IMAGE BASED OBJECT COUNTING

A project report submitted to
MALLA REDDY UNIVERSITY
in partial fulfillment of the requirements for the award of degree of

BACHELOR OF TECHNOLGY
in
COMPUTER SCIENCE & ENGINEERING (AI & ML)

Submitted by

M. Satyam Babu	**:**	**2211CS020288**
M. Dileep Reddy	**:**	**2211CS020289**
M. Dheemanth	**:**	**2211CS020290**
M. Lahari	**:**	**2211CS020291**
M. Arshith	**:**	**2211CS020292**

Under the Guidance of
Dr.A.SIVARANJANI
Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING (AI & ML)

MALLA REDDY UNIVERSITY
(Telangana State Private Universities Act No.13 of 2020 and G.O.Ms.No.14, Higher Education (UE) Department)

2024

MALLA REDDY UNIVERSITY

(Telangana State Private Universities Act No.13 of 2020 and G.O.Ms.No.14, Higher Education (UE) Department)

COLLEGE CERTIFICATE

This is to certify that this is the Bonafide record of the Application Development entitled, **Image Based Object Counting** using machine learning submitted by M.Satyam Babu (2211CS020288), M.Dileep Reddy (2211CS020289), M.Dheemanth (2211CS020290), M.Lahari (2211CS020291), M.Arshith (2211CS020292) B.Tech II year II semester, Department of CSE (AI&ML) during the year 2023-2024. The results embodied in the report have not been submitted to any other university or institute for the award of any degree or diploma.

PROJECT GUIDE

HEAD OF THE DEPARTMENT
Dr.Thayyaba Khatoon
CSE(AI&ML)

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

We have been truly blessed to have a wonderful internal guide **Dr.A.Sivaranjani Asst.Professor , Department of AIML, Malla Reddy University** for guiding us to explore the ramification of our work and we express our sincere gratitude towards her for leading us through the completion of project.

We would like to say our sincere thanks to **Mrs.SivaRanjani, Inchrage & Assistant professor, Department of AIML, B.Tech, Malla Redd University** for providing seamless support and right suggestions are given in the development of the project.

We wish to express our sincere thanks to **Dr.Thayyaba Khatoon, HOD AIML Department, Malla Reddy University** for providing us with the conductive environment for carrying through our academic schedule and project with ease.

We wish to express our sincere thanks to **Dean sir, Vice Chancellor sir and The Management of Malla Reddy University** for providing excellent infrastructure and their visionary thoughts to prepare ourselves industry ready by focusing on new technologies,

Finally, we would like to thank our family members and friends for their moral support and encouragement to achieve goal.

M.Satyam Babu (2211cs020288),

M.Dileep Reddy (2211CS020289),

M.Dheemanth (2211CS020290),

M.Lahari (2211CS020291),

M.Arshith (2211CS020292)

ABSTRACT

Image-based object counting using machine learning (ML) is an innovative approach aimed at automatically detecting and quantifying objects within images or video streams. This project leverages advanced ML techniques to develop a robust and efficient system capable of accurately counting various objects in diverse environments. The primary goal is to create a scalable solution applicable to multiple domains, including traffic management, retail inventory monitoring, wildlife conservation, and crowd analysis. The project begins with the collection and annotation of a comprehensive dataset containing images with varied object instances. These images undergo a series of preprocessing steps such as resizing, normalization, and augmentation to enhance the quality and diversity of training samples. The core of the project involves selecting and implementing a suitable ML model architecture for object detection and counting. Popular architectures like Faster R-CNN, YOLO, SSD, EfficientDet, and Mask R-CNN are evaluated to determine the best fit for the specific application requirements. The project demonstrates the potential of ML-based object counting systems to revolutionize various industries by providing accurate, automated, and scalable solutions for object detection and quantification. The outcomes of this project highlight the effectiveness of combining ML techniques with comprehensive data preparation and rigorous model evaluation to address complex real-world challenges in object counting.

CONTENTS

CHAPTER 1

INTRODUCTION

1.1 PROBLEM DEFINITION:

In numerous applications, the ability to accurately count objects within images or video streams is critical. From managing traffic flow and inventory control in retail to monitoring wildlife populations and ensuring safety in crowded environments, the need for automated object counting systems is evident. Traditional methods, often manual and labor-intensive, are not only time-consuming but also prone to errors and inconsistencies. Machine learning (ML) offers a promising solution to these challenges by providing automated, scalable, and accurate object counting capabilities.

1.2 OBJECTIVE OF THE PROJECT:

1. Create an automated object counting system that uses machine learning to detect and count objects within images or video streams accurately.
2. Evaluate and select suitable machine learning models (e.g., Faster R-CNN, YOLO, SSD, EfficientDet, Mask R-CNN) that balance accuracy and computational efficiency.
3. Implement data preprocessing techniques (e.g., resizing, normalization, augmentation) to improve the quality and diversity of training data.
4. Optimize the chosen ML model through iterative training, validation, and hyperparameter tuning to achieve high accuracy and efficiency.
5. Provide a user-friendly interface for easy input of images or video streams, model configuration, and result visualization.
6. Design the system to be scalable and adaptable to various environments and conditions, ensuring reliable performance in real-world applications.

