OBJECT IDENTIFICATION SYSTEM

A project report [thesis] submitted in partial fulfilment of The degree of

MASTER OF COMPUTER APPLICATION

by

ANKUSH KUMAR NAIK (ROLL NO- 15101FT22019)

Under the guidance of

DR. SABYASACHI PATTNAIK

PROFESSOR DEPARTMENT OF CS

and

co-guidance of

MR.NIKHILANANDA DHAL

ASST. DEPT. OF CS

P.G Department of Computer Science Fakir Mohan University Vyasa Vihar, Balasore-756019, Odisha

DECLARATION

*I, **ANKUSH KUMAR NAIK**, do hereby certify* that the thesis/project report entitled "**OBJECT IDENTIFICATION SYSTEM**", being submitted to FM University, Balasore, Odisha for the award of Master of Computer Applications, is an original piece of work done by me and the same has not been submitted elsewhere for any other academic degree or diploma to this university or any other university.

ANKUSH KUMAR NAIK

CERTIFICATE

This is to certify that the thesis/project report entitled "**OBJECT IDENTIFICATION SYSTEM**", submitted by **ANKUSH KUMAR NAIK**, for the award of the degree of MCA from FM University, Balasore, Odisha, India, is a bonafide record of work carried out by him under our guidance. Neither this thesis/project report nor any part of it has been submitted for any degree or academic award elsewhere.

(Signature of HOD)	**(Signature of Guide)**

ACKNOWLEDGEMENT

My specially thanks **to Dr. SABYASACHI PATTNAIK**, PG Department of Computer Science, who guided me with full support in completing my project through his constant encouragement and suggestions.

I extend my sincere thanks to Dr. Ashanta Ranjan Routray (H.O.D) PG Dept. of CS, Prof. Satchidananda Dehuri , PG Dept. of CS, Prof. Sabyasachi Patnaik, PG Dept. of CS, Dr. Minati Mishra (Assistant Professor), PG Dept. of CS, Ms. Monalisa Jena (Assistant Professor), PG Dept. of CS, Dr. Manaswini Pradhan PG Dept. of CS, Mr. Nikhilananda Dhal, PG Dept. of CS, for their continuous inspiration and intellectual supporting this project work.

Special thanks to my family members especially to my parents and friends whose uninterrupted love, inspiration and blessings helped me to complete my Project Work.

With Best Regards.
ANKUSH KUMAR NAIK

ABSTRACT

Object Identification System

The accurate and real-time identification of objects within images, videos, and live webcam feeds remains a challenging task in computer vision. Traditional methods often struggle with speed and accuracy, hindering applications such as surveillance, autonomous driving, and augmented reality.

The Object Identification system aims to provide a convenient way to identify objects in a given image or video or webcam using machine learning. Object identification and detection play a crucial role in various domains, including surveillance, autonomous vehicles, and image/video analysis.

The project will involve implementing the YOLO algorithm using OpenCV, a popular computer vision library. YOLO's single forward pass approach enables faster and more accurate object detection compared to traditional methods. The system will undergo training on a dataset containing various object classes to optimize performance. Real-time object detection will be achieved by leveraging the parallel processing capabilities of modern GPUs.

Hardware requirements include a computer with a dedicated GPU for optimal performance, webcam for testing real-time detection, and sufficient memory for handling large datasets. Software requirements include OpenCV, Python programming language, and libraries for machine learning and deep learning.

This project will contribute to the field of computer vision by providing a fast and accurate solution for object identification in various media formats. The developed system has the potential to enhance numerous applications, including surveillance systems, autonomous vehicles, and augmented reality, thereby advancing the capabilities of AI-powered technologies in real-world scenarios.

CONTENTS

DECLARATION……………………………………………………………………………..02

CERTIFICATE………………………………………………………………………………03

ACKNOWLEDGEMENT…………………………………………………………………...04

ABSTRACT………………………………………………………………………………..05

CHAPTER-1:INTRODUCTION……………………………………………………….10-11

1.1 History……………………………………………………………………………………10

1.2 Object Recognition……………………………………………………………………….10

1.3 Object Segmentation……………………………………………………………………...11

CHAPTER-2: PROJECT ANALYSIS AND BACKGROUND……………………….12-15

2.1 Introduction to Object Detection…………………………………………………………12

2.2 Convolutional Implementation of the sliding window…………………………………...12

2.2.1 Transformation of Layers into Convolutional Process……………………..……13

2.2.2 Convolutional Version of the Sliding Window………………………………..13

2.3YOLO Algorithm…………………………………………………………..………..15

2.3.1 Introduction to YOLO…………………………………………………………15

2.3.2 Grid Division and Localization Algorithm……………………………………15

2.3.3 Model Output and Target Variable…………………………………..............15

CHAPTER-3: YOLO OBJECT DETECTION ……………………………..………...17-23

3.1 What is Object Detection? ……………………………………………………………..17

3.2 How does product detection work?..17

3.3 What is the purpose of YOLO?...18

3.4 Non-maximum Suppression……………………………………………………………21

3.5 Using YOLO using OpenCV…………………………………………………………..23

CHAPTER 4: SYSTEM REQUIREMENTS……………………………………………..24-36

4.1 Introduction……………………………………………………………………………..24

4.1.1 PYTHON………………………………………………..……………………….24

4.1.2 PyCharm Community edition 2024.1……………………………………………..24

4.2 Complete System Requirements and Description…………………………………………...24

4.2.1 Overview……………………………………………………………………….24

4.2.2. Steps to be followed………………………………………………….............25

I.OpenCV………………………………………………………………25

II. NumPy………………………………………………………………27

III. TensorFlow…………………………………………………………28

IV. Pandas………………………………………………………………28

V. SciPy…………………………………………………………...30

VI. Matplotlib…………………………………………………………..30

VII. Pillow………………………………………………………...........30

VIII. Argparse……………………………………………………………..30

IX. Tkinter……………………………………………………………..31

4.3 Darknet……………………………………………………………………………31

i. How It Works…………………………………………………….................31

ii. Discovery Employing A Pre-Trained Model………………………………32

iii. Coco.names ……………………………………………………………34

CHAPTER 5: SYSTEM DESIGN……………………………………………….....37-39

5.1 Use Case Diagram…………………………………………………………...37

5.2. Data Flow Diagram……………………………………………………………….38

5.3. Sequence Diagram……………………………………………………………..39

CHAPTER 6: CODING……………………………………………..………..40-45

6.1 Coding……………………………………………………………………………..40

6.2. Description of The Code……………………………………………………….45

CHAPTER 7: OUTPUT SCREENS……………………………………………46-52

7.1 GUI Window Output………………………………………………...................46

7.2 Object Detection in Images Output……………………………………………46

7.3 Object Detection in video Output………………………………………………49

7.4 Object Detection in Webcam Output………………………………...................51

CHAPTER 8: SCOPES OF IMPROVEMENT AND CONCLUSION………………53-53

8.1 Scopes of Improvement……………………………………………………...53

8.2 Conclusion……………………………………………………………………..53

BIBLIOGRAPHY…………………………………………………………..………54

LIST OF FIGURES

Figure 2.1……12

Figure 2.2……13

Figure 2.3……13

Figure 2.4……14

Figure 2.5……14

Figure 2.6……14

Figure 2.7……15

Figure 2.8……16

Figure 3.1……17

Figure 3.2……19

Figure 3.3……20

Figure 3.4……20

Figure 3.5……21

Figure 3.6……21

Figure 3.7……22

Figure 4.1……32

Figure 4.2……33

Figure 5.1……37

Figure 5.2……38

Figure 5.3……39

Figure 7.1……46

Figure 7.2……46

Figure 7.3……47

Figure 7.4……47

Figure 7.5……48

Figure 7.6……48

Figure 7.7……49

Figure 7.8…………………………………………………………………………………49
Figure 7.9…………………………………………………………………………………50
Figure 7.10………………………………………………………………………………..50
Figure 7.11………………………………………………………………………………..51
Figure 7.12………………………………………………………………………………..51
Figure 7.13………………………………………………………………………………..52

CHAPTER-1:INTRODUCTION

1.1 History

Until a few years ago, the realm of graphics development, encompassing both software and hardware, was primarily the domain of skilled developers. Many programmers in numerous companies dedicated their efforts to this field, often resorting to operating system interventions to address graphical issues. However, progress in addressing conventional tasks like facial recognition, license plate identification, and remote image analysis and editing has been slow. Moreover, existing technological solutions are often prohibitively expensive. Consequently, there has been a burgeoning interest in automating the development of software tools tailored for solving complex intellectual challenges, with a focus on international initiatives. Effective image processing tools must enable the analysis and recognition of content in previously unexplored domains, empowering novice programmers to adopt best practices. Similarly, Windows toolkits facilitate the creation of interfaces adept at resolving diverse application challenges.

