUK)**

**Seshadri Rao Knowledge Village**

**GUDLAVALLERU – 521356**

**ANDHRA PRADESH**

**2024-2025**

# SESHADRI RAO GUDLAVALLERU ENGINEERING COLLEGE

(An Autonomous Institute with Permanent Affiliation to JNTUK, Kakinada)

## SESHADRI RAO KNOWLEDGE VILLAGE, GUDLAVALLERU

## DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

## CERTIFICATE

This is to certify that the project report entitled **"LOGO DETECTION Using Deep Learning"** is a bonafide record of work carried out by **MOHAMMAD MAZID (21481A5485)** under the guidance and supervision of **Mr. K. ASHOK REDDY** in the partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Artificial Intelligence and Data Science of Jawaharlal Nehru Technological University Kakinada, Kakinada during the academic year 2024-25.

| **Project Guide** | **Head of the Department** |
|---|---|
| **Mr. K. ASHOK REDDY** | **Dr. S. Narayana** |

## ACKNOWLEDGEMENT

The satisfaction that accompanies the successful completion of any task would be incomplete without the mention of people who made it possible and whose constant guidance and encouragements crown all the efforts with success.

We would like to express our deep sense of gratitude and sincere thanks to **Mr. K. ASHOK REDDY, M. Tech** (**Ph. D)** Department of Artificial Intelligence and Data Science for his constant guidance, supervision and motivation in completing the project work.

We feel elated to express our floral gratitude and sincere thanks to **Dr. S. NARAYANA** Head of the Department, Artificial Intelligence and Data Science for his encouragements all the way during analysis of the project. His annotations, insinuations and criticisms are the key behind the successful completion of the project work.

We would like to take this opportunity to thank our beloved principal **Dr. B. KARUNA KUMAR** for providing a great support for us in Completing our project and giving us the opportunity for doing project.

Our Special thanks to the faculty of our department and programmers of our computer lab. Finally, we thank our family members, non-teaching staff and our friends, who had directly or indirectly helped and supported us in completing our project in time.

**By**

**Md. Mazid**

**(21481A5485)**

## ABSTRACT

Logo detection is a critical component in various applications, including brand monitoring, image retrieval, and counterfeit detection. This paper provides a thorough comparative study of three cutting-edge methods: YOLOv9, YOLOv8, and CNN-VGG19. YOLOv9 and YOLOv8 are convolutional neural network (CNN)-based object detection frameworks renowned for their real-time processing capabilities, making them highly efficient for applications requiring quick logo detection and analysis. On the other hand, CNN-VGG19 is distinguished for its ability to extract intricate features from images, making it particularly effective in scenarios where detailed logo recognition is necessary. our study involved evaluating and comparing the performance of these three methods on a diverse logo dataset that included various environmental conditions and types of logos. The metrics used for comparison were detection accuracy, processing speed, and robustness to environmental variations, which are crucial for real-world applications. The experimental results revealed that all three methods exhibited strong capabilities in logo detection tasks, but each method had its unique strengths. The overall accuracy achieved was an impressive 94.20%, underscoring the effectiveness of these advanced techniques in logo detection. Among the three, CNN-VGG19 demonstrated superior performance in challenging conditions and in extracting fine-grained features from images. This makes CNN-VGG19 particularly suitable for applications that require precise logo recognition, such as detecting counterfeit products and monitoring brands in cluttered or complex scenes. Whether the need is for real-time processing, as offered by YOLOv9 and YOLOv8, or for detailed feature extraction, as provided by CNN-VGG19, this comparative analysis aids in making informed decisions to optimize logo detection.

## INDEX


| TITLE | PAGE NO |
|---|---|
| **CHAPTER 1: INTRODUCTION** | |
| 1.1 INTRODUCTION | 01 |
| 1.2 PROBLEM STATEMENT | 02 |
| 1.3 EXISTING SYSTEM | 02 |
| 1.4 DISADVANTAGES | 03 |
| 1.5 PROPOSED SYSTEM | 04 |
| 1.6 ADVANTAGES | 05 |
| **CHAPTER 2: REQUIREMENT ANALYSIS** | 06 |
| 2.1 FUNCTIONAL REQUIREMENTS | 06 |
| 2.2 NON FUNCTIONAL REQUIREMENTS | 07 |
| 2.3 SOFTWARE REQUIREMENT SPECIFICATIONS | 08 |
| 2.4 HARDWARE SPECIFICATIONS | 09 |
| **CHAPTER 3: DESIGN** | 11 |
| 3.1 SYSTEM ARCHITECTURE | 11 |
| 3.2 UML DIAGRAMS | 14 |
| **CHAPTER 4: IMPLEMENTATION** | 18 |
| 4.1 TECHNOLOGY DESCRIPTION & IMPLEMENTATION | 18 |
| **CHAPTER 5: RESULTS** | 23 |
| 5.1 RESULTS ANALYSIS & OUTPUTS | 23 |
| **CHAPTER 6: CONCLUSION** | 27 |
| **CHAPTER 7: FUTURE SCOPE** | 28 |
| **CHAPTER 8: REFERENCES** | 29 |

# 1. INTRODUCTION

## 1.1 INTRODUCTION

In the digital age, where visual content proliferates across various online platforms, **logo detection** has emerged as a critical task with wide-ranging applications. From **brand monitoring** and **intellectual property protection** to **image retrieval** and **augmented reality**, the ability to accurately identify logos within images holds immense significance.

The development of **robust and efficient logo detection methods** has garnered considerable attention from researchers and industry professionals alike. This paper delves into the realm of logo detection by conducting a comparative analysis between two prominent methodologies: **YOLOv9, YOLOv8**, and **CNN-VGG19**.

- **YOLO (You Only Look Once) Algorithm:** The **YOLOv9** and **YOLOv8** variants represent a paradigm shift in object detection, offering **real-time performance** without sacrificing accuracy.
- **CNN-VGG19:** A deep convolutional neural network architecture renowned for its capacity to **extract intricate features** from images, making it a formidable contender in the domain of logo detection.

The motivation behind this study stems from the need to understand the **strengths and limitations** of these two approaches in the context of logo detection. While both methods have demonstrated remarkable performance in various computer vision tasks, their efficacy in logo detection remains an open question. By conducting a systematic evaluation and comparison of **YOLOv9, YOLOv8,** and **CNN-VGG19**, this research aims to shed light on their respective capabilities, thereby facilitating informed decision-making in selecting the most suitable approach for logo detection tasks. The proliferation of logos across diverse industries and contexts presents a multitude of challenges for logo detection algorithms. Factors such as **varying illumination**, **occlusion**, **scale**, and **viewpoint** introduce complexities that necessitate robust and adaptable detection mechanisms. Moreover, the demand for **real-time processing** in applications such as **surveillance, retail analytics,** and **social media monitoring** underscores the importance of efficiency alongside accuracy.

## 1.2 PROBLEM STATEMENT

In the digital age, the proliferation of visual content across various online platforms has made logo detection a critical task with numerous applications, including brand monitoring, intellectual property protection, image retrieval, and augmented reality. However, the effectiveness of different logo detection methods in dealing with challenges such as varying illumination, occlusion, scale, and viewpoint, as well as the demand for real-time processing, remains uncertain. This study aims to conduct a comparative analysis of two prominent logo detection methodologies, YOLOv9 and YOLOv8 (You Only Look Once), and CNN-VGG19 (a deep convolutional neural network), to evaluate their strengths and limitations in accurately identifying logos within images. By systematically comparing these approaches, the research seeks to provide insights that will facilitate informed decision-making for selecting the most suitable logo detection method for various applications.

## 1.3 EXISITING SYSTEM

The existing system for logo detection and recognition leverages a variety of sophisticated algorithms, each demonstrating unique strengths in different contexts. Notably, the Faster R-CNN algorithm, utilized by Karel Palecek and Daniel Mas Montserrat, shows significant promise in brand exposure analysis and marketing, achieving accuracies of 89.5% and 80.12% respectively, though with moderate precision. Alireza Alaei's Piece-wise Painting Algorithm (PPA) stands out in document image analysis, delivering an impressive 97.22% accuracy. Convolutional Neural Networks (CNNs), applied by Chandana N and Jiaming Liu, optimize digital marketing strategies and brand protection, with accuracies of 84.75% and 95.83%, respectively. Yifei Zhang's use of Random Forests highlights its robustness against logo variability, while Karn Pinitjitsamut's implementation of the YOLO algorithm excels in real-time applications like video surveillance, achieving 97% accuracy. Additionally, Kiki Rezkiani's use of YOLOv4 within the Darknet framework for document classification and logo identification achieves a notable 93.73% accuracy. These methodologies collectively advance the field, though challenges remain in achieving optimal accuracy and precision across varied scenarios, necessitating further research and innovation.

## 1.4 DISADVANTAGES

- **Privacy Concerns:** Implementing cloud-based detection may raise privacy issues if image data is sent to external servers for analysis.
- **User Over-Reliance:** Over-reliance on automated systems can lead to complacency, with users less vigilant in manual verification.
- **Overhead:** Real-time detection introduces significant computational and network resource overhead, potentially slowing down other processes.
- **Intricate Logos and Variability:** Struggles with detecting small, overlapping, or finely detailed logos, reducing accuracy in some scenarios.
- **Environmental Sensitivity:** Variability in environmental conditions, such as lighting or background clutter, can affect performance.