1.3 SCOPE OF THE PROJECT:

The scope of this project encompasses the development, implementation, and evaluation of an image-based object counting system utilizing machine learning (ML) techniques. The project aims to provide an automated, accurate, and scalable solution for counting objects in images and video streams across various application domains. By clearly defining the scope, this project aims to deliver a comprehensive and effective solution for image-based object counting using machine learning, addressing the needs of various industries and applications.

CHAPTER 2
ANALYSIS

2.1 PROJECT PLANNING AND RESEARCH :

Project planning and research are critical steps to ensure the successful development and deployment of an image-based object counting system using machine learning (ML). Develop an automated system for accurate object detection and counting in images and videos. Utilize advanced ML models to enhance accuracy and efficiency. Ensure scalability, adaptability, and user-friendliness of the system. Identify and engage with stakeholders to understand their needs and expectations. Define functional and non-functional requirements, including performance, scalability, and usability criteria. Conduct a thorough review of existing research, technologies, and methodologies in image-based object counting and ML. Assess multiple ML models (e.g., Faster R-CNN, YOLO, SSD, EfficientDet, Mask R-CNN) for suitability based on accuracy, speed, and resource requirements. Train selected models on the annotated dataset, using transfer learning and hyperparameter tuning to optimize performance. Develop a modular architecture, including components for data input, preprocessing, model inference, and result visualization. Integrate the trained ML model into the system, ensuring compatibility with hardware and software platforms. Design a user-friendly interface for easy interaction, configuration, and visualization of results. Deploy the system on selected platforms (e.g., cloud, on-premises) with considerations for scalability and maintenance. Conduct rigorous testing in various real-world scenarios to validate system performance and robustness. Prepare comprehensive documentation, including system design, implementation details, user guides, and maintenance procedures. Regularly update the system with new data, model improvements, and software enhancements to maintain high performance and adaptability.

2.2 SOFTWARE REQUIREMENT SPECIFICATION :

2.2.1 Software Requirements:

- Python 3.8 or later
- Database: MySQL
- PyCharm, VSCode, Jupyter Notebook - IDE’s
- Python Libraries: NumPy, Pandas, Matplotlib

2.2.2 Hardware Requirements:

- Memory (RAM): Sufficient RAM to handle the data and computations.
- Processor: A processor for performing machine learning tasks.
- Storage: Storage space for the application and data that manages.
- GPU: Required level of GPU for faster training and high performance inference.

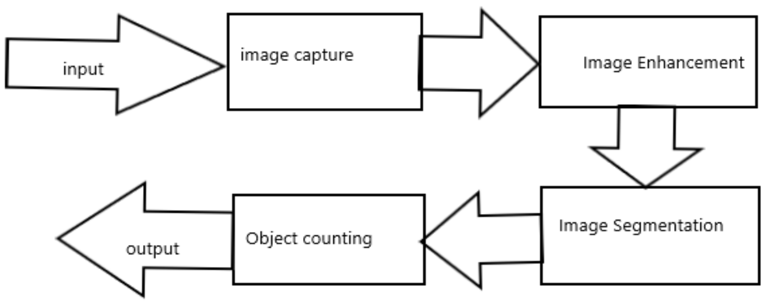
2.3 MODEL SELECTION AND ARCHITECTURE:

- CNNs (Convolutional Neural Networks) are fundamental architectures for image processing tasks, including object counting. They are particularly useful for feature extraction in images. Excellent at capturing spatial hierarchies in images. Can be fine-tuned for specific object counting tasks.

- R-CNNs (Region-based CNNs) and their derivatives, such as Fast R-CNN and Faster R-CNN, are popular choices for object detection. High accuracy for detecting and localizing objects. Faster R-CNN incorporates region proposal networks (RPN) for improved performance.

- SSD (Single Shot Multi-box Detector) is another real-time object detection model, offering a balance between speed and accuracy. Faster than R-CNN variants. Capable of detecting multiple objects in an image.

Architecture:

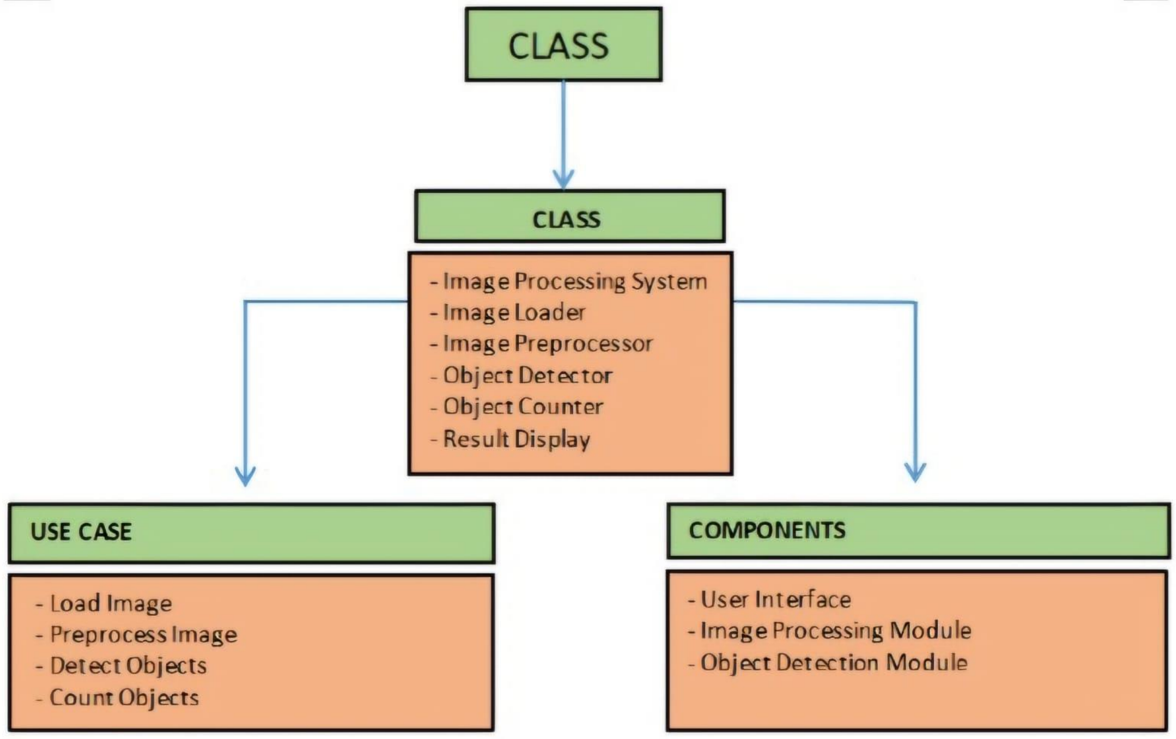


CHAPTER 3
DESIGN

3.1 INTRODUCTION:

Designing an image-based object counting system using machine learning (ML) involves a comprehensive approach to address the complexities of object detection, localization, and counting within images or video streams. This design introduction outlines the key aspects and considerations in developing such a system.

3.2 UML DIAGRAM:



3.3 DATA SET DESCRIPTIONS:

- **Object Classes**: The dataset should include multiple object classes that need to be counted. For example, if the project focuses on traffic monitoring, classes may include cars, buses, bicycles, pedestrians, etc. For retail inventory counting, classes could be different product categories.

- **Variability:** The dataset should capture variations in object appearances, such as different colors, sizes, orientations, occlusions, and backgrounds. This variability helps the ML model generalize better to unseen data.
- **Quantity:** The dataset needs to have a sufficient number of images with diverse scenes containing the object classes of interest. A larger dataset can help improve the model's accuracy and generalization ability.
- **Quality:** Images should be of high quality with clear visibility of objects. Low-resolution or blurry images can adversely affect model performance.
- **Data Split:** Divide the dataset into training, validation, and testing sets. The training set is used for model training, the validation set for hyperparameter tuning and model selection, and the testing set for final evaluation of the trained model.

3.4 DATA PREPROCESSING TECHNIQUES:

- **Data Collection:** Collect images from various sources, including real-world scenarios relevant to the application domain (e.g., traffic cameras, retail stores, wildlife cameras). Online datasets and repositories can also be utilized.
- **Data Augmentation:** Rotate images by a certain angle (e.g., 90 degrees) to simulate variations in object orientation. Flip images horizontally or vertically to account for different object perspectives. Zoom into or scale images to simulate variations in object size and distance. Introduce random noise to images to make the model more resilient to noisy data.
- **Data Normalization:** Normalize pixel values by subtracting the mean and dividing by the standard deviation of the dataset. Apply min-max scaling to scale pixel values to a specific range (e.g., 0 to 1 or -1 to 1).
- **Data Balancing:** Increase the number of samples in minority classes by duplicating or generating synthetic data. Decrease the number of samples in majority classes by randomly removing instances. Assign higher weights to minority classes during model training to give them more importance.

- **Noise Reduction and Image Enhancement:**

  Apply Gaussian blur to smooth out noise and improve object edges. Use edge detection algorithms (e.g., Canny edge detector) to enhance object boundaries. Adjust image histograms to improve contrast and visibility of objects.

- **Quality Control and Validation**: Perform visual inspection of augmented images to verify that transformations are applied correctly. Validate annotations and object counts after preprocessing to ensure alignment with ground truth data.

3.5 METHODS AND ALGORITHMS:

- **Image Decoding**:

  cv2.imdecode: Converts the uploaded image data from bytes to a NumPy array representing the image.

- **Image Conversion**:

  cv2.cvtColor: Converts the image from BGR to RGB format for accurate color representation in Matplotlib.

- **Object Detection**:

  cv.detect_common_objects: Detects common objects using a pre-trained model (e.g., YOLO or MobileNet-SSD). It returns bounding boxes, labels, and the count of detected objects.

  Algorithm: Typically, these pre-trained models are based on Convolutional Neural Networks (CNNs) that have been trained on large datasets to recognize various objects.

- **Drawing Bounding Boxes**:

  draw_bbox: Draws bounding boxes around detected objects, annotating the image with labels and confidence scores.