1.2 Object Recognition

Object recognition describes a group of comp vision-related jobs, such as identifying objects in digital images. Image classification involves activities such as predicting the category of objects in the image. Object localization means determining the location of one or more objects in an image and drawing a box around them. Object detection combines these two tasks and finds and classifies one or more objects in an image. When users or practitioners use the term “object recognition,” they usually mean “object detection.” It can be difficult for first time users to differentiate between different computer vision jobs.

Image classification involves determining the category or type of objects present in an image. This process requires analyzing the input, typically an image depicting objects, and generating an output in the form of category labels, which may consist of one or more numerical identifiers.

Object positioning entails identifying objects within an image and delineating their locations using bounding boxes. The input for this task is an image containing one or more objects, and the output comprises one or more bounding boxes, each specifying the position of an object through its width and height.

Object Detection encompasses both identifying objects within an image using bounding boxes and determining their respective categories or types. Similar to object positioning, the input is an image containing objects, and the output includes bounding boxes defining object locations along with labels assigned to each box, indicating the category of the detected object.

1.3 Object Segmentation

An extension of computer vision beyond segmentation is object segmentation, often termed "object instance segmentation" or "semantic segmentation." Here, the aim is to identify individual instances of recognized objects by precisely delineating specific pixels associated with each object, rather than using coarse bounding boxes. This nuanced analysis underscores the significance of object recognition in tackling complex computer vision tasks.

Consider the straightforward task of image classification, contrasted with the more intricate distinctions involved in object spatial perception. The terminology used for these tasks, including object recognition, can sometimes blur the lines between them.

Humans possess an innate ability to detect and identify objects in images swiftly and accurately, even performing complex tasks like identifying multiple objects and diagnosing issues with minimal prior experience. Thanks to advancements such as increased data availability, faster GPUs, and improved algorithms, training computers to achieve high-accuracy object identification and classification is now achievable. Essential concepts in this realm include product discovery, regional product identification, transactional loss considerations for both product and regional searches, and the concept of "You Only Look Once" (YOLO) for rapid object detection.

Object identification entails assigning labels to images, while object localization involves outlining bounding boxes around detected objects. Object discovery, which is more challenging, combines these tasks by delineating bounding boxes encompassing all objects of interest within an image and assigning their corresponding labels. Collectively, these tasks fall under the umbrella of object recognition.

Region-based Convolutional Neural Networks (R-CNN) represent a key technology for accurately identifying objects within regions, leveraging cognitive functions to enhance model processing. Additionally, the "You Only Look Once" (YOLO) approach represents a distinct family of technology products tailored for speed and real-time application.

CHAPTER-2: PROJECT ANALYSIS AND BACKGROUND

2.1 Introduction to Object Detection

The objective of object detection is to identify all occurrences of objects belonging to known categories, such as people, cars, or faces, within an image. While paintings typically feature only a few instances of objects, they can appear across various positions and scales, necessitating comprehensive exploration. Each displayed image conveys specific information, ranging from the object's position, location, and scale to its spatial extent defined by bounding boxes. In certain scenarios, pose information is intricate, incorporating parameters for nonlinear transformations. For instance, in face detection with masks, computations involve determining the positions of the eyes, nose, and mouth, alongside establishing the bounding box encompassing the face. An illustration in Figure 2.1 demonstrates the process of searching for a bicycle in an image, highlighting various potential locations. Object detection systems generate models for object classes based on training data. While a single sample may suffice for rigid objects within an image, multiple training models are generally necessary to encompass diverse object classes.

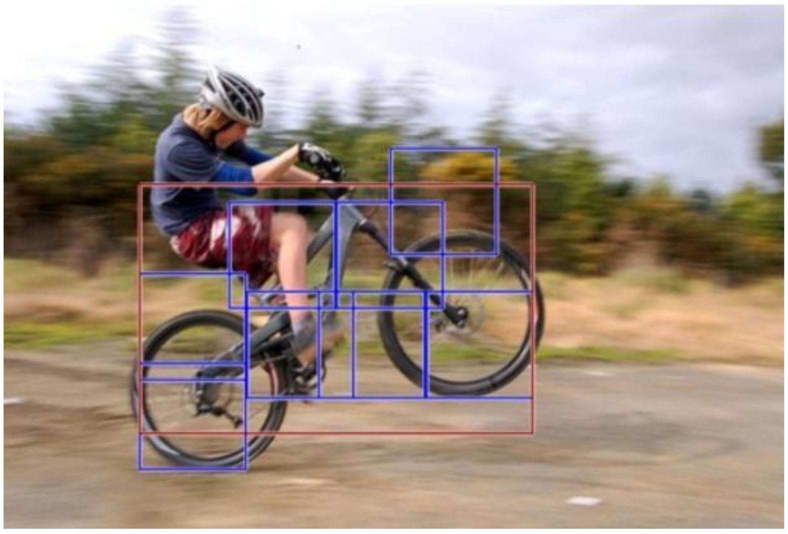


Figure 2.1

2.2 Convolutional Implementation of the sliding window

Convolutional implementation of the sliding window Before discussing the implementation ofthe sliding window using convention, let's first define how to transform the layers in a convolutional process. Figure 2.2 shows a simple network with two connected layers, one image .

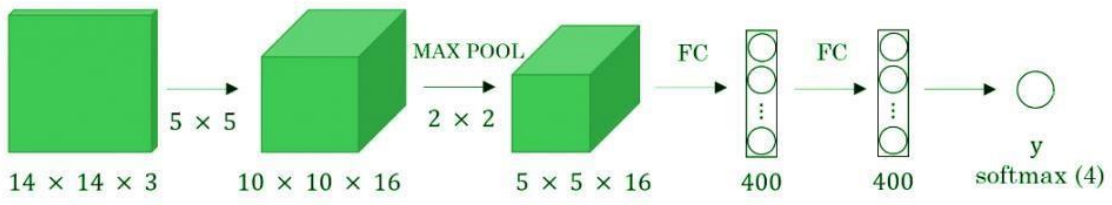


Figure 2.2

2.2.1 Transformation of Layers into Convolutional Process

With one_dimensional convolutional layers, all layers will be converted to convolutional layers. The width and height of this layer are equal to 1, and the filter is equal to the shape of the connected layer. Figure 2.3 shows an example.

2.2.2 Convolutional Version of the Sliding Window

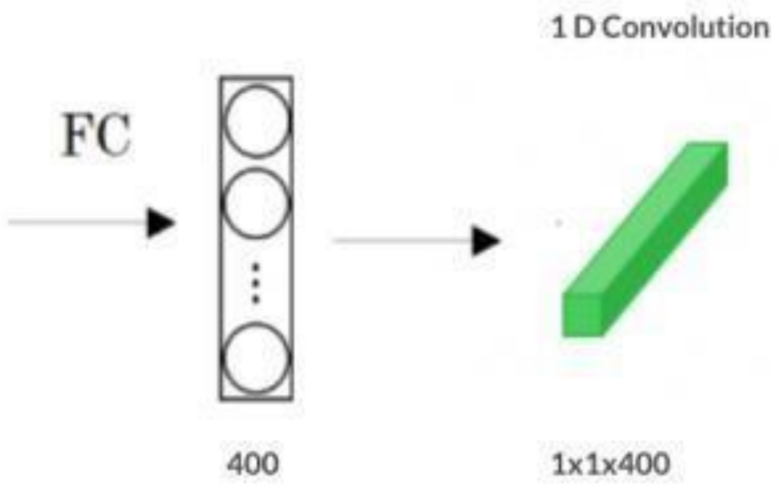


Figure 2.3

We can use the idea of changing all layers to layers in the model and replace the layers with a one-dimensional layer. The amount of filter in a layer is same as the image of a connected layer.