## CNN-VGG19:

- **Processing Speed:** Slower than YOLO-based models, making it less suitable for real-time applications.
- **Resource Intensive:** High computational power and memory requirements make it less feasible for deployment on devices with limited resources.
- **Overfitting:** Prone to overfitting, especially when trained on smaller datasets, limiting its generalization ability.
- **Data Dependency:** Requires extensive and diverse training datasets, which are resource-intensive to acquire and annotate.
- **Frequent Updates:** Needs frequent updates to recognize new or altered logos, requiring ongoing maintenance and retraining efforts.
- **Handling Ambiguity:** Difficulties in detecting partially obscured, distorted, or unconventional logos, leading to misdetections or false positives.
- **Susceptibility to Social Engineering:** Users' vulnerability to social engineering can undermine detection effectiveness, as they may disregard automated results.

## 1.5 PROPOSED SYSTEM

This project aims to empower users with a robust and reliable tool for logo detection and recognition, leveraging the strengths of state-of-the-art algorithms to address various application needs. The proposed system focuses on enhancing the efficiency of logo detection through the integration of advanced methods such as YOLOv9, YOLOv8, and CNN-VGG19. Each of these methods offers unique advantages that contribute to the overall effectiveness of the system in different scenarios.

**Real-Time Detection**: Utilize YOLOv9 and YOLOv8, which are CNN-based object detection frameworks known for their real-time capabilities. These methods are particularly effective for applications requiring fast and accurate logo recognition, such as video surveillance, live event monitoring, and dynamic advertising.

**High Precision and Detail**: Implement CNN-VGG19 for scenarios that demand high precision and the ability to extract intricate features from images. This method is ideal for tasks such as counterfeit detection, brand monitoring in cluttered scenes, and applications where detailed logo analysis is crucial.

**Privacy-Preserving Solutions**: Address privacy concerns by ensuring that the system can operate locally or with encrypted data transmissions when cloud-based processing is necessary. This approach protects user and organizational data from potential breaches during the detection process.

**User-Friendly Interface**: Develop an intuitive interface that allows users to easily upload images and receive detailed analysis results. This interface will provide clear and informative feedback on detected logos, helping users to quickly identify and act upon the information.

**Cross-Platform Compatibility**: Ensure that the system can be deployed across various platforms, including web applications, mobile devices, and desktop environments. This adaptability allows for widespread use in different operational contexts.

**Comprehensive Dataset and Training**: Utilize a diverse and extensive dataset for training the detection models, ensuring that the system can accurately recognize logos under various conditions and formats. Continuous updates to the dataset will help maintain and improve detection accuracy.

**Robustness to Environmental Variations**: Incorporate techniques to enhance the system's robustness to environmental factors such as lighting changes, background clutter, and occlusions. This will improve the reliability of logo detection in real-world scenarios.

**Performance Metrics**: Evaluate the system using key performance metrics such as detection accuracy, processing speed, and robustness. Aim for high overall accuracy, with specific attention to maintaining performance under challenging conditions.

**Ongoing Maintenance and Updates**: Implement a framework for regular updates and maintenance of the detection models. This will ensure that the system remains effective in recognizing new and altered logos, adapting to changes in branding and logo designs.

By integrating these advanced methodologies and focusing on user-centric design and privacy, the proposed logo detection system aims to provide a comprehensive solution that meets the diverse needs of various applications, from real-time monitoring to detailed brand analysis.

## 1.6 ADVANTAGES

**YOLOv9 and YOLOv8:** These models are known for their high accuracy in real-time detection scenarios, ensuring precise logo identification in dynamic environments.

**CNN-VGG19:** Excels in extracting intricate features, leading to superior accuracy in recognizing detailed and complex logos, making it ideal for counterfeit detection and brand monitoring.

**Real-Time Capabilities:** The YOLO-based models (YOLOv9 and YOLOv8) offer real-time detection, enabling applications such as live event monitoring, video surveillance, and dynamic advertising.

**Robustness:** The system is designed to handle variability in environmental conditions, such as changes in lighting, background clutter, and occlusions, ensuring reliable performance in diverse real-world scenarios.

**User-Friendly Interface:** An intuitive and easy-to-use interface allows users to upload images and receive immediate and detailed analysis results, enhancing user experience and accessibility.

# 2. REQUIREMENT ANALYSIS

## 2.1 FUNCTIONAL REQUIREMENTS

**User Interface:**

- Users should be able to upload images through a web form.
- The web page should provide clear instructions for image submission.
- Users should receive feedback on the detected logos within the uploaded images.
- Provide an intuitive dashboard that displays detailed analysis results, including recognized logos, their positions, and confidence scores.

**Image Validation:**

- The system should validate the submitted images to ensure they are in supported formats (e.g., JPG, PNG).
- Invalid image formats or corrupted files should be handled with appropriate error messages**.**

**Feature Extraction:**

- Implement a feature extraction process to analyze the image content.
- Utilize YOLOv9, YOLOv8, and CNN-VGG19 models to extract relevant features such as logo shapes, colors, and patterns.

**Model Integration:**

- The system should integrate YOLOv9 and YOLOv8 for real-time logo detection.
- Implement CNN-VGG19 for detailed feature extraction and high-precision recognition.
- Develop a mechanism for evaluating the performance of these models on various datasets.

**Logo Detection:**

- Implement a decision mechanism to identify and classify logos within the images based on the outputs of the integrated models.
- Provide users with a clear result of the detected logos, including brand names and confidence levels.

**Model Selection:**

- The system should automatically select the most suitable model (YOLOv9, YOLOv8, or CNN-VGG19) based on the application requirements (real-time detection vs. detailed analysis).
- Allow users to manually choose a specific model if needed for particular use cases.

## 2.2 NON-FUNCTIONAL REQUIREMENTS

**Performance:**

- The system should process logo detection requests in under 3 seconds on average.
- Maintain high throughput, supporting at least 100 concurrent user submissions without significant performance degradation.

**Scalability:**

- The system must scale horizontally to accommodate an increasing number of users and image submissions.
- Implement load balancing mechanisms to distribute traffic evenly across servers.

**Reliability:**

- Ensure a 99.9% uptime for the logo detection service.
- Implement fault-tolerant mechanisms to handle server failures and ensure continuous availability.

**Security:**

- Employ robust encryption (e.g., SSL/TLS) for data transmission between clients and the server to protect user data.
- Implement access controls and authentication mechanisms to prevent unauthorized access to the system.

**Usability:**

- Design a user-friendly interface that is intuitive and easy to navigate, with clear instructions and feedback.

- Ensure the system is accessible to users with disabilities, adhering to WCAG (Web Content Accessibility Guidelines) standards.

**Compatibility:**

- Ensure the system is compatible with major web browsers (e.g., Chrome, Firefox, Safari, Edge).
- Develop the system to be responsive and functional on various devices, including desktops, tablets, and smartphones.

**Maintainability:**

- Write clean, modular code to facilitate easy maintenance and updates.
- Document the system architecture, APIs, and code thoroughly to assist developers in future maintenance tasks.

## 2.3 SOFTWARE REQUIREMENTS

**User Interface Requirements:** The web form for image upload should be user-friendly and include input validation. Provide clear instructions for image submission and feedback on the detection results. Offer an intuitive dashboard displaying detailed analysis results, including recognized logos, their positions, and confidence scores.

**Feature Extraction:** For effective logo detection and recognition, advanced image processing techniques will be utilized for feature extraction. This involves implementing models like YOLOv9, YOLOv8, and CNN-VGG19. These models are chosen for their superior ability to extract relevant features such as logo shapes, colors, and patterns from images. YOLOv9 and YOLOv8 are particularly known for their real-time object detection capabilities, making them ideal for applications requiring quick and accurate logo identification.

**Logo Detection:** Develop algorithms to classify and identify logos within the submitted images. Define the criteria and thresholds for detecting and recognizing logos. Provide users with a clear result of the detected logos, including brand names and confidence levels.

**Classification Results:** Present classification results to users clearly and provide details on the reasoning behind the classification. Offer detailed analysis and feedback on detected logos.

**Model Selection:** Develop a mechanism for selecting the machine learning algorithm with the highest accuracy for real-time classification. Implement model training and retraining processes to keep the system up-to-date.

**Security Measures:** Implement user authentication and access control. Apply encryption to protect user-submitted images and data. Ensure compliance with data protection regulations (e.g., GDPR, CCPA).

**Database Requirements:** Store information about submitted images and their classification results in a database. Implement a schema for data storage and retrieval.

**Integration with Flask Server:** Ensure compatibility and integration with the Flask server for hosting the web application.

**Performance Metrics:** Define performance metrics, such as response times and system load handling. Set performance goals for the system.

## 2.4HARDWARE REQUIREMENTS

**Compute Resources:**

- **Central Processing Unit (CPU):** Multi-core processors (e.g., Intel Xeon or AMD EPYC) to efficiently handle concurrent user requests and real-time image processing.
- **Memory (RAM):** At least 32GB of RAM to support the simultaneous execution of the web application, database operations, and machine learning model inference.