- **Interactive widgets**:

  FileUpload: Provides a widget for uploading image files.

  upload.observe: Sets up an event listener to trigger the on_upload_change function when a file is uploaded.

CHAPETR 4

DEPLOYMENT AND RESULTS

4.1 INTRODUCTION:

After extensive development, training, and testing phases, the deployment and results phase is a critical stage in the project "Image-Based Object Counting Using ML." This phase involves deploying the trained machine learning models into a production environment and analyzing the results obtained from object counting tasks. The deployment phase marks the transition of the developed ML models and the entire object counting system from a development environment to a production-ready setup. It involves setting up infrastructure, deploying models, integrating with user interfaces or APIs, and ensuring scalability and reliability for real-world usage.

4.2 SOURCE CODE:

```
import tkinter as tk

from tkinter import ttk, filedialog, messagebox

import cv2

import numpy as np

import cvlib as cv

from cvlib.object_detection import draw_bbox

from PIL import Image, ImageTk

# Dictionary to store registered users

users = {}

def register_user():

    username = register_username_entry.get()

    password = register_password_entry.get()

    if username in users:
        messagebox.showerror("Error", "Username already exists")
    else:
        users[username] = password
        messagebox.showinfo("Success", "User registered successfully")
def login_user():
    username = login_username_entry.get()
    password = login_password_entry.get()
    if username in users and users[username] == password:
        messagebox.showinfo("Success", "Logged in successfully")
        show_upload_page()
    else:
        messagebox.showerror("Error", "Invalid username or password")


def upload_image():
    file_path = filedialog.askopenfilename(filetypes=[("Image files", "*.jpg *.jpeg *.png")])
    if file_path:
        image_data = open(file_path, 'rb').read()
        count_objects(image_data)


def count_objects(image_data: bytes):
    img = cv2.imdecode(np.frombuffer(image_data, np.uint8), cv2.IMREAD_COLOR)
    if img is None:
        messagebox.showerror("Error", "Image not found or unable to load")
        return

    bbox, label, conf = cv.detect_common_objects(img)
    output = draw_bbox(img, bbox, label, conf)

    # Convert the image to RGB and display it using PIL
    output = cv2.cvtColor(output, cv2.COLOR_BGR2RGB)
    img_pil = Image.fromarray(output)
    img_tk = ImageTk.PhotoImage(image=img_pil)
    result_label.config(image=img_tk)
    result_label.image = img_tk

    messagebox.showinfo("Result", f"Number of objects detected: {len(label)}")

def show_register_page():
    register_frame.pack(fill="both", expand=True)
    login_frame.pack_forget()
    upload_frame.pack_forget()

def show_login_page():
    login_frame.pack(fill="both", expand=True)
    register_frame.pack_forget()
    upload_frame.pack_forget()
def show_upload_page():
    upload_frame.pack(fill="both", expand=True)

    login_frame.pack_forget()

    register_frame.pack_forget()


# Main application window

root = tk.Tk()

root.title("Object Detection App")


# Register frame

register_frame = ttk.Frame(root)

register_username_label = ttk.Label(register_frame, text="Username:")

register_username_label.pack(pady=5)

register_username_entry = ttk.Entry(register_frame)

register_username_entry.pack(pady=5)

register_password_label = ttk.Label(register_frame, text="Password:")

register_password_label.pack(pady=5)

register_password_entry = ttk.Entry(register_frame, show="*")

register_password_entry.pack(pady=5)

register_button = ttk.Button(register_frame, text="Register", command=register_user)

register_button.pack(pady=20)

go_to_login_button = ttk.Button(register_frame, text="Go to Login",
command=show_login_page)

go_to_login_button.pack(pady=5)
```

```
# Login frame

login_frame = ttk.Frame(root)

login_username_label = ttk.Label(login_frame, text="Username:")

login_username_label.pack(pady=5)

login_username_entry = ttk.Entry(login_frame)

login_username_entry.pack(pady=5)

login_password_label = ttk.Label(login_frame, text="Password:")

login_password_label.pack(pady=5)

login_password_entry = ttk.Entry(login_frame, show="*")

login_password_entry.pack(pady=5)

login_button = ttk.Button(login_frame, text="Login", command=login_user)

login_button.pack(pady=20)

go_to_register_button = ttk.Button(login_frame, text="Go to Register",
command=show_register_page)

go_to_register_button.pack(pady=5)


# Upload frame

upload_frame = ttk.Frame(root)

upload_button = ttk.Button(upload_frame, text="Upload Image", command=upload_image)

upload_button.pack(pady=20)

result_label = ttk.Label(upload_frame)

result_label.pack(pady=20)

show_login_page()


root.mainloop()
```

4.3 MODEL IMPLEMENTATION AND TRAINING:

- **Install Required Libraries:** Make sure you have the necessary libraries installed. You can install them using pip.
- **Define Object Detection Function**: Create a function that takes image data as input, performs object detection, and displays the results.
- **Create File Upload Widget:** Use IPython widgets to create a file upload widget for uploading images.
- **Define Image Upload Callback:** Define a callback function that is triggered when an image is uploaded. This function will call the object detection function with the uploaded image data.
- **Run the Code:** Run your code, upload an image using the file upload widget, and see the object detection results.