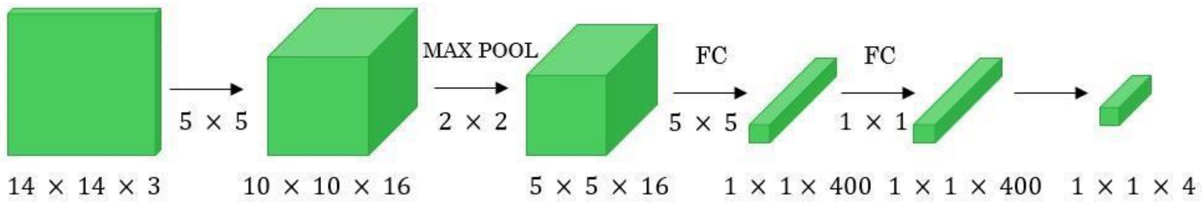


Figure 2.4

Expanding upon the aforementioned process, let's integrate the convolutional version of the sliding window. Initially, let's delve into the Convolutional Neural Network (ConvNet), excluding the connected layer, which we'll analyze for the forthcoming demonstration.

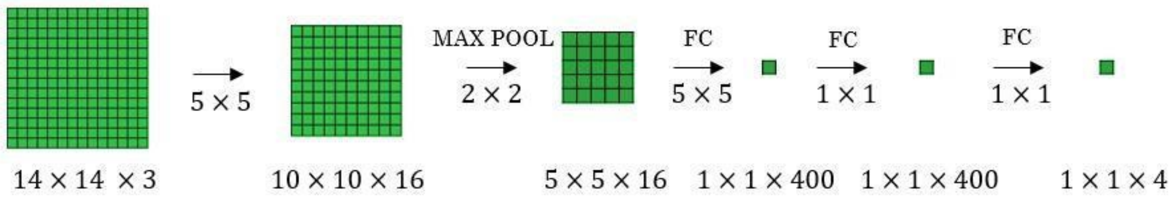


Figure 2.5

Expect that the measure of the input picture is 16 × 16 × 3rd portion of the framework and is passed to ConvNet. In any case, instep of doing this, we bolster the whole picture (such as 16 × 16 × 3) straightforwardly into the prepared ConvNet (see Figure 2.6). This comes about in an yield lattice of 2×2×4. Each cell of the yield network speaks to the editing esteem and the conveyance esteem of the editing picture. For illustration, the cleared out cell (green) of the yield lattice in Figure 2.6 speaks to the comes about of the to begin with sliding window. Other cells in the lattice speak to the comes about of the remaining window capacities.

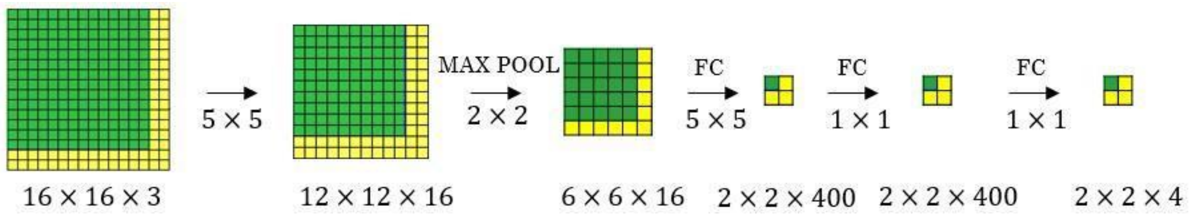


Figure 2.6

The pitch of the sliding window is decided by the number of channels utilized in the Max Pool layer. In the case over, the max pooling layer has two channels, coming about in the sliding window being moved in steps of 2, permitting for four thoughts per yield. The fundamental advantage of utilizing this strategy is that the sliding window runs and calculates all comes about at the same time. Subsequently, this handle is very quick. The impediment of this strategy is that the area of the checkbox is not exceptionally precise.

2.3YOLO Algorithm

2.3.1 Introduction to YOLO

YOLO algorithm is a better algorithm that can solve the problem of accuracy of box prediction when using convolutional sliding window technology. YOLO is the short form of Look Only Once and was founded in 2015 .

This is popular because it provides accuracy while operating on the fly. The algorithm only needs one propagation step through the network to make predictions.

2.3.2 Grid Division and Localization Algorithm

This approach entails partitioning the image into grids, subsequently implementing image segmentation and localization algorithms on each grid cell, as elaborated in the Object Localization section. For instance, consider an input image with dimensions of 256×256.

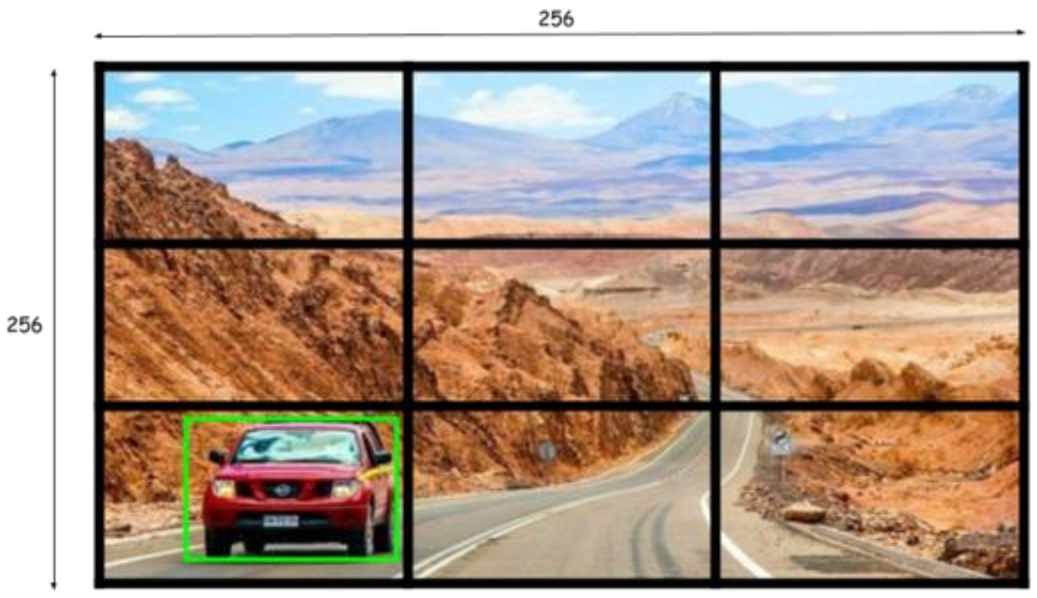


Figure 2.7

2.3.3 Model Output and Target Variable

Let's continue by executing the picture classification and
localization calculation on person framework cells. Inside each lattice cell of the picture,
we characterize the target variable as
Yi,j = [pcbxbybhbwc1c2c3c4]T. This comprises a vector of estimate 1 × 9. With a sliding window convolution approach, we'll handle this prepare. Considering the target variable shape for each lattice cell is 1 × 9, and there are 9 (3 × 3) framework cells, the
model's last yield will comprise of...

Final Output = 3x3x9

2.3.4 Advantages of YOLO Algorithm

The advantage of the YOLO algorithm is that it is very fast and the bounding box estimation is more accurate. Also in practice, to get more accurate predictions, we use a better grid such as 19×19 if the target display has a 19×19×9 image.