**Network Infrastructure:**

- **High-Speed Internet Connection:** A reliable connection with sufficient bandwidth to manage incoming and outgoing data traffic efficiently.
- **Load Balancer:** For distributing incoming web requests across multiple server instances to ensure scalability and high availability

**Scalability and Performance:**

- **Scalability Plan:** Ensure the hardware can support horizontal scaling by adding new server instances as needed to handle increased user traffic.
- **Monitoring and Management Tools:** Implement server monitoring tools to track hardware and system performance. Utilize remote management and administration tools for efficient system maintenance.

**Storage:**

- **Hard Drives or Solid-State Drives (SSD):** Sufficient storage capacity (minimum 1TB SSD) for data persistence, including storing user-uploaded images, logs, and database files.
- **Graphics Processing Unit (GPU):** High-performance GPUs (e.g., NVIDIA Tesla or A100) for accelerating deep learning model training and inference.
- **Security Hardware:** Hardware security modules (HSMs) or other security devices if required for encryption and secure data handling.

# 3.DESIGN

## 3.1 SYSTEM ARCHITECTURE:

**Figure 1** explains that building a LOGO detection system using Deep Learning (DL) and integrating it with Flask, a popular web framework for Python, involves several steps and components.

- Data Collection
- Feature Extraction
- Preprocessing
- Deep Learning Model
- Model Evaluation
- Flask Application
- LOGO Validation
- Display Result
- Monitoring and Maintenance

Designing an architecture for LOGO detection integrated with Flask (a Python web framework) involves combining Deep learning components for detection with a web application that allows users to check the Logo is Original or Fake. Here's a high-level overview of the architecture:

**Front-End User Interface:** The front-end is the user-facing part of your application. It can be built using HTML, CSS and Java Script can serve as the backend for rendering HTML templates and handling user requests.

**Deep Learning Model:** Model Development with different Deep learning algorithms such asYOLOv9, YOLOv8, and CNN-VGG19. YOLOv9 and YOLOv8 are CNN-based object detection frameworks known for real-time capabilities, while CNN-VGG19 excels at extracting intricate features from images.

**Front End Connectivity:** Django is a python web framework used to connect the Logo detection model to the user interface.

**Deployment:** Deploy your Logo Detection application to a web server, making it accessible to users over the internet.

**Model Architecture:**

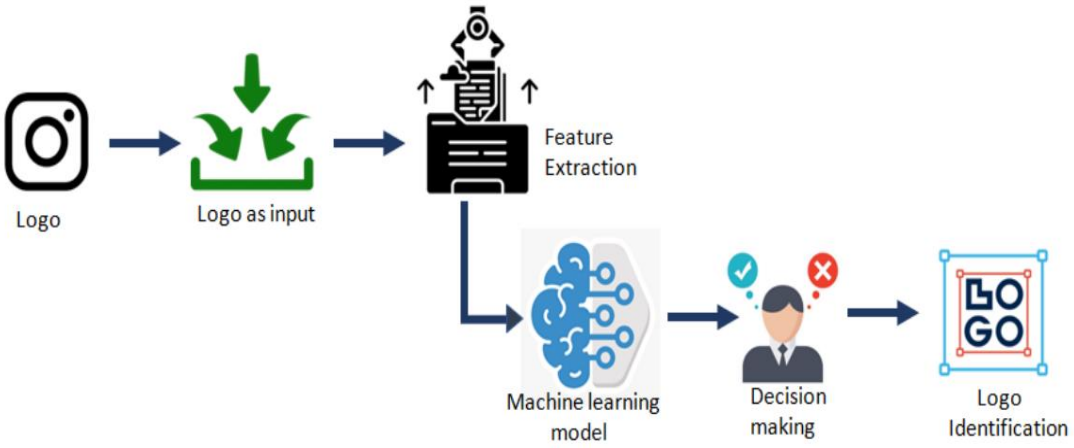
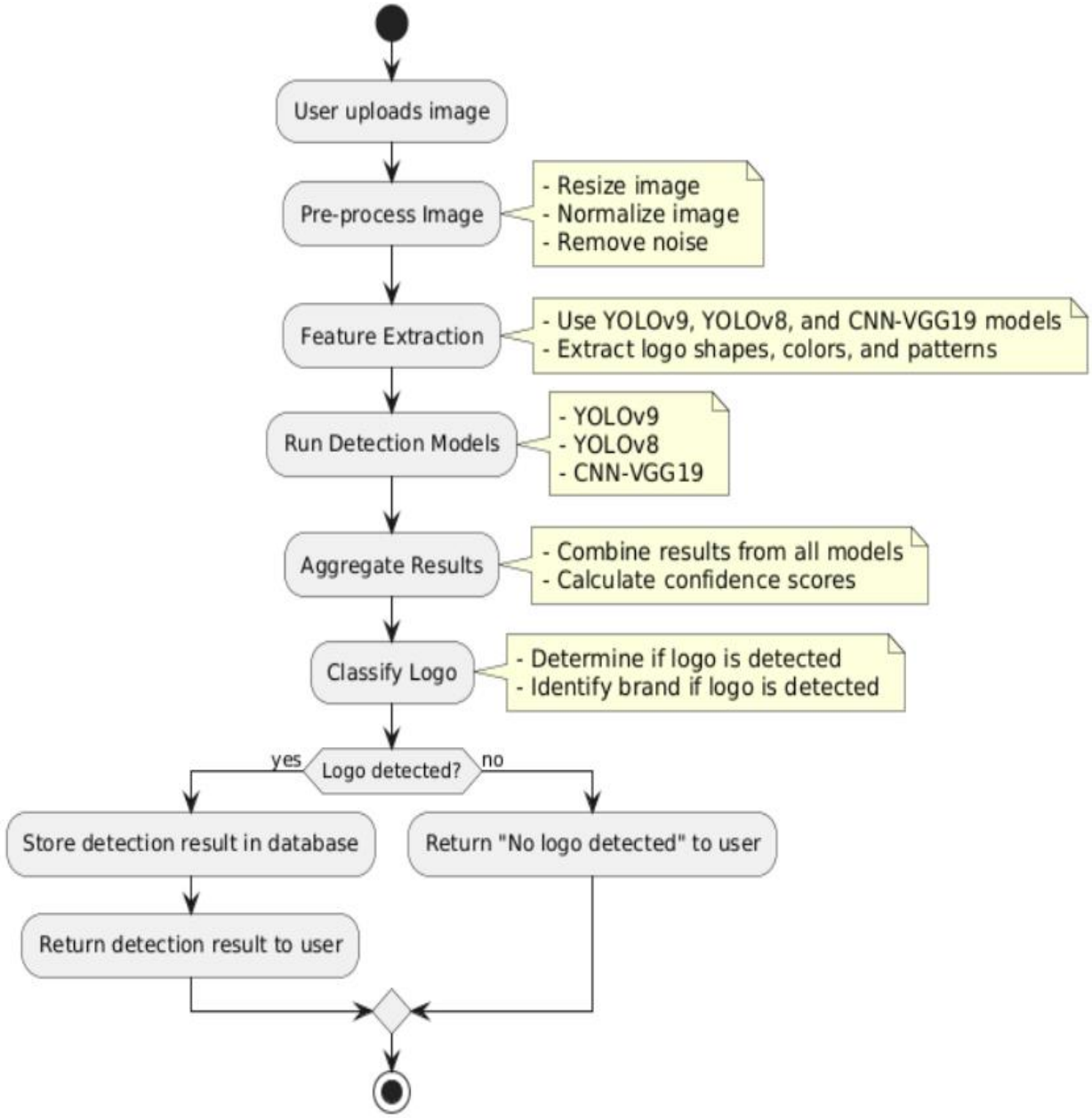