4.4 MODEL EVALUATION METRICS:

- **Precision:** Measures the accuracy of the positive predictions. It is the ratio of true positive (TP) detections to the total number of positive detections (TP + false positive (FP)). High precision indicates a low false positive rate.
- **Recall:** Measures the model's ability to detect all relevant objects. It is the ratio of true positive (TP) detections to the total number of actual positive objects (TP + false negative (FN)). High recall indicates a low false negative rate.
- **F1 Score:** The harmonic mean of precision and recall. It provides a balance between precision and recall, especially when you need to find an optimal balance between false positives and false negatives.
- **Mean Average Precision:** Calculates the average precision across different classes and different IoU thresholds. It provides a single metric to compare models.
- **Confusion Matrix:** Provides a complete picture of how well the model is performing by showing true positives, false positives, true negatives, and false negatives.

4.5 MODEL DEPLOYMENT:

- **Prepare Test Images:** Gather a set of test images that you want to use for evaluating the object detection model.
- **Run Inference on Test Images:** Use the deployed model to perform object detection on the test images. You can modify the count_objects function to accept image file paths as input and then perform inference on these images.
- **Evaluate Performance:** After running inference on all test images, evaluate the model's performance based on metrics such as:

  Accuracy: Percentage of correctly detected objects.
  Precision: Ratio of true positive detections to the total number of detected Objects.
  Recall: Ratio of true positive detections to the total number of ground truth Objects.
  F1 Score: Harmonic mean of precision and recall.
- **Error Analysis:**

  Analyze any false positives or false negatives encountered during testing. Look for patterns or common scenarios where the model may be struggling to accurately detect objects. Adjustments to the model architecture, training data, or hyperparameters may be necessary based on this analysis.
- **Iterative Improvement:**

  Based on the testing and validation results, make iterative improvements to the model. This may involve retraining the model with additional data, fine-tuning hyperparameters, or using different model architectures to achieve better performance.
- **Documentation:**

  Document the testing process, results, and any improvements made to the model. This documentation will be valuable for tracking progress and communicating the model's performance to stakeholders.

4.6 WEB GUI's DEVELOPMENT:

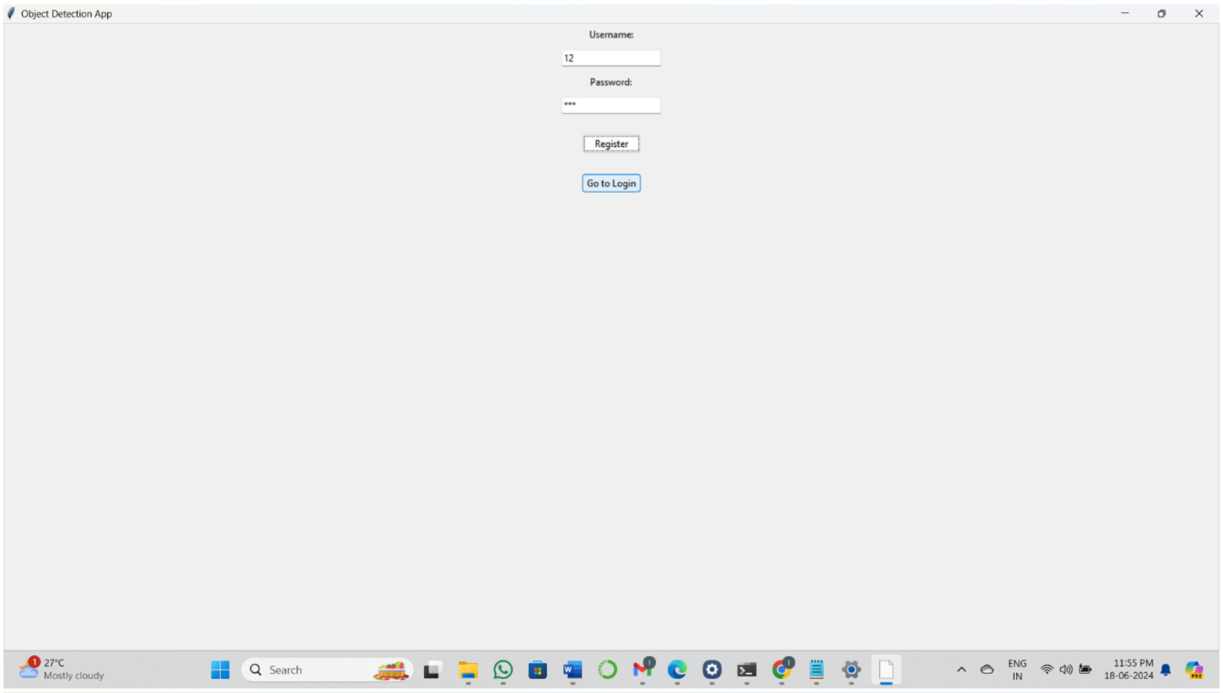
- **Frontend Development:**
  Frameworks/Libraries**:** Use modern frontend frameworks like React, Angular, or Vue.js for creating dynamic and responsive user interfaces. Design**:** Implement a clean, intuitive design using CSS frameworks like Bootstrap or Material-UI. Ensure the interface is accessible and visually appealing.
- **Backend Development:** Frameworks: Use robust backend frameworks like Django (Python), Flask (Python), or Node.js to handle server-side logic. APIs**:** Develop RESTful APIs to facilitate communication between the frontend and backend. These APIs will handle tasks such as image uploads, processing, and returning results.
- **Machine Learning Integration:**