CHAPTER-3: YOLO OBJECT DETECTION

3.1 What is Object Detection?

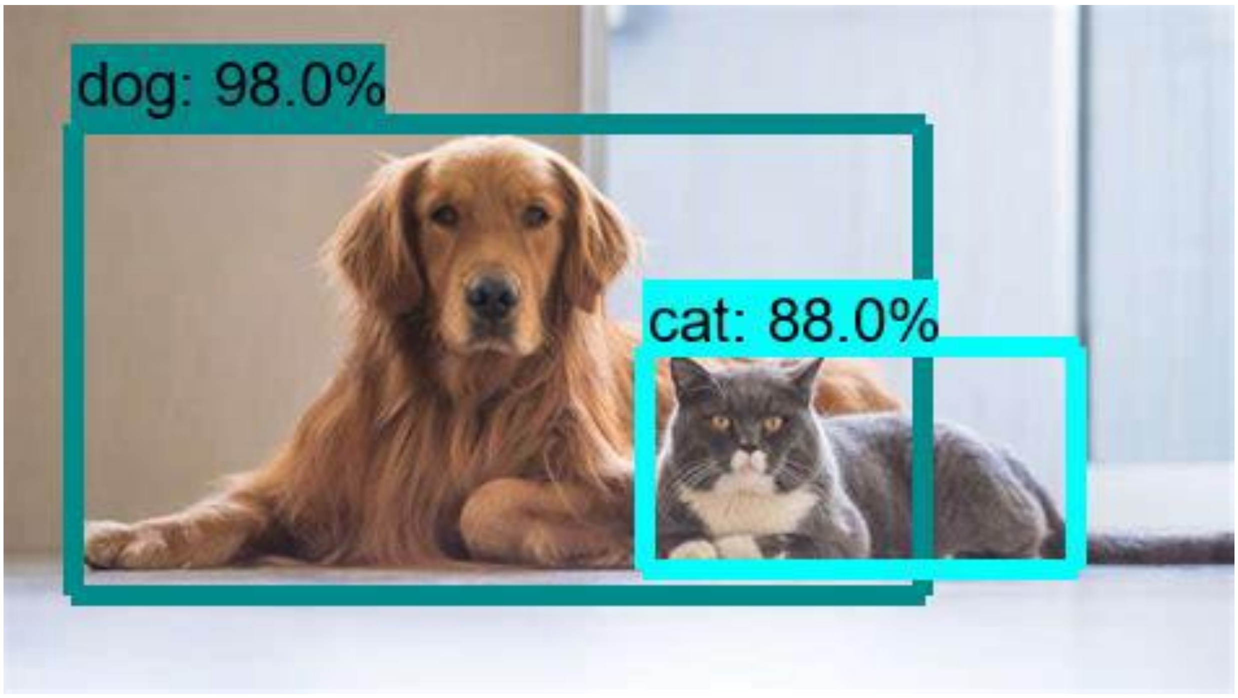


Figure 3.1

Object detection is a computer vision technology in which a software system can detect, locate and track objects through images or videos. The specific tool of object detection is to identify the class of an object (person, table, chair, etc.) and its specific function in the image. Show the location by drawing a box around the object. The checkbox for the item may or may not be displayed. The ability to find objects in an image reflects the performance of the algorithm used for detection. Face detection is an example of object detection.

These search engines may be pre-trained or trained from scratch. In most applications, we use pre-training weights from pre-trained models and then fine-tune them according to our needs and different applications.

3.2 How does product detection work?

In this section, we will briefly introduce the different methods used in product search. There are two ways to search for items; These are:

1- single shot detection

2- Two Shot detection -

As the name suggests, this method has two stages. The first of these is recommendations for the region, and then in the second stage, these areas are separated and estimated location optimization is made.

The Faster-RCNN variant is a popular choice for four models. In the regional concept phase, we use networks such as ResNet50 as feature extractors. We achieve this by removing the last layer of the mesh and using only the layers to remove features from the image. This is often a better method since the network has already been trained and can remove features from the image. A small network then goes on to use cross-specific techniques to predict category-independent frame recommendations based on a grid of tiling patterns by area, scale, and aspect ratio.

In the second stage, these boxes are required to be collected from the middle map calculated in the first stage. The ready box is fed into the rest of the feature extractor, where the prediction and regression heads are added on top of the network. Finally, we get category and category-specific box refinements for each box in the output.

In contrast, a search bypasses the request level of the field and results in final field and point estimates on the fly. YOLO is a popular example of this approach, and we'll talk about how it works in the next section.

It should be noted that the dual detection model achieves better performance, but single shot is at the sweet spot in terms of performance and speed/resource, making it suitable for checking products in food. Processes such as product tracking are where prediction is more important.

3.3 What is the purpose of YOLO?

As mentioned earlier, YOLO stands for "You Only Look Once" and is a single search engine proposed by Joseph Redmon in May 2016. It is a search algorithm.

Compared to other one-shots, YOLO performs surprisingly well in terms of speed and accuracy. It's not the most accurate algorithm when it comes to detecting objects, but it makes up for it with its speed, so it provides a good balance between speed and accuracy.) YOLO object detection algorithm overview

YOLO network divides the input image into an S × S grid. If the center of the ground truth falls on a cell, the cell is responsible for detecting the object.

All grid cell predict B bounding boxes and their negative scores and estimated rankings as shown below:

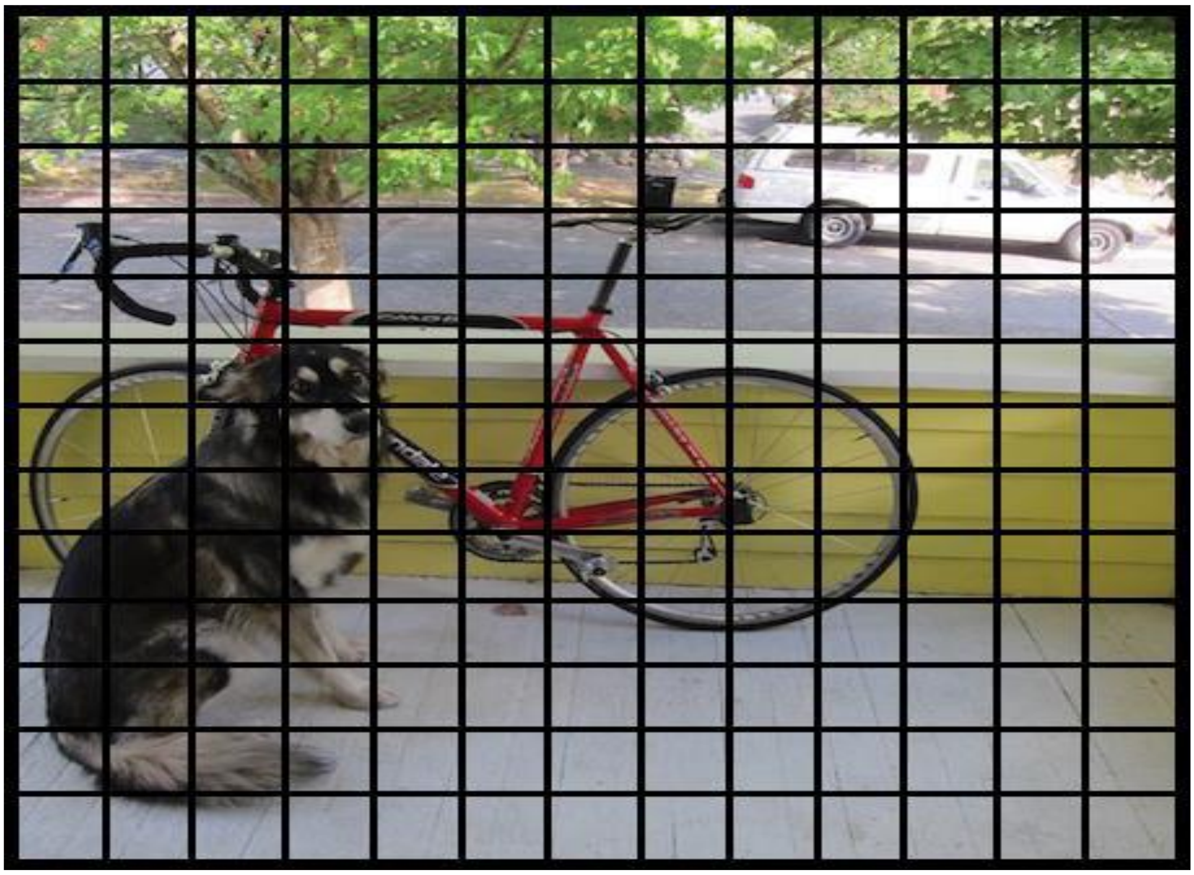


Figure 3.2

- B Facilitates of bounding boxes – YOLO gauges the 4 arranges (bx, by, bw, bh) of each box based on the bounding lines. Here bx are the x and y arranges of the midpoint of the kin relative to the lattice. The bh esteem is the proportion of the stature of the bounding box to the stature of the network cell, and the bw esteem is the proportion of the width of the bounding box to the width of the lattice cell.
- The result of the availability of goods(Objectness Score (P0))- The objectness score is passed through a sigmoid process and evaluated as a pro bability with a value ranging from 0 to 1.

- Class prediction – if the bounding box contains an object, the network predicts the probability of K number of classes.