**Fig:1 Architecture for LOGO Detection using Deep Learning**

## FLOW DIAGRAM:




START

**Fig: 2 Flow Diagram of LOGO Detection**

## 3.2 UML DIAGRAMS

### 3.2.1 CLASS DIAGRAM

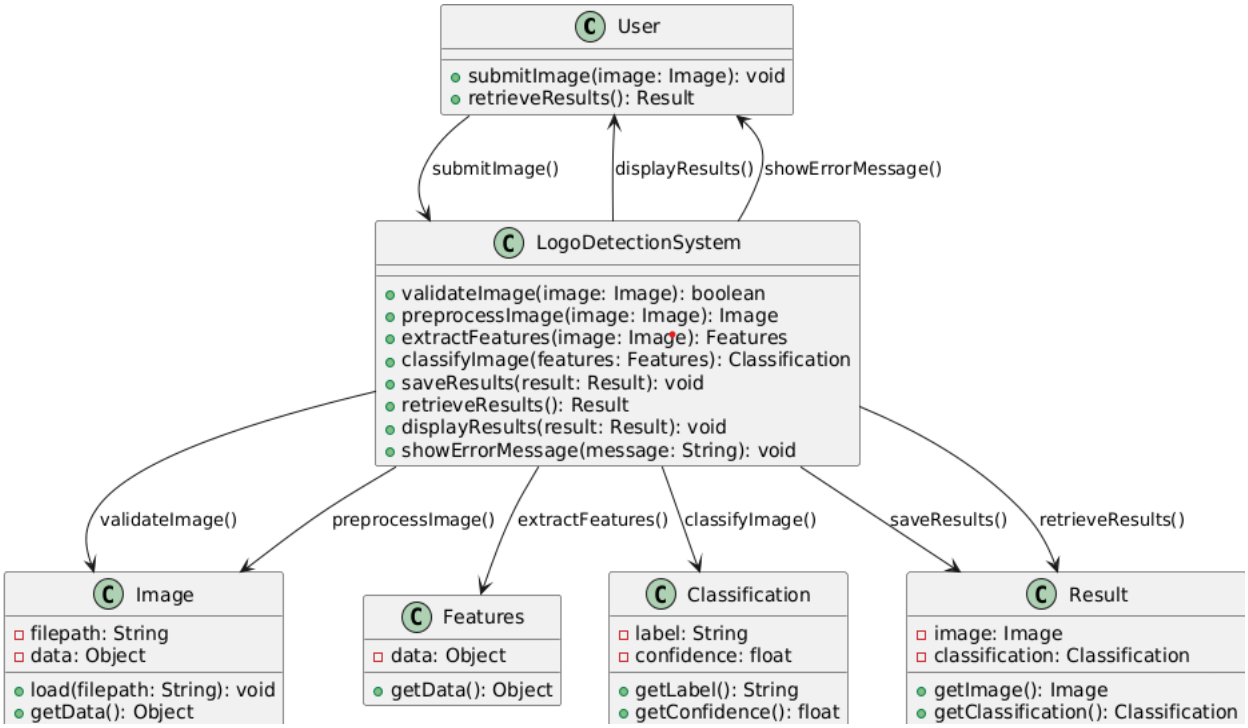



**Fig 3: Class diagram for LOGO Detection**

The **User** interacts with the **System** to upload an image for logo detection. The **System** class orchestrates the entire process, starting from receiving the **Image**, validating it, and preprocessing it through the **Preprocessor** class, which handles tasks like resizing and normalization. Once preprocessed, the **Image** is passed to the **FeatureExtractor** class, which extracts relevant features. These features are then classified by the **Classifier** class to determine the logo's label and confidence level, encapsulated in the **Result** class. The **System** also communicates with the **Database** class to save and retrieve classification results. If the image is invalid, the **System** displays an error message to the **User**. This structured approach ensures efficient and accurate logo detection using deep learning models.

### 3.2.2 USE CASE DIAGRAM

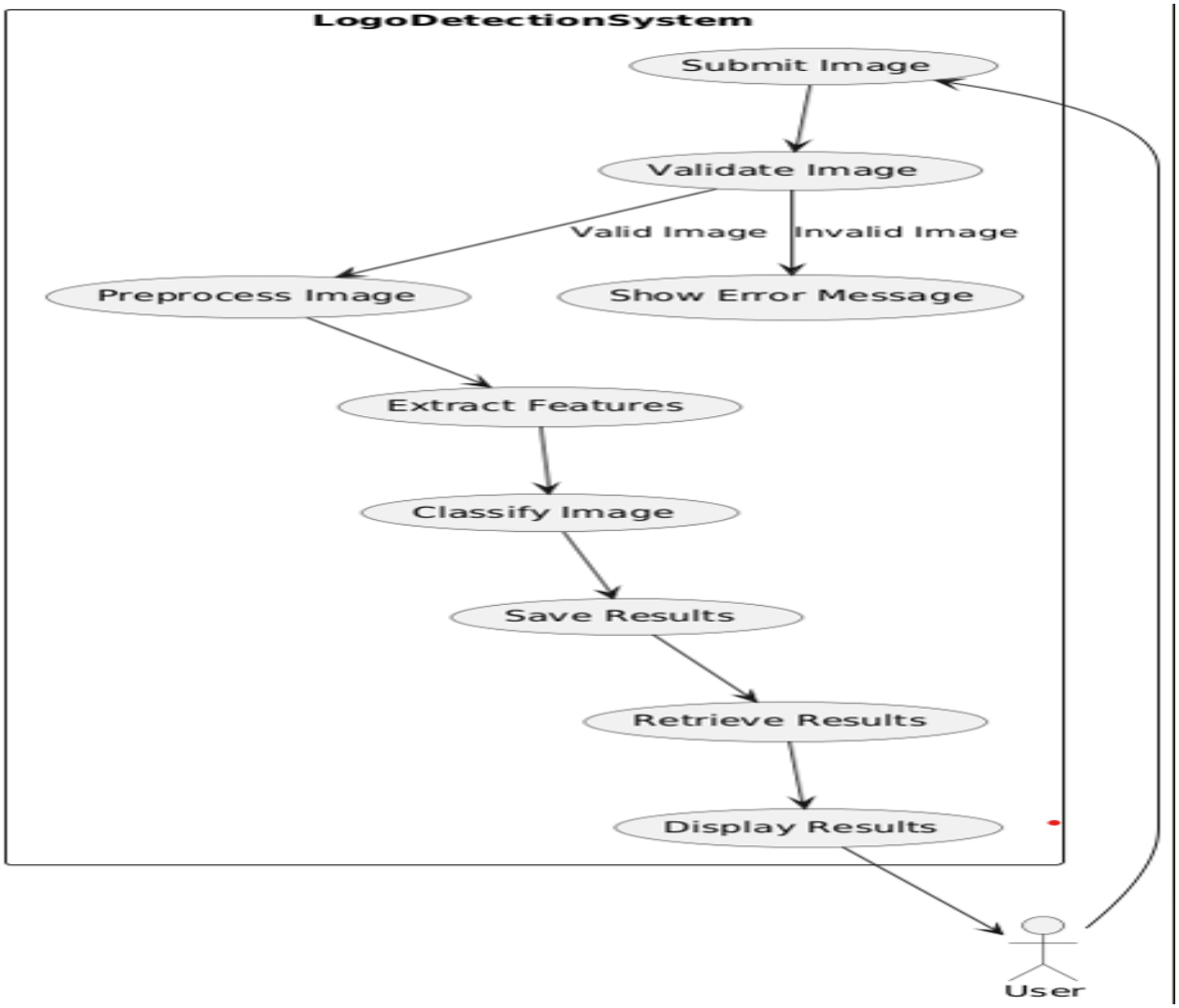



**Fig 4: Use case diagram for LOGO Detection**

The use case diagram for the logo detection project shows the interactions between the User and the Logo Detection System. The user uploads an image, which is then validated and processed by the system. The system performs preprocessing, feature extraction, and classification of the image, saving and retrieving the results to display back to the user. If the image is invalid, the system shows an error message, ensuring clear communication and robust functionality.

### 3.2.3ACTIVITY DIAGRAM

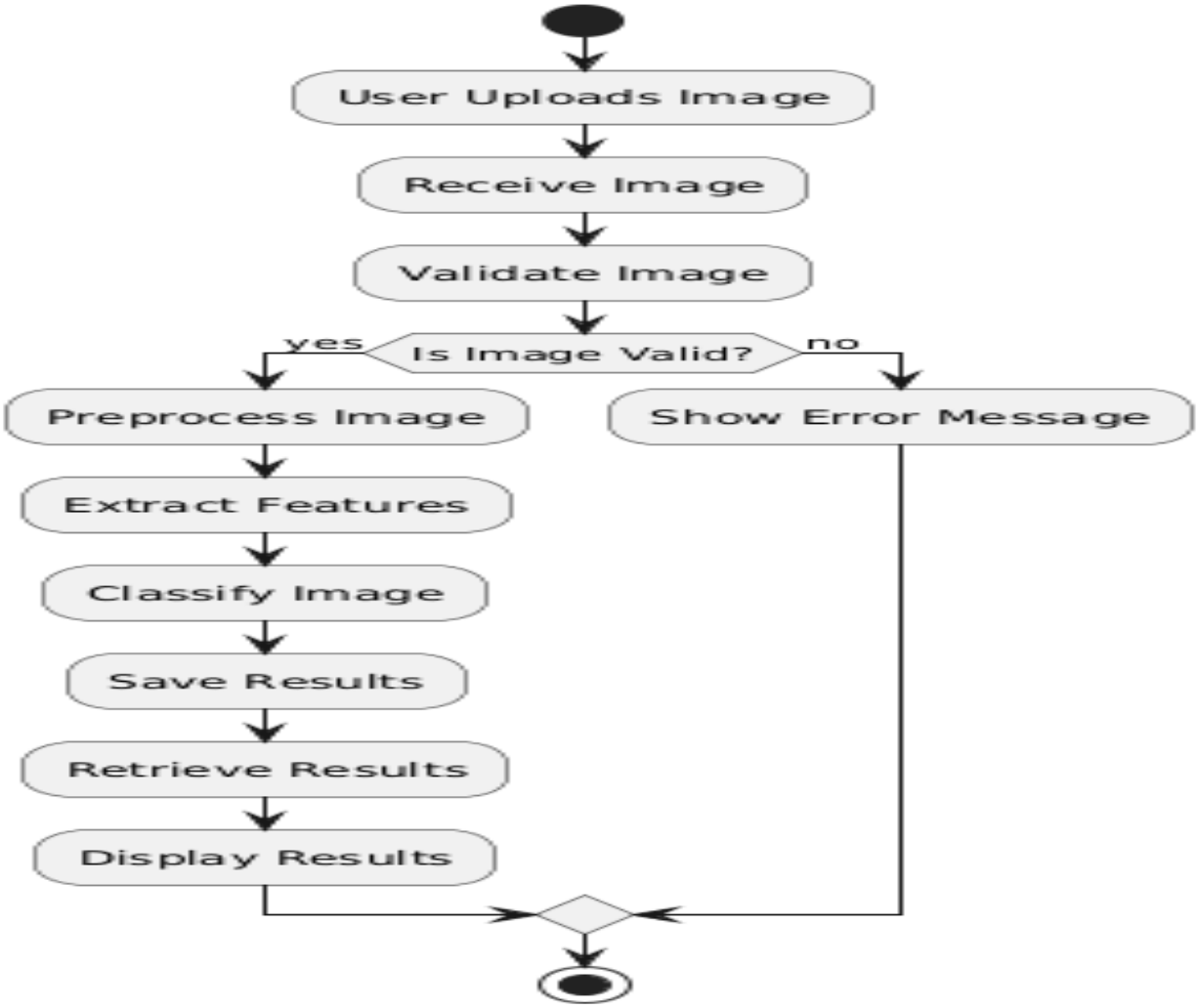



**Fig 5: Activity diagram for LOGO Detection**

The process begins when the user uploads an image. The system then receives and validates the uploaded image to ensure it's suitable for further processing. If the image is valid, it undergoes preprocessing steps such as resizing and normalization to prepare it for analysis. Next, features are extracted from the preprocessed image, which are then used to classify the image with a deep learning model. Once classified, the results are saved and retrieved for display. Finally, the system presents the results to the user. If the image is found to be invalid during validation, an error message is displayed to the user.