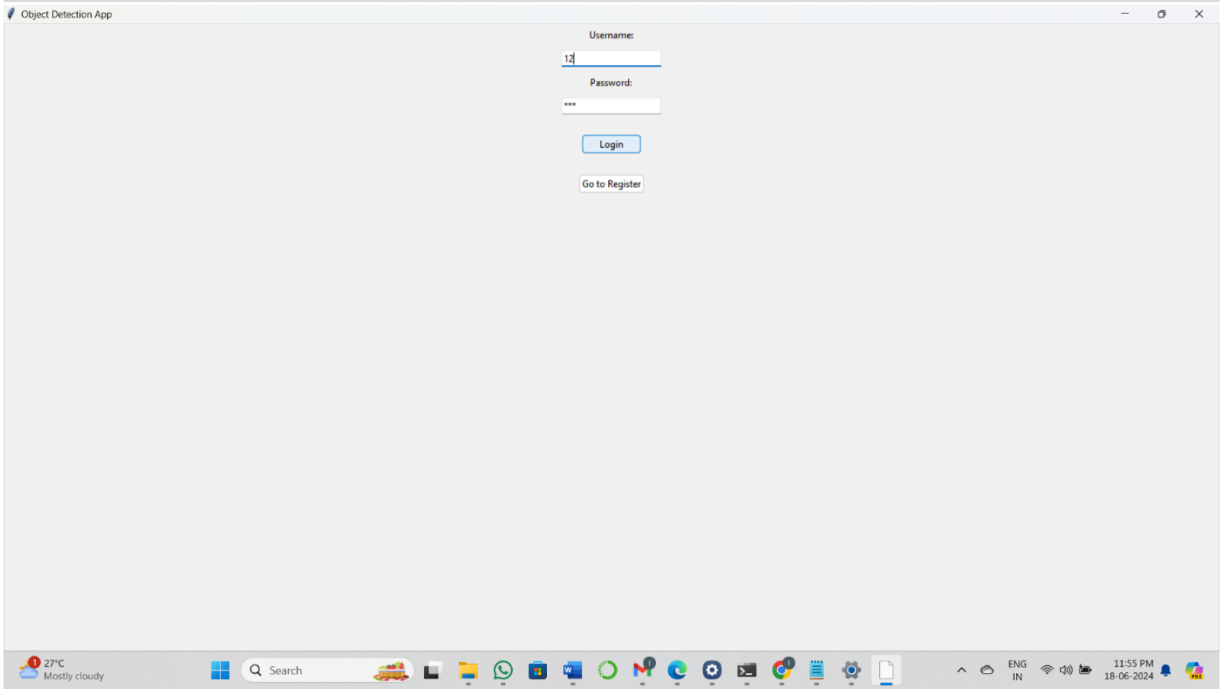
  Model Deployment: Use platforms like TensorFlow Serving, Flask-RESTful, or FastAPI to deploy the ML models. Ensure the models are optimized for quick response times. Inference: Ensure that the ML model can process images in real-time or near real-time to provide swift feedback to the user.