The bounding box will look like this (the higher the confidence level, the thicker the box is drawn):

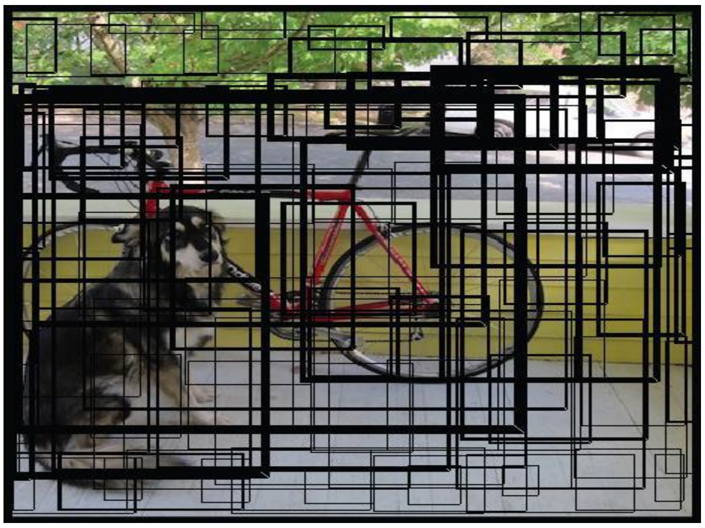


Figure 3.3

Finally, the bounding box's confidence and class predictions are combined into a final score that tells us the probability that the bounding box contains an object of a certain type. For example, the big yellow box on the left actually contains "black" items:

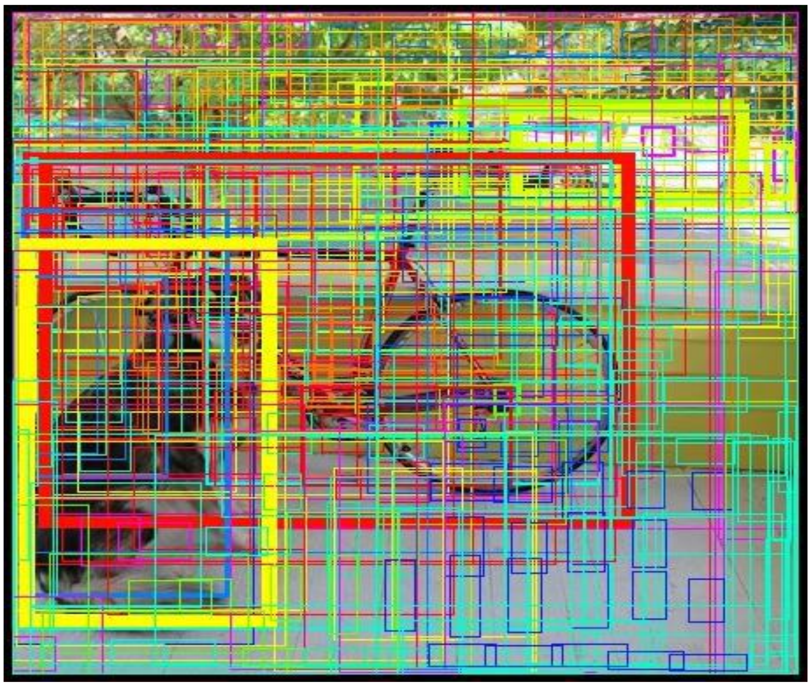


Figure 3.4

It turns out that most of these boxes have very low scores, so we only keep the boxes with the highest final score above the threshold. Additionally, unlimited detection (NMS) is designed to solve the problem of multiple detection of the same image. We will explain this briefly in the next section. So the final prediction is:

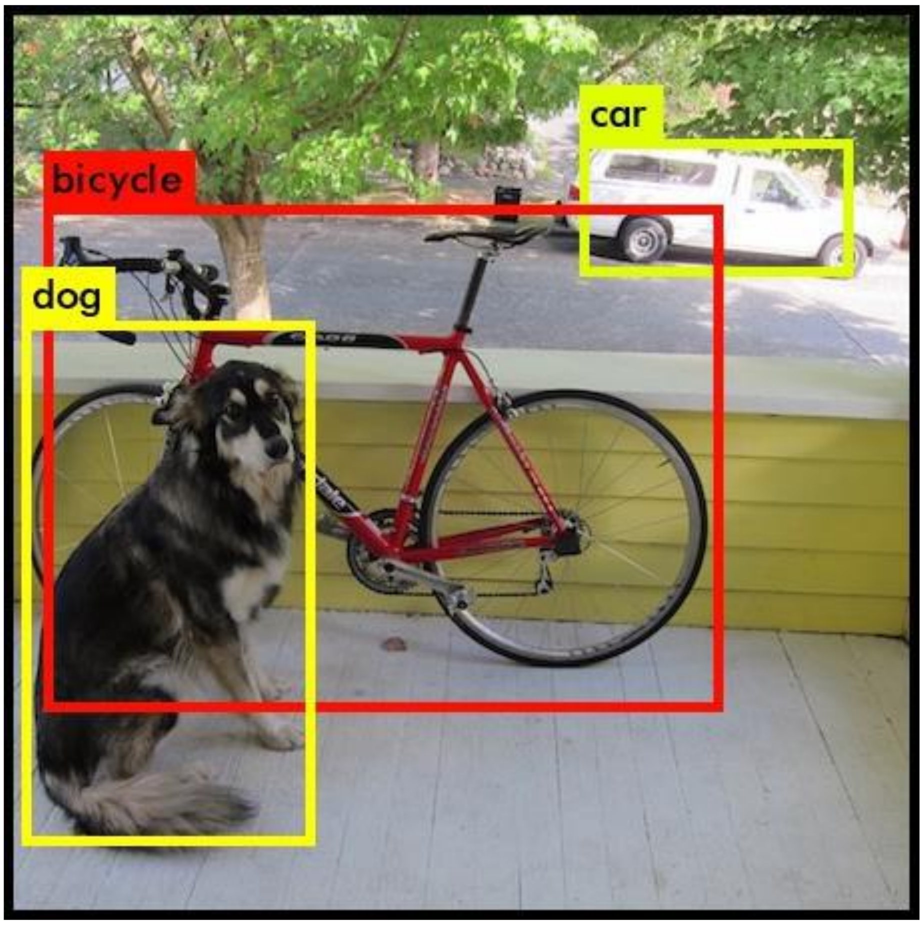


Figure 3.5

It should be noted that before V3, YOLO used the softmax function to calculate the ranking score. In V3 the author decided to use sigmoid instead. This is because Softmax imposes the assumption that each container has only one class, which is usually not the case. So if an object belongs to one class, it is guaranteed not to go to another class. While this assumption may be true for some datasets, it doesn't work when we have categories like Woman and Person. The multi-label process can model the product accurately. This is why the author did not use Softmax initially.

3.4 Non-maximum Suppression

Non-maximum Suppression or NMS uses the very important function called "Intersection over Union", or IoU. Here is how we calculate IoU.

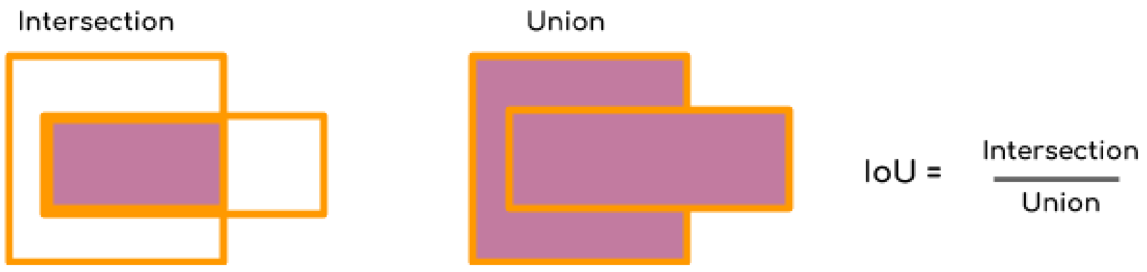


Figure 3.6

We define the box using the two corners (top left and bottom right) (x1,y1,x2,y2) instead of the center and height/width. Next, we also need to find the intersection of the two boxes (xi1, yi1, xi2, yi2); where:

xi1 = maximum value of x1 coordinates of two boxes

yi1 = maximum value of y1 coordinates of two boxes

xi2 = minimum value of x2 coordinates of two boxes

yi2 = minimum value of y2 coordinates of two boxes

Remember that To calculate the area of a rectangle (or box), we multiply its height (y2 – y1) by its width (x2 – x1).

So to calculate IoU first calculate the intersection area with this formula

area_intersection =(xi2 – xi1)*(yi2 – yi1)

Then calculate the union area

union_area = (area of box 1 + area of box 2) – field_intersection

So IoU=area_intersection/union_area

Now, to do Non maximum 22uppression, the steps are:

1. Select box with highest score.

2. Calculate its overlap with all other boxes and remove boxes that overlap more than a threshold (let's call it iou_threshold).

3. Return to step 1 and repeat until there are no boxes left with a lower score than the currently selected box.

This procedure will remove boxes with a size that overlaps the selection box. Only the best boxes.

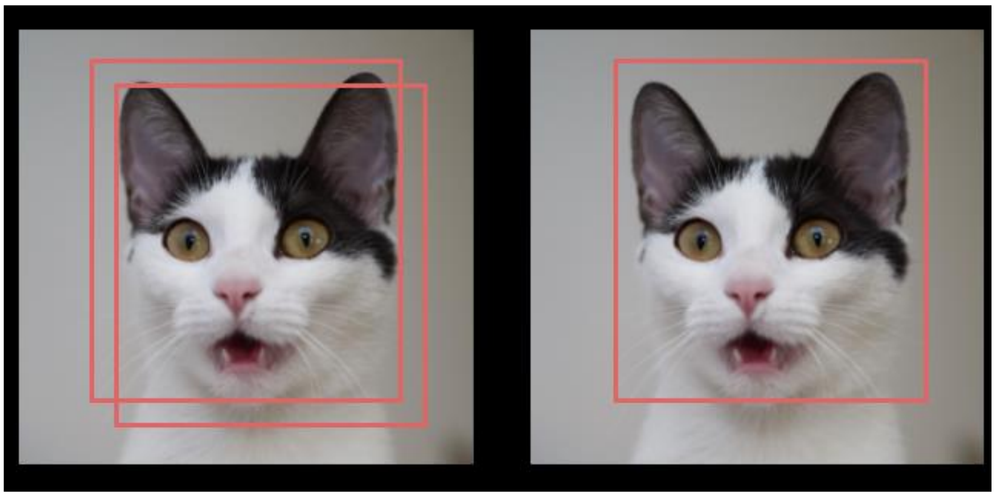


Figure 3.7

3.5 Using YOLO using OpenCV

There are many ways to use the YOLO algorithm, the most popular of which is Darknet. But here we will use OpenCV to implement the YOLO algorithm because it is very simple. First of all, you need to install OpenCV on your computer using this command on the command line.

```
Pip install opencv-python
```

YOLO from OpenCV, we need three files: yoloV3.weights, yoloV3.cfg and yoloV3.cfg coco.names (tag full list) This model is trained on labels). Now open a python file in this folder and start coding:

First of all, we will load the model using the “cv2.dnn.ReadNet()” function. Name the automatically found settings and frame.

CHAPTER 4: SYSTEM REQUIREMENTS

4.1 Introduction

We need Python in our PC to write and run the program. After that we need an Python IDE to easily execute the program. We will use PyCharm Community edition 2024.1 version IDE to run our program.

4.1.1 PYTHON:

Python is an interpreter based, object-oriented, high-level programming language with dynamic semantics. Its advanced data creation process combined with dynamic typing and dynamic linking makes it very attractive as a script or word for rapid application development and linking objects that are already together. Python’s easy-to-learn syntax emphasizes readability, reducing maintenance cost. Python supports modules and packages that support program modularization and code reuse. The Python interpreter and public libraries are freely available in source code or binary form for all major platforms and can be freely distributed.

4.1.2 PyCharm Community edition 2024.1

PyCharm Community edition 2024.1 is a free and open-source Integrated Development Environment (IDE) specifically designed for Python development by JetBrains. It provides a comprehensive set of features to enhance your Python coding experience, including:

Core functionalities:

This version of pycharm provides functionalities like:

Code Editing

Debugging

Testing

Version Control

Built-in Terminal

4.2 Complete System Requirements and Description

4.2.1 Overview

1-Install Python on your computer system

2-Install ImageAI and its dependencies like Numpy, OpenCV, pandas. TensorFlow,SciPy,Matplotlib,Argparse

3- Install GUI libraries like Tkinter and Pillow to make a integrated GUI to provide ease of access.

4-Download the Object Detection model file(Retinanet)

4.2.2. Steps to be followed

1- Download and install python latest version(python version 3.12) from below link-

https://www.python.org/downloads/

2- Install the following dependencies via pip:

I.OpenCV

OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV aims to provide a unified framework for computer vision and enable the use of machine vision in commercial products. OpenCV, a BSD licensed product, makes it easy for businesses to use and modify code. These algorithms can be used to identify and recognize faces, identify objects, classify people moving in videos, track camera movement, track moving objects, extract 3D models of objects, create 3D point clouds from a stereo camera, and combine images to decode images. Problems. OpenCV has a user community of over 47,000 people and is estimated to have been downloaded over 18 million times. The library is widely used by companies, research groups and government agencies. For example, Applied Minds, VideoSurf, and Zeitera) primarily use OpenCV. OpenCV has been applied to everything from stitching together Street View imagery, detecting intrusions into Israeli surveillance footage, tracking mining equipment in China, helping robots navigate and store items in the Willow Garage, discovering dead people in swimming pools in Europe, and conducting investigations. In the interactive art of Spain and New York, Turkey examines the garbage on the catwalks, examines the labels of products in factories around the world and discovers the fast face in Japan. Mac os operating system. OpenCV is primarily geared towards real-time visualization and leverages MMX and SSE instructions when available. Fully functional CUDA and OpenCL interfaces are currently under active development. There are over 500 algorithms and about 10 times more power creating or supporting those algorithms. OpenCV is written natively in C++ and has a templated interface that works seamlessly with STL containers.

Pip install opencv-python -command

OpenCV in the log

Unofficial pre-built CPU-only OpenCV package for Python.

If you want to compile data from code to support other modules (such as CUDA), please see the compilation section of the manual.

Installation and use

1. If you have installed a previous/another manually installed (=not installed by pip) version of OpenCV (e.g. the cv2 module in the root of the Python.site package), delete it before installation to avoid problems.

2. Make sure your pip version is correct (minimum supported version is 19.3): pip install –upgrade pip. Use pip -V to check the version. For example, Linux distributions often ship with legacy pips; This can cause many unexpected problems, especially in Manylinux mode.

3. Choose the right package for your environment:

There are four different packages from which you only have to choose one (see options 1, 2, 3 and 4 below). Do not install different packages in the same location. No plugin architecture: all packages use the same name (cv2). If you have many different packages installed in the environment, use pip uninstall to remove them all and then reinstall just one package.

Packages for standard graphics environments (Windows, macOS, almost all GNU/Linux distributions)

Option 1 – Main module package: pip install opencv-python

Option 2 – Full package (includes) main modules and contrib/extra modules): pip install opencv-contrib-python (check OpenCV file for listed contrib/extra modules)

Suitable for server (headless) environment (e.g. Docker, cloud environment, etc.), has GUI library does not have dependencies

This package is smaller than the two packages mentioned above because they do not have GUI functionality (no Qt/other GUI used). This means that these components avoid the heavy chain of X11 libraries, so you get a smaller Docker image. If cv2.imshow etc. If you are not using it, you should always use this package. Or you use a package other than OpenCV (like PyQt) to build the GUI.

Option 3 – Headless head module kit: pip install opencv-python-headless

Option 4 – Complete headless kit (including main module and additive/additional modules): pip install opencv- contrib-python-headless (check contribution/extra modules listed in OpenCV file)

Import packages:

import cv2

All packages include haarcascade archives. Cv2.data.haarcascades can be used as a shortcut to the data folder. Example:

cv2.CascadeClassifier(cv2.data.haarcascades + “haarcascade_frontalface_default.xml”)

II. NumPy

NumPy is a simple package for scientific calculations in Python. It is a Python library that provides fast processing, mathematical calculations, logic, efficient operations, sorting, selection, I/O, various tools for discrete operations, various features (such as masks and matrices), and various functions. Fourier transform, simple linear algebra, simple statistical operations, stochastic simulation, etc.

The core of the NumPy package is the ndarray object. It encompasses an N-dimensional array of homogeneous data types, and many tasks are performed in compiled code to improve performance. There are several important differences between NumPy arrays and standard Python arrays:

NumPy arrays are created at a fixed size, unlike Python lists (which can grow dynamically). Changing the size of Ndarray will create a new array and delete the original array.

The elements in a NumPy array must all have the same data type and therefore the same size in memory. Exception: May contain arrays of objects (including Python, NumPy), which allows arrays of different objects.

NumPy arrays facilitate advanced arithmetic and other types of operations on large data sets. In general, these functions are more efficient and involve less code than using Python's built-in arrays. Although these generally support Python ad-hoc input, they convert the input to NumPy arrays before processing and often output NumPy arrays. In other words, to effectively use most (or most) of today's Python-based scientific/mathematical software, it is not enough to know how to use Python's built-in compatibility mode; It is also necessary to know how to use NumPy arrays. .

pip install numpy -command

Description

Provides:

1.A powerful N-dimensional array object

2.Complex (post) functions

3.Tools for integrating C/C++ and Fortran code

4.Uses linear algebra, Fourier transform, and random number functions and more

Beyond its obvious technical uses, NumPy can be used as a multi-purpose container for general information. Any type of data can be identified. This allows NumPy to integrate seamlessly and quickly with many libraries.

All NumPy logs distributed for PyPI are licensed under the BSD license.

III. TensorFlow

Tensorflow is an open source software library for data flow and variable programming in a wide range of applications. Neural networks etc. It is a code library also used in machine learning. It was released under the Apache License 2.0 on November 9, 2015 , developed by Google Brain. Tensorflow can run on multiple CPUs and GPUs on a single device (with optional CUDA and SYCL extensions to compute the overview of the running image). TensorFlow is available on multiple platforms including 64-bit Linux, macOS, Windows, and mobile computers including Android and iOS. , GPU, TPU) and desktop server stack to mobile and edge devices. The name Tensorflow comes from the work that neural networks do on different parts of data called tensor.

Pip install tensorflow -command

IV. Pandas

pandas is a Python package that provides fast, flexible and expressive data structures designed as a set of processes (tabular, multidimensional, possibly heterogeneous) and whose time series data is simple and intuitive. It aims to be a lightweight building block for practical, real-world documentation in Python. It also has a broader goal of being the most powerful and flexible open source data analysis/management tool in any language. He went to this goal.

Pandas is well suited for many types of data:

– tabular data with different types of columns, such as SQL tables or Excel spreadsheets

– Ordered and Unordered ((not necessarily frequency) time series data.

- Data matrix with rows and columns (homogeneous type or heterogeneous)

- Other analysis/statistics methods Actually, the data does not need to be fully marked before being placed in the panda data structure.

Pandas' two main data models, Series (one-dimensional) and DataFrame (two-dimensional), are used in many financial, statistical and social studies, and for R users in various engineering fields. Pandas is based on NumPy and is designed to integrate well with many other third-party libraries in business analytics.

Here are some things pandas are good at:

- Easily handle missing data (like NaN) in floating point and non-floating data

Differences: Rows vs DataFrames and advanced objects can be added and removed exactly with the list or user can ignore the list and add Series, DataFrame etc. may allow listing of items. Calculate Automatically align data for you

- Powerful, flexible batch capability to perform split-apply-join operations on datasets to aggregate and transform data

- Easily put other Python Jagged, diverse transform indexed data Convert from NumPy data structure to DataFrame objects

- Smart tag-based slicing, beautifully indexing and subsetting large datasets

- Intuitive merging and combining datasets

- Easy dataset reshaping and rotation

- Hierarchical labels for axes (can have multiple part labels to mark)

- Powerful to load and/save data from flat files (CSV and delimited), Excel files, libraries IO tools transfer data from ultra-fast HDF5 format

- Time series specific features: date generation and frequency conversion, change window statistics, date change and delay.

Most of these principles are intended to address shortcomings often encountered when using other languages/research sites. Data processing for data scientists is usually divided into several stages: organizing and cleaning the data, analysis/design, and then arranging the results display in a format suitable for layout or tabular figures. Pandas is the best tool for all these tasks.

Pip install pandas – command

V. SciPy

SciPy has numerous modules for optimization, straight variable based math, integration, insertion, extraordinary capacities, FFT, flag and picture preparing, Tribute solvers, and other assignments common in designing. SciPy abstracts center on NumPy cluster objects and are portion of the NumPy gather, which incorporates instruments such as Matplotlib, pandas, and SymPy, as well as an expansion of the logical library. This NumPy stack works essentially to other applications such as MATLAB, Octave, and Scilab. NumPy stacks are in some cases called SciPy stacks. It is moreover backed by NumFOCUS, a community establishment that underpins the creation and get to of research.

Pip install scipy -command

VI. Matplotlib

Matplotlib is a Python programming language graphics library and its NumPy code extension. Tkinter provides an object-oriented API for drawing into applications using GUI tools such as wxPython, Qt, or GTK+.

Pip install matplotlib -command

VII. Pillow

Python Image Library is a free Python programming language library that provides support for opening, editing, and saving images in various formats. Windows, Mac OS X and Linux all run here.

Pip installpillow -command

VIII. Argparse

The argparse module makes it easy to write user-friendly command line interfaces. The program interprets the arguments it needs and argparse will determine how to parse the arguments from sys.argv. The argparse module also gets help and uses words and throws errors when the user supplies the wrong words to the program.

Python As of >= 2.7 and >= 3.2, the argparse module is stored in Python's standard library. For users who still need Python < 2.7 or < 3.2 support, it is also available as a separate package that attempts to be compatible with models in the standard library but also supports older Python versions

Pip install argparse – command

IX. Tkinter

Tkinter is a template library for GUI applications in Python. Tkinter has many controls for creating GUI applications. Tkinter comes when we install Python. While installing Python, we need to select the td/tk and IDLE checkboxes. This will install tkinter, we don't need to install it separately. Installed point of use. Run the following commands in the command prompt to install Tkinter.

Pip install tk – command

Pip install tkintertable – command

4.3 Darknet

Darknet is an open source neural organize system composed in C and CUDA. It is quick, simple to introduce, and underpins CPU and GPU computation. You'll discover the source on GitHub.

YOLO is one of the module of Darknet

You only see once (YOLO) may be a state-of-the-art, real-time protest discovery system. On a Pascal Titan X it forms pictures at 30 FPS and includes a mAP of 57.9% on COCO test-dev.

i. How It Works

Prior location frameworks repurpose classifiers or localizers to perform discovery. They apply the show to an picture at different areas and scales. High scoring locales of the picture are considered detections.

We utilize a completely distinctive approach. We apply a single neural arrange to the total picture. This arrange isolates the picture into locales and predicts bounding boxes and probabilities for each locale. These bounding boxes are weighted by the predicted probabilities.

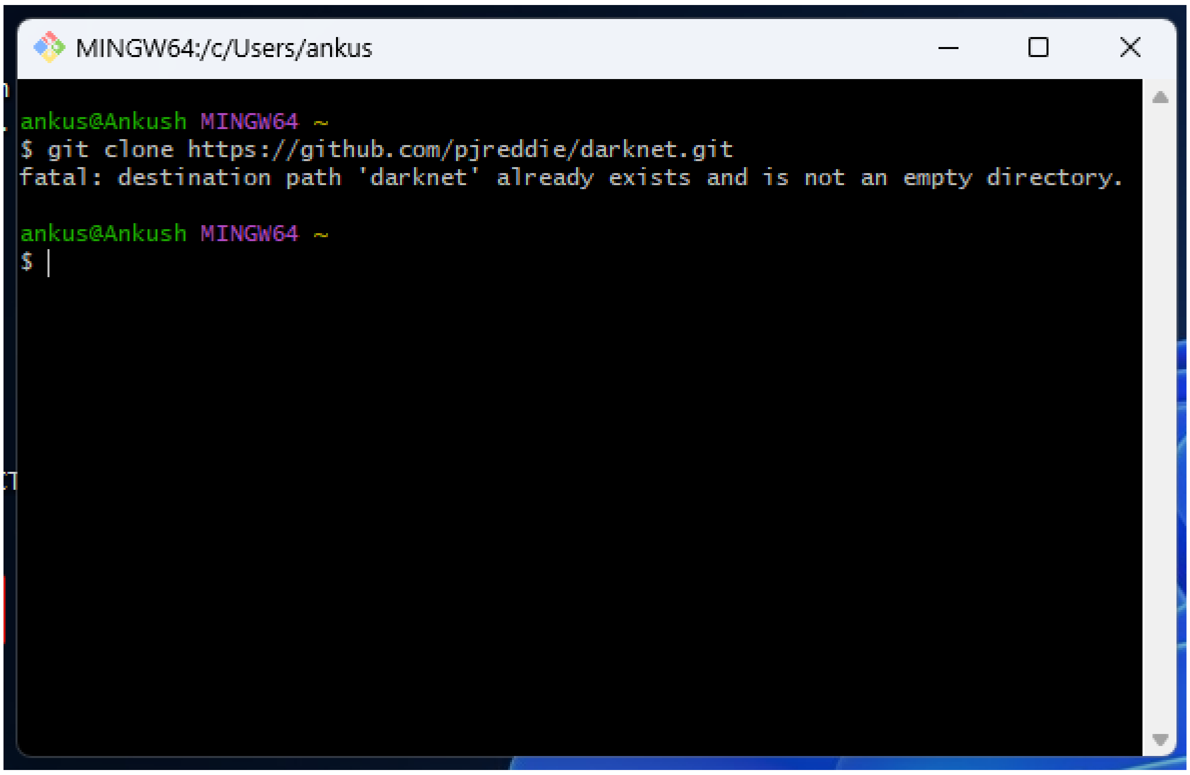
Our model has a few preferences over classifier-based frameworks. It looks at the entire picture at test time so its forecasts are educated by worldwide setting within the picture. It too makes expectations with a single network evaluation not at all like frameworks like R-CNN which require thousands for a single picture. This makes it amazingly quick, more than 1000x faster than R-CNN and 100x quicker than Quick R-CNN. See our paper for more points of interest on the complete system.

ii. Discovery Employing A Pre-Trained Model

Here we are going direct you through recognizing objects with the YOLO framework employing a pre-trained model. In the event that you don't as of now have darknet introduced, you ought to do that to begin with utilizing underneath source .

First you have got to install The GIT Bash if you haven't, utilizing this underneath link

https://git-scm.com/downloads



```
ankus@Ankush MINGW64 ~
$ git clone https://github.com/pjreddie/darknet.git
fatal: destination path 'darknet' already exists and is not an empty directory.

ankus@Ankush MINGW64 ~
$ |
```

Figure 4.1

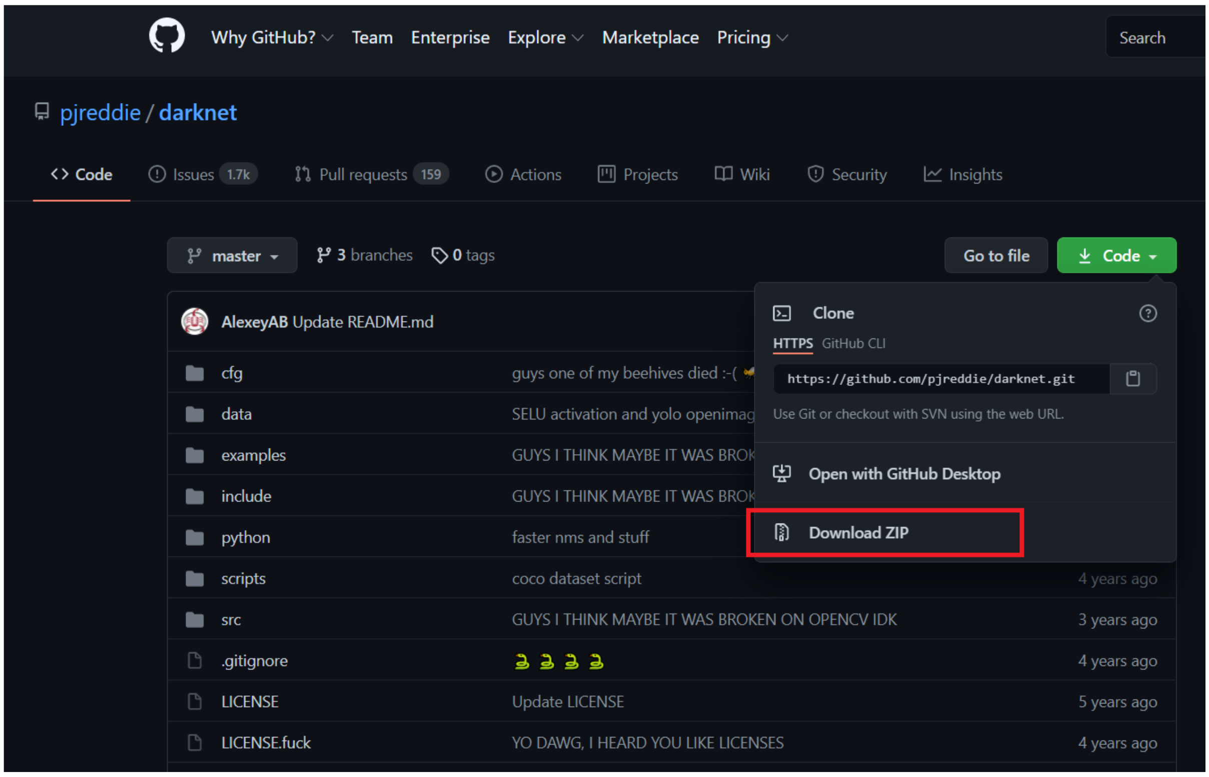
After Installing Git bash now we are going to install darknet using Git bash.

Run the below command in the git bash.

Git clone https://github.com/pjreddie/darknet.git

or if you want to install without using Git bash then we have to go to the github.com to download the file using below link.

https://github.com/pjreddie/darknet



```
https://github.com/pjreddie/darknet.git
```

Use Git or checkout with SVN using the web URL.

AlexeyAB Update README.md

cfg	guys one of my beehives died :-(	
data	SELU activation and yolo openimag	
examples	GUYS I THINK MAYBE IT WAS BROK	
include	GUYS I THINK MAYBE IT WAS BROK	
python	faster nms and stuff	
scripts	coco dataset script	4 years ago
src	GUYS I THINK MAYBE IT WAS BROKEN ON OPENCV IDK	3 years ago
.gitignore	🐍 🐍 🐍 🐍	4 years ago
LICENSE	Update LICENSE	5 years ago
LICENSE.fuck	YO DAWG, I HEARD YOU LIKE LICENSES	4 years ago

Figure 4.2

Click download zip to download the zip file from the darknet. Once the download is complete, you should unzip the file and put it in c:\user\yoursystemname\

You already have your YOLO configuration file in the cfg/ subdirectory. You should download the pre-weight trained folder from here.

https://pjreddie.com/media/files/yolov3.weights

After the download is completed, you must paste the file into the darknet folder. Git bash

git clone https://pjreddie.com/media/files/yolov3.weights.git

Note:

If you want to know what to do with this Weight and config then the neural network architectural model is inside yolov3.cfg file and the weights trained before are stored in the yolov3.Weights file.

iii. Coco.names

The coco.names record contains the names of the diverse objects that our demonstrate has been prepared to distinguish. We store them in a list called classes.

Below is the names of coco.names file

person

bicycle

car

motorbike

aeroplane

bus

train

truck

boat

traffic light

fire hydrant

stop sign

parking meter

bench

bird

cat

dog

horse

sheep

cow

elephant

bear

zebra

giraffe

backpack

umbrella

handbag

tie

suitcase

frisbee

skis

snowboard

sports ball

kite

baseball bat

baseball glove

skateboard

surfboard

tennis racket

bottle

wine glass

cup

fork

knife

spoon

bowl

banana

apple

sandwich

orange

broccoli

carrot

hot dog

pizza

donut

cake

chair

sofa

pottedplant

bed

diningtable

toilet

tvmonitor

laptop

mouse

remote

keyboard

cell phone

microwave

oven

toaster

sink

refrigerator

book

clock

vase

scissors