### 3.2.4 SEQUENCE DIAGRAM

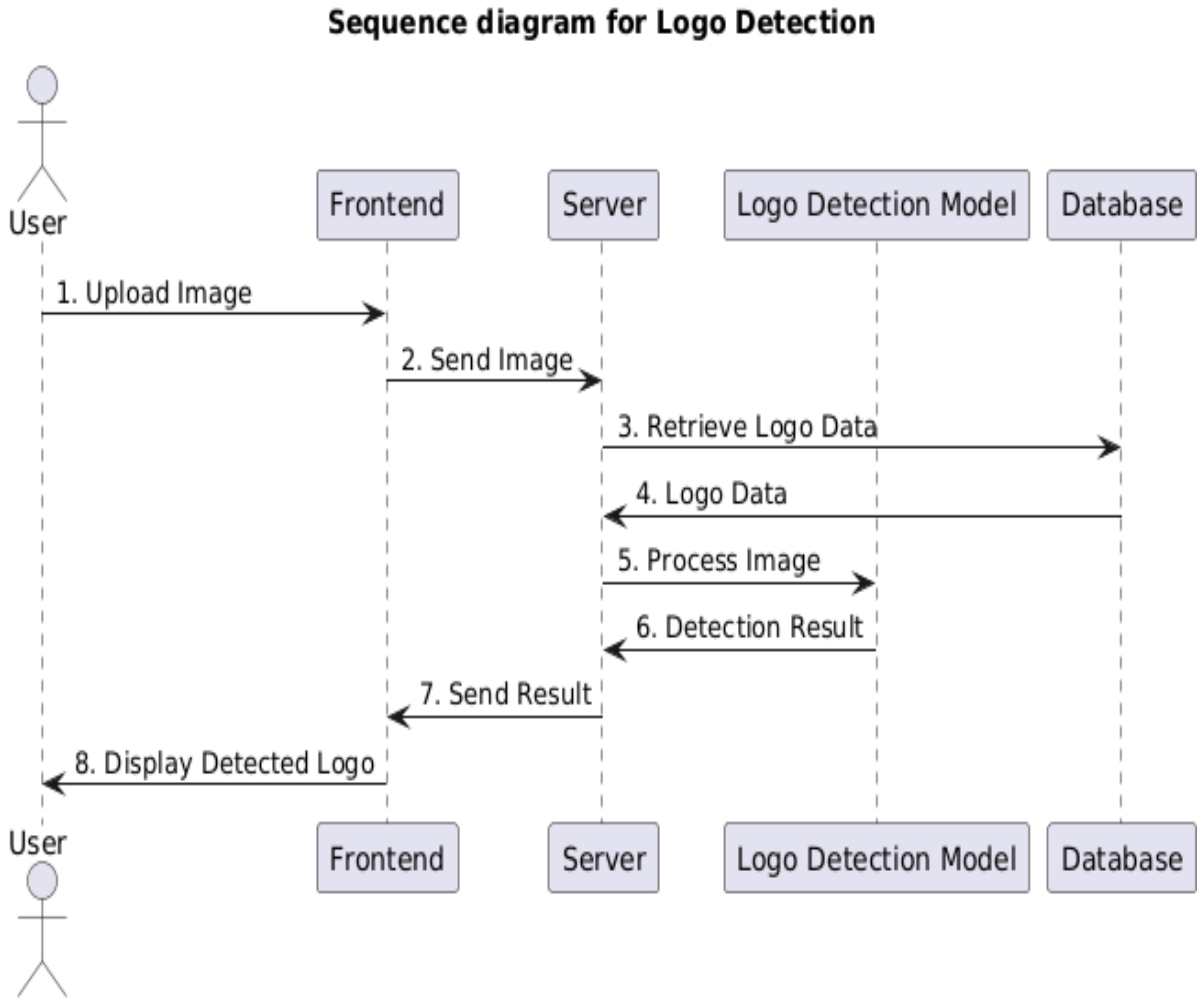



**Fig 6: Sequence diagram for LOGO Detection**

The process begins when the user uploads an image. The system then receives and validates the uploaded image to ensure it's suitable for further processing. If the image is valid, it undergoes preprocessing steps such as resizing and normalization to prepare it for analysis. Next, features are extracted from the preprocessed image, which are then used to classify the image with a deep learning model. Once classified, the results are saved and retrieved for display. Finally, the system presents the results to the user. If the image is found to be invalid during validation, an error message is displayed to the user.

# 4. IMPLEMENTATION

## 4.1 TECHNOLOGY DESCRIPTION & INSTALLATION STEPS

**DATA READING:** In the initial stages of our project, we begin by acquiring datasets for logo detection. This involves accessing platforms like Kaggle and Roboflow to source diverse and relevant data. Upon navigating these platforms, we locate and download the dataset files pertinent to our project after logging in and accessing the dataset pages on Kaggle. Similarly, on Roboflow, users can efficiently manage their data by uploading their own or exploring publicly available datasets upon signing into their accounts and obtain the datasets.

**DATA PREPROCESSING:** Data preprocessing is essential for enhancing the quality and effectiveness of image datasets in machine learning tasks. These steps encompass various strategies to ensure consistency, improve model generalization, and mitigate biases within the data. Common preprocessing steps include resizing images to a uniform size, normalizing pixel values to a standard range, and cropping images to focus on relevant regions of interest. Additionally, grayscale conversion, histogram equalization, and data balancing contribute to improving the quality and balance of the dataset. Feature extraction using pre-trained convolutional neural networks enables the extraction of relevant features, while data cleaning eliminates corrupted or low-quality images. Dimensionality reduction methods may also be applied to reduce computational complexity while preserving important features.

**DATA VISUALIZATION:** The specific values on the Y-axis will vary depending on how you choose to split the data. However, in general, training sets encompass a larger portion (often around 70%) to train the model. Validation sets (around 15%) are used to monitor the model's performance during training and adjust hyper parameters if needed. Finally, test sets (around 15%) provide a final assessment of the model's generalizability on unseen data. By visualizing the distribution in a histogram, you can easily check if there are any significant imbalances in the dataset splits. In the case of the histogram you sent, here's a more detailed breakdown:

- Training Set (70%): This is the largest portion of the data, indicating the model will be trained primarily on these images. Based on the training dataset we can decide that how well that your is trained.

•Validation Set (15%): This slice of the data is used to fine-tune the model during training and prevent overfitting.

•Testing Set (15%): This is the smallest set and represents unseen data. The model's performance on this set is a strong indicator of how well it will generalize to real-world scenarios. It is used to test the trained model. If the input is given to your trained model, it predicts the input accurately then it is consider as high accurate and decide the precision as High.

**Table 1 : Train-Valid-Test**

| S. No | Data Set | Data Points |
|---|---|---|
| 1. | Training | 11791 |
| 2. | Validation | 4000 |
| 3. | Testing | 4000 |
| 4. | Total Data Points | 19791 |

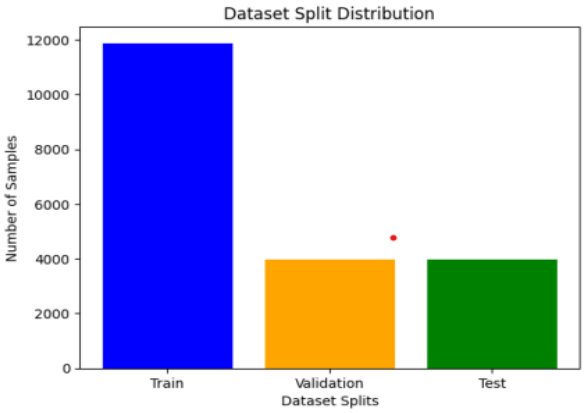



**Fig 7: Dataset Distribution**

**CNN-VGG19** is a convolutional neural network (CNN) architecture known for its effectiveness in image recognition tasks. It comprises 19 layers, including convolutional layers with small 3x3 filters and max-pooling layers. VGG19 has been pre-trained on large datasets like ImageNet, capturing generic features useful for various visual tasks. It can be fine-tuned for specific applications, such as logo detection, by retraining its weights on a target dataset. Its deep architecture allows it to learn intricate patterns in images, making it.

**YOLOv8**, or You Only Look Once version 8, is a state-of-theart object detection model. It offers real-time detection of objects in images and videos and is known for its speed and accuracy. YOLOv8 achieves this by dividing the input image into a grid and predicting bounding boxes and class probabilities for each grid cell simultaneously. With its improved architecture and training techniques, YOLOv8 enhances object detection performance across various scenarios, making it a popular choice for computer vision tasks.

**YOLOv9**, an evolution of the You Only Look Once (YOLO) object detection model, represents a significant advancement in real-time object detection. With enhanced architecture and training strategies, YOLOv9 achieves superior accuracy and efficiency in detecting objects within images and videos. YOLOv9 continues to push the boundaries of object detection capabilities, making it a leading choice for various computer vision applications.

**1. Mathematical Representation & process of Cnn-vgg19:**

The core of VGG19 lies in its convolutional layers. These layers apply filters of specific sizes (often 3x3 in VGG19) across the input image, producing feature maps. The calculation within a convolutional layer can be simplified to a dot product between the filter weights and the local image patch: Output [i, j] = Σ(Filter[m, n] * Input[i + m - 1, j + n - 1]).

**Activation Functions:**

VGG19 heavily utilizes the ReLU (Rectified Linear Unit) activation function after most convolutional layers. ReLU introduces non-linearity and helps the network learn complex patterns. The mathematical expression for ReLU is: ReLU(x) = max (0, x) where x is the input to the activation function.

**2. Mathematical Representation & process of YOLOv8:**

Network Operations:

1. Convolutions: $Yij = f(\Sigma(Wkl * X(i+k,j+l)) + b)$
    - Yij: Output value at position (i,j) in the feature map
    - Wkl: weight value at position (k,l) in the kernel
    - X(i+k,j+l): Input value at position (i+k,j+l)

    f: activation function (e.g., ReLU, LeakyReLU)

    - b: Bias term

2. Pooling: (maximum or average)

- Yij = Function(X(i,j),..., X(i+k,j+l))

3. Head Convolution:

- Head_conv(Fused_feature_maps)

4. Bounding Box Parameters:

- tx, ty = sigmoid(Head_conv(Fused_feature_maps)) (normalized center offsets)
- tw, th = exp(Head_conv(Fused_feature_maps)) (Width and height predictions)

5. Confidence Score:

- conf = sigmoid(Head_conv(Fused_feature_maps)) (object presence probability)

6. Class Probabilities:

- Pc = softmax (Head_conv(Fused_feature_maps)) (probability of each object class).

**3. Mathematical Representation & process of YOLOv9:**

Input: X (Image)

Output: Y (Feature Map)

Network Operations:

1. Backbone:

- Let X denote the input image.
- The backbone network processes X through a series of convolutional layers, denoted by Conv_backbone(X).
- The output feature maps from the backbone are denoted as F_backbone = Conv_backbone(X).

2. Feature Pyramid Network (FPN):

- The FPN takes the output feature maps F_backbone from the backbone and generates multi-scale feature maps F_FPN = {FPN_1(F_backbone), FPN_2(F_backbone), ...}.

3. Neck:

- The neck of YOLOv9 further processes the multi-scale feature maps F_FPN, denoted as F_neck = Neck(F_FPN).
- The output of the neck module is a refined set of features suitable for object detection.

4. Bounding Box Parameters:

- Bounding box parameters are predicted from the feature maps F_head.
- Let tx, ty, tw,and th denote the predicted bounding box parameters, calculated using functions applied to F_head.

5. Confidence Score:

- The confidence score represents the probability of object presence in each grid cell.
- Let conf = sigmoid(F_head) denote the confidence score.

6. Class Probabilities:
   - Class probabilities indicate the likelihood of each grid cell containing objects of different classes.
   - Let Pc = softmax (F_head)represent the class probabilities.

## Correlation Heat map:

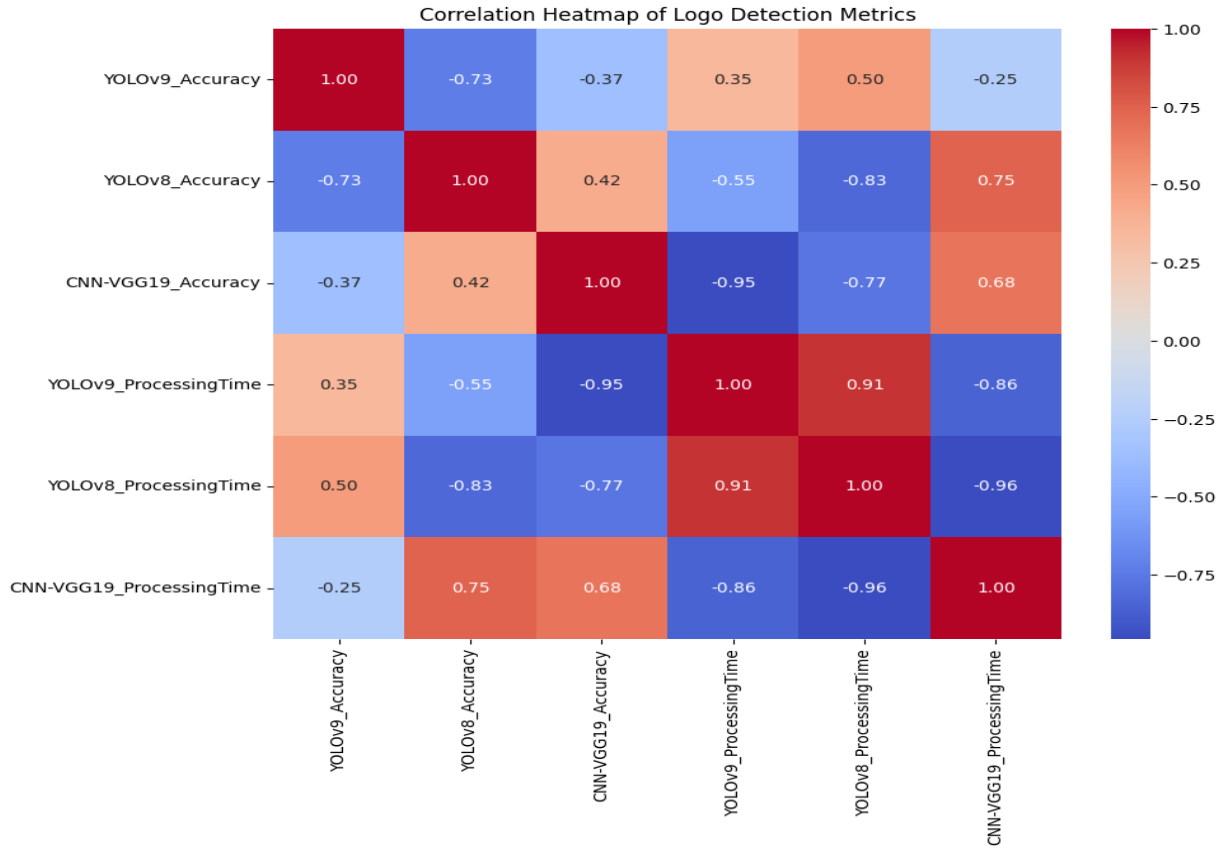



**Fig 8: Correlation Heat Map for YOLOv9, YOLOv8, CNN-VGG19**

# 5. RESULTS

## 5.1 RESULTS ANALYSIS & OUTPUTS:

**YOLOv8:** The boxes highlight detected logos on the original image, thanks to our YOLOv8 model. Text labels beside each box reveal the brand and its confidence score. A high score, like "0.89" for Puma, indicates strong confidence in the detection. This confirms our model's ability to pinpoint various logos within the scene. Our analysis of the YOLOv8 logo detection confusion matrix reveals both successes and areas for improvement. While the model can identify some logos accurately (True positives), it occasionally misses others. This suggests potential limitations in distinguishing logos from similar objects. Examining the confusion matrix alongside other metrics like mAP will provides a complete picture of the model's effectiveness and guide further development to enhance its logo detection capabilities.

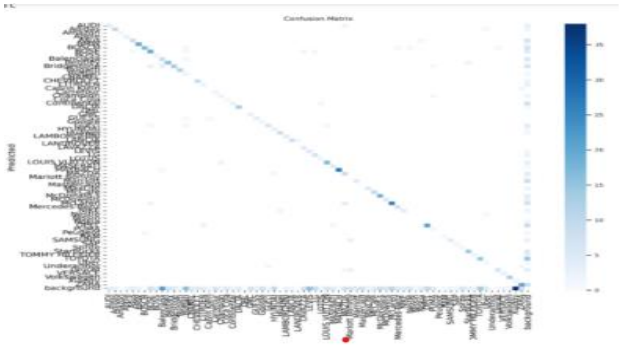


**Fig 9: Confusion matrix for YOLOv8**

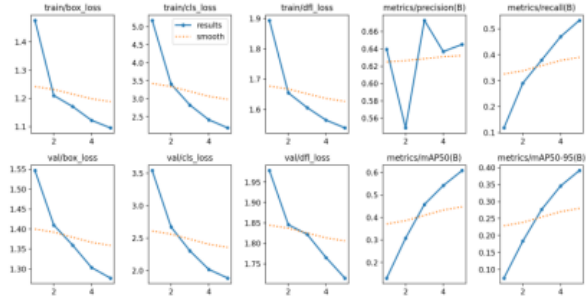



**Fig 10: Train & Validation loss for YOLOv8**


**Fig 11: Results of YOLOv8**

**YOLOv9:** The blue boxes highlight potential logos it identified, with labels revealing the brand and confidence score (like the high confidence "hp"). While YOLOv9 seems to be learning well from its training data, there's always room for improvement. The “0.91” is the higher confidence value which indicates the strong confidence in the detection of logos. Missing parts of logos or encountering similar designs might lead to missed detections or mistaking non-logo elements for logos. Analyzing this image helps us identify areas where we can refine YOLOv9's training process. By using higher-quality images and expanding the training data with a wider variety of logos, we can make YOLOv9 even better at detecting logos on products.

**Fig 12: Confusion matrix of YOLOv9**

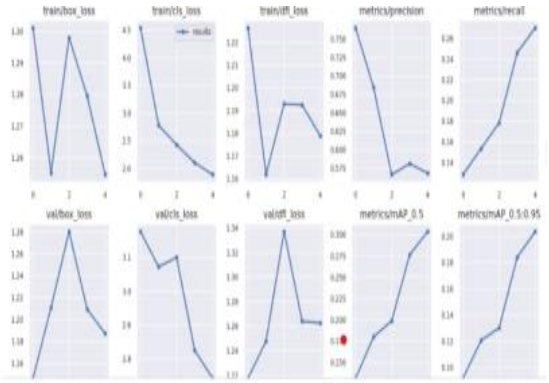


**Fig 13: Train & Validation loss of YOLOv9**


**Fig 14: Results of YOLOv9**

**CNN-VGG19:**In our experiment using CNN-VGG19 for logo detection, this graph visualizes the model's accuracy throughout training. The blue line represents how well the model performs on training examples, ideally increasing as it learns to recognize logos. The orange line shows accuracy on a separate validation set, indicating how well the model generalizes to unseen logos. We want both lines to climb as training progresses. While a small gap between the lines is normal, a large or growing gap, or a decrease in validation accuracy, could indicate overfitting. Overfitting happens when the model memorizes the training data too well and performs poorly on new logos. By monitoring this graph, we can ensure the model learns effectively and avoid overfitting. We might even stop training when the validation accuracy reaches a stable point (plateau) to prevent overfitting.


**Fig 15:Graph between Loss & Validation Loss in CNN**

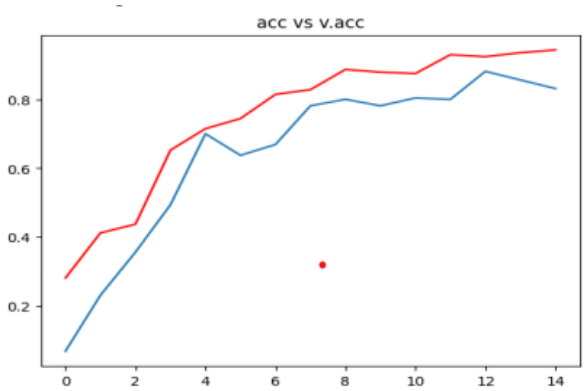



**Fig 16: Graph between Accuracy & Validation Accuracy in CNN**

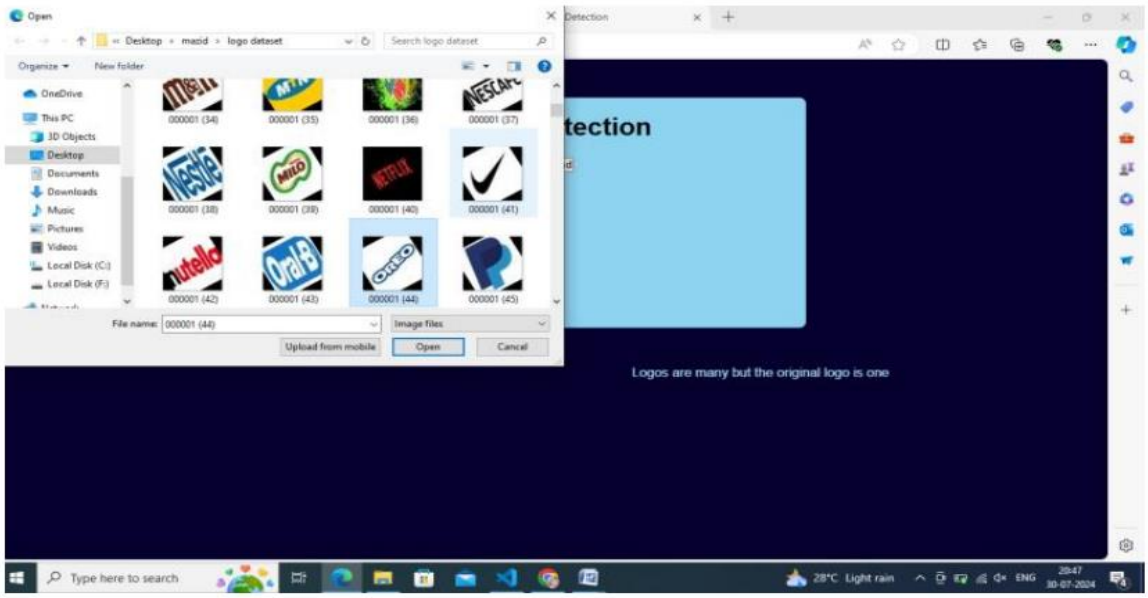



**Fig 15: Giving input to the CNN model (Ex: Oreo)**

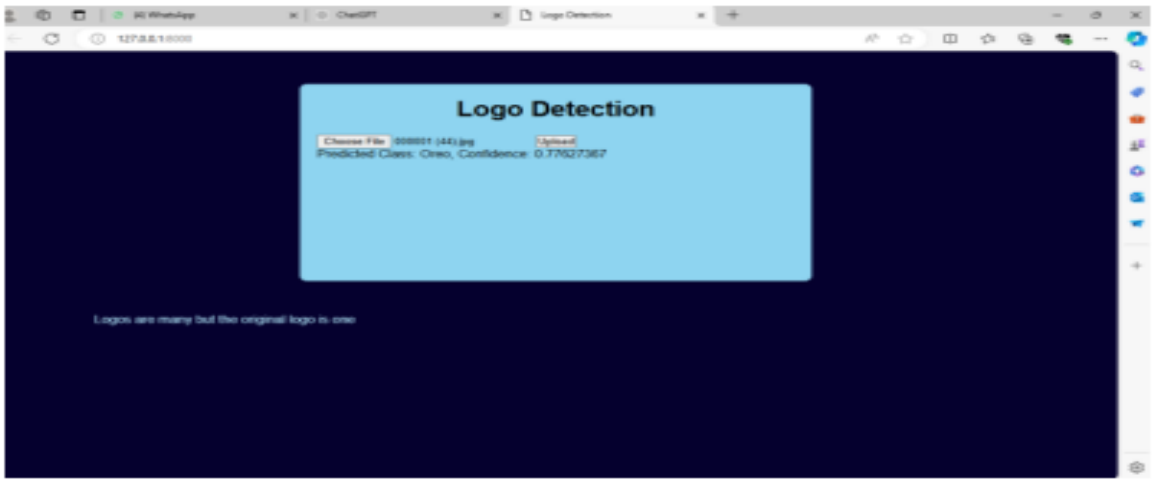



**Fig 16: Predicted Output as Oreo with Confidence of 77%**

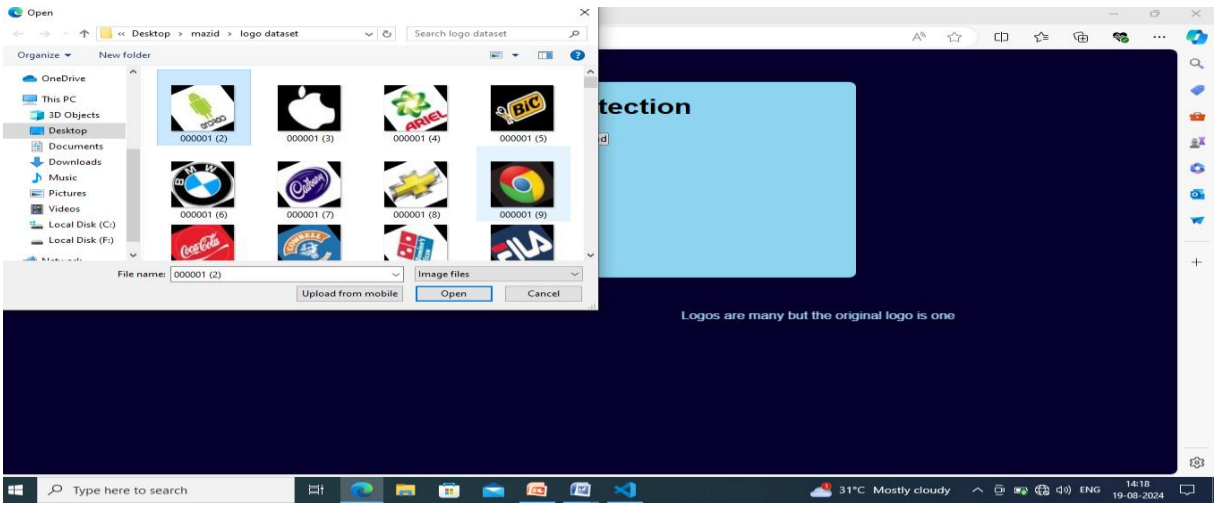



**Fig 15: Giving input to the CNN model (Ex: Android)**

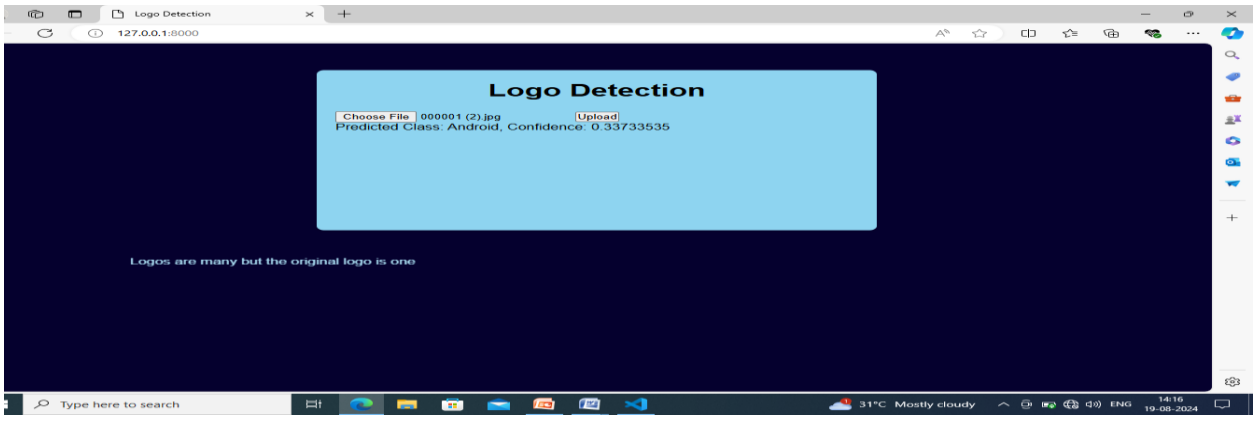



**Fig 16: Predicted Output as Android with Confidence of 33%**

# 6. CONCLUSION

In conclusion, our exploration of YOLOv8, YOLOv9, and CNN-VGG19 for logo detection highlighted a crucial trade-off between speed and accuracy. This trade-off is essential to consider when selecting the appropriate model for a specific application.

**YOLOv8 and YOLOv9** are part of the You Only Look Once (YOLO) family of models, which are known for their single-stage architecture. This architecture enables the model to detect objects in real-time, making them highly suitable for applications that require fast processing times. YOLO models perform detection and classification in a single step, leading to faster inference speeds. However, while their accuracy is generally good, it might not match the highest levels achieved by more complex models like CNN-VGG19.To maximize the performance of YOLOv8 and YOLOv9, it is crucial to focus on optimizing training data and evaluation metrics. Properly annotated and diverse training datasets can significantly enhance the model's ability to detect logos accurately. Additionally, selecting appropriate evaluation metrics helps in fine-tuning the model to meet the desired performance criteria. Finally our exploration of YOLOv8, YOLOv9, and CNN-VGG19 for logo detection revealed a critical trade-off between speed and accuracy. YOLOv8 and YOLOv9, with their single-stage architecture, prioritize real-time performance, and their effectiveness can be further enhanced through optimized training data and evaluation metrics. While **CNN-VGG19** boasted higher potential accuracy in our experiments, achieving an **accuracy of 94.20** for our logo detection model, its two-stage process came at the cost of speed. It excels in scenarios where precision is paramount. Ultimately, the best choice hinges on your project's requirements. For real-time needs, YOLO models reign supreme. If unmatched accuracy trumps speed, CNN-VGG19 might be the way to go. By understanding this trade-off, we can make informed decisions when selecting a logo detection method for various applications.

# 7. FUTURE SCOPE

- **Brand Monitoring and Analytics**: Use the detection model to analyze brand visibility across various media channels, providing valuable insights for marketing and advertising purposes.
- **Intellectual Property Protection**: Develop tools that use the logo detection model to monitor unauthorized use of logos, helping brands protect their intellectual property.
- **Cloud-based Detection**: Deploy the logo detection system on cloud platforms (like AWS, Azure, or GCP) to handle larger-scale image processing tasks, enabling the service to manage a high volume of images simultaneously.
- **API Development**: Develop a RESTful API that can be used by other applications or services to perform logo detection. This could open up possibilities for integrating your model with other systems.

# 8. REFERENCES


[1] Kandula, Ashok Reddy, R. Sathya, and S. Narayana. "Multivariate Analysis on Personalized Cancer Data using a Hybrid Classification Model using Voting Classifier." International Journal of Intelligent Systems and Applications in Engineering 11.1 (2023): 354-362.

[2] KANDULA, ASHOK REDDY, R. SATHYA, and S. NARAYANA. "COMPARATIVE ANALYSIS OF MACHINE LEARNING TECHNIQUES ON GENETIC MUTATION-BASED CANCER DIAGNOSIS DATA." Journal of Theoretical and Applied Information Technology 100.6 (2022).

[3] Vaijinath V. Bhosle and Vrushsen P. Pawar, "Automatic Logo Extraction and Detection for Document Verification using SIFT and SURF", International Journal of Engineering Research Technology, pp. 555-560, 2017.

[4] Shivakumar G, Ravikumar M, Shivaprasad B J, Guru D. S., "Extraction of Logo from Real Time Document Images Using Masking and Median Filter Approaches", 2022 3rd International Conference for Emerging Technology (INCET), pp.1-7, 2022.

[5] N. Nagajothi, Shabana Memon, G. Mohan, Parashuram Shankar Vadar, Anita Soni, S. Prince Sahaya, "Design and Development of a Novel Algorithm to Predict Fake Logo using Learning based Digital Image Analysis Methodology", 2024 5th International Conference on Mobile Computing and Sustainable Informatics (ICMCSI), pp.492-499, 2024.

[6] Kanokporn Pakhamwang, Narathip Sriherun, Noppason Waikawee, Promchat Jaijith, Suppakarn Chansareewittaya, Mahamah Sebakor, "The Logo Detection in an Illegitimate Web Page", 2023 7th International Conference on Information Technology (InCIT), pp.204-208, 2023.

[7] Shajib Ghosh, Patrick Craig, Jake Julia, Nitin Varshney, Hamed Dalir, Navid Asadizanjani, "DeepICLogo: A Novel Benchmark Dataset for Deep Learning-Based IC Logo Detection", 2023 IEEE Physical Assurance and Inspection of Electronics (PAINE), pp.1-8, 2023

**Program Outcomes (POs):**

Engineering Graduates will be able to:

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

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

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

**Conduct investigations of complex problems:** 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.

**Modern tool usage:** 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.

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

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

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

**Individual and team work:** Function effectively as an individual, and as a member or leaderin diverse teams, and in multidisciplinary settings.

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

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

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

**Program Specific Outcomes (PSOs):**

Engineering students will be able to

1. Process, interpret the real-world data to formulate the model for predicting and forecasting.
2. Apply machine learning techniques to design and develop automated systems to solve world problems.

**PROJECT PROFORMA**

| Classification of Project | Application | Product | Research | Review |
|---|---|---|---|---|
| | √ | | | |

## Note: Tick Appropriate category

| Project Outcomes | |
|---|---|
| Course Outcome (CO1) | Acquire technical competence in the specific domain during the training. |
| Course Outcome (CO2) | Identify the problem statement based on the requirements of the industry |
| Course Outcome (CO3) | Adapt project management skills on par with industrial standards. |
| Course Outcome (CO4) | Develop a system model to obtain a solution and generate are port. |

**Mapping Table**

| AD3510: INTERNSHIP/ INDUSTRIAL TRAINING/ PRACTICAL TRAINING | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Course outcomes | Program Outcomes and Program Specific Outcome | | | | | | | | | | | | | | |
| | PO1 | PO2 | PO3 | PO4 | PO5 | PO6 | PO7 | PO8 | PO9 | PO10 | PO11 | PO12 | | PSO1 | PSO2 |
| CO1 | 3 | 2 | 2 | 2 | 2 | | | 2 | 2 | 2 | 1 | 2 | | 2 | |
| CO2 | 3 | 3 | 2 | 2 | 1 | | | 2 | 2 | 2 | 1 | 2 | | 2 | 2 |
| CO3 | 1 | | 1 | | 1 | 1 | 1 | 2 | 2 | 2 | 3 | 2 | | 2 | |
| CO4 | 3 | 2 | 3 | 3 | 3 | 2 | 1 | 2 | 2 | 2 | 3 | 2 | | 2 | 2 |
| **INTERNSHIP/ INDUSTRIAL TRAINING/ PRACTICAL TRAINING** | 3 | 2 | 2 | 2 | 2 | 1 | 1 | 2 | 2 | 2 | 2 | 2 | | 2 | 1 |

**Note: Map each project outcomes with POs and PSOs with either 1 or 2 or 3based onlevel of mapping as follows:**

1-Slightly (Low) mapped 2-Moderately (Medium) mapped 3-Substantially (High) mapped