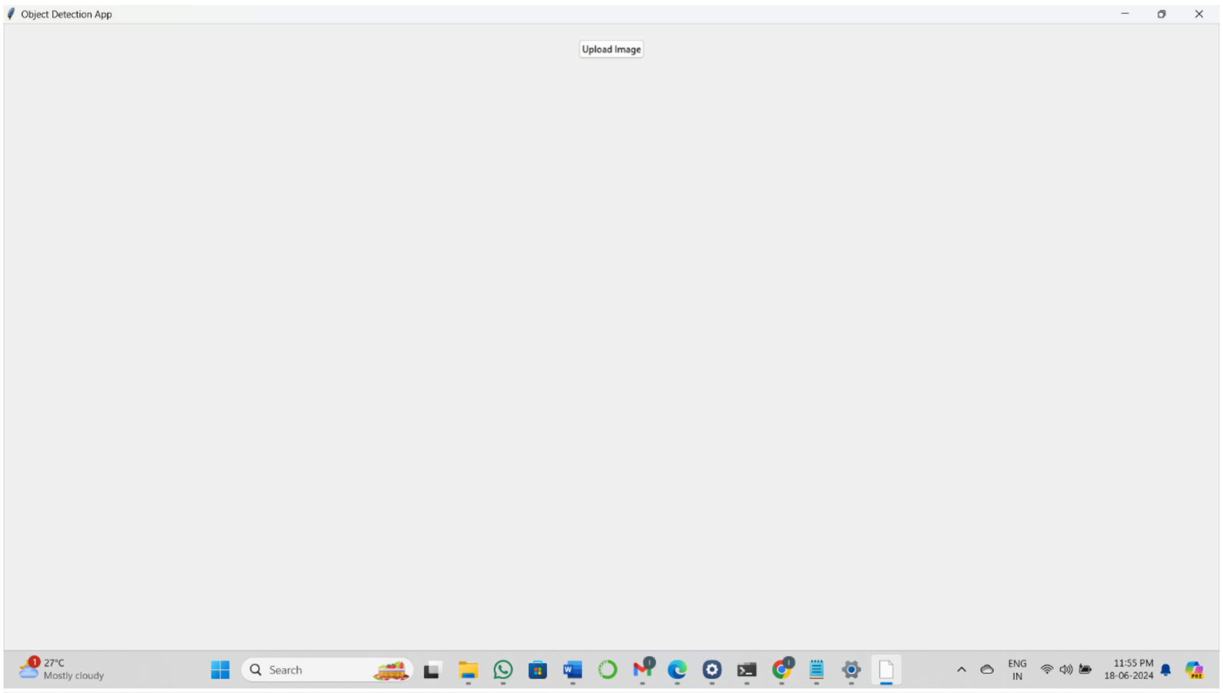
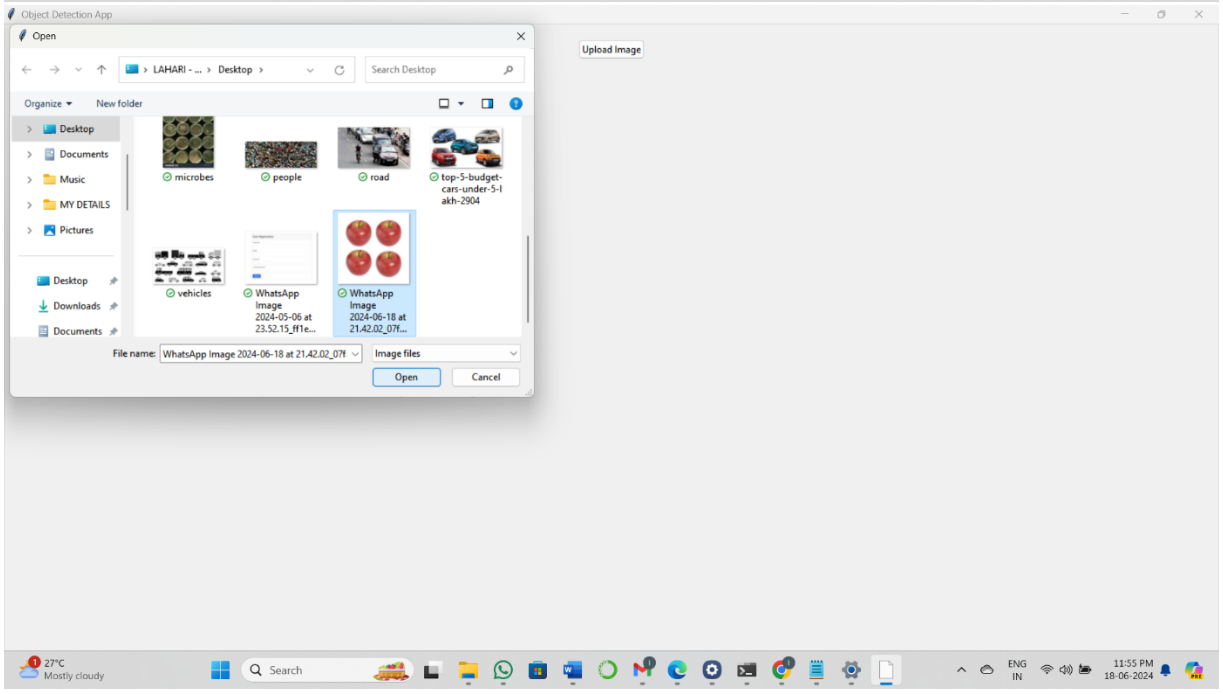
4.7 RESULTS:

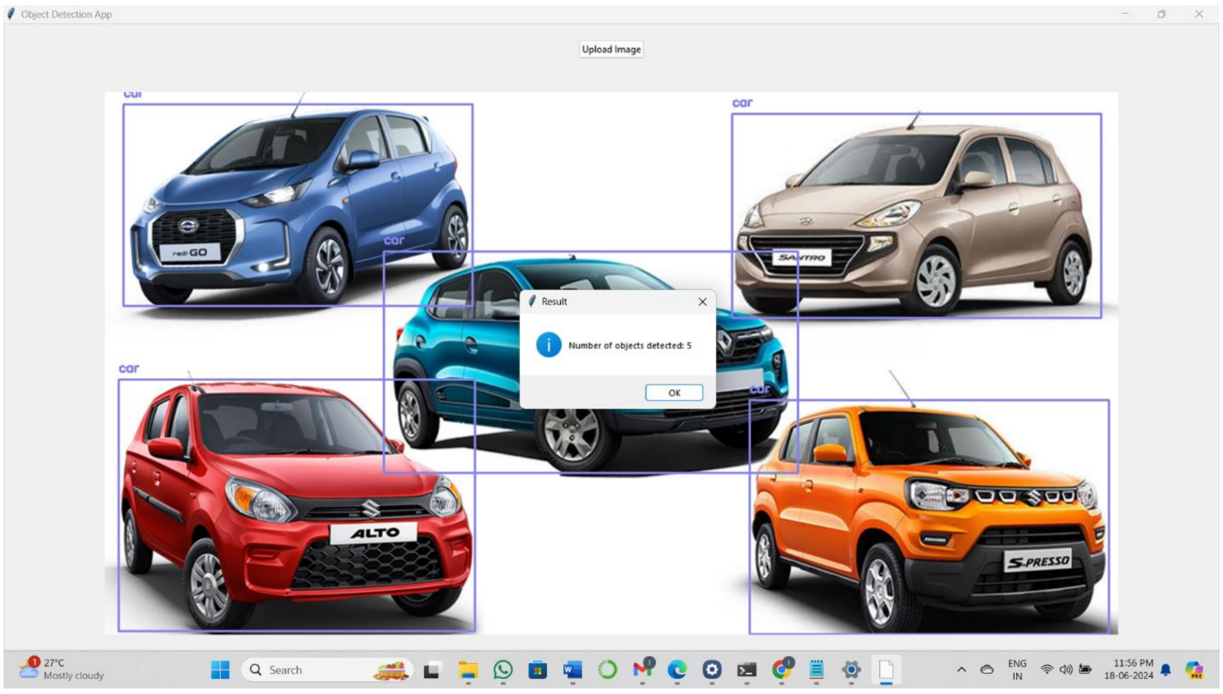
The project "Image-based object counting using machine learning" leverages advanced deep learning techniques to accurately count objects in various scenes. Recent developments have focused on improving object detection and tracking, In specific applications like vehicle counting on highways, deep convolutional networks have been utilized to detect and track multiple objects simultaneously. Challenges such as scale changes and illumination variations have been addressed using feature point detection algorithms like ORB, which provide faster and more accurate tracking compared to traditional methods like SIFT These advancements highlight the versatility and effectiveness of machine learning in object counting applications, from traffic management to manufacturing, by providing accurate, scalable, and efficient solutions.

OUTPUT SCREENS:






Object Detection App
Upload Image
27°C
Mostly cloudy
Search
ENG
IN
11:55 PM
18-06-2024

Object Detection App
Open
LAHARI - ... › Desktop ›
Search Desktop
Organize
New folder
Desktop
Documents
Music
MY DETAILS
Pictures
Desktop
Downloads
Documents
microbes
people
road
top-5-budget-cars-under-5-l akh-2904
vehicles
WhatsApp Image 2024-05-06 at 23.52.15_ff1e...
WhatsApp Image 2024-06-18 at 21.42.02_07f...
File name: WhatsApp Image 2024-06-18 at 21.42.02_07f
Image files
Open
Cancel
Upload Image
27°C
Mostly cloudy
Search
ENG
IN
11:55 PM
18-06-2024



Number of objects detected: 5

CHAPTER 5
CONCLUSION

5.1 PROJECT CONCLUSION:

The "Image-Based Object Counting Using ML" project represents a significant step forward in the domain of computer vision and machine learning, particularly in the context of object detection and counting within images or video streams. n conclusion, the project's achievements underscore the transformative power of machine learning in solving complex object counting tasks. The project not only delivers accurate and efficient object counting capabilities but also lays the groundwork for future advancements and applications in the field of computer vision and AI.

5.2 FUTURE SCOPE:

The project "Image-Based Object Counting Using ML" has significant potential for future enhancements, extensions, and applications. By exploring these avenues of future scope, the project can evolve into a comprehensive and versatile solution with broader impact and applicability across various domains and use cases. Continuous innovation, collaboration, and responsible development practices will be key to realizing the full potential of image-based object counting using machine learning.

REFERENCES

1. "AndroMoney (Expense Track) - Apps on Google Play", [online] Available: https://play.google.com/store/apps/details?id=com.kpmoney.android.

2."44% of World Population will Own Smartphones in 2017", [online] Available: https://www.strategyanalytics.com/strategyanalytics/blogs/smart-phones/2016/12/21/44-ofworld-populationwill-own-smartphones-in

3."Expense Manager - Apps on Google Play", [online] Available: https://play.google.com/store/apps/details?id = com.expensemanager.

4. Online Income and Expense Tracker S. Chandini1, T. Poojitha2, D. Ranjith3, V.J. Mohammed Akram4, M.S. Vani5, V. RajyalakshmiJ. Breckling, Ed., The Analysis of Directional Time Series: Applications to Wind Speed and Direction, ser. Lecture Notes in Statistics. Berlin, Germany: Springer, 1989, vol. 61.

5. Expense Tracker ATIYA KAZI1 , PRAPHULLA S. KHERADE2 , RAJ S. VILANKAR3 , PARAG M. SAWANT4 1 Professor, Department of Information Technology, Finolex Academy of Management and Technology, Ratnagiri, Maharashtra, India. 2, 3, 4 Department of Information Technology, Finolex Academy of Management and Technology, Ratnagiri, Maharashtra, India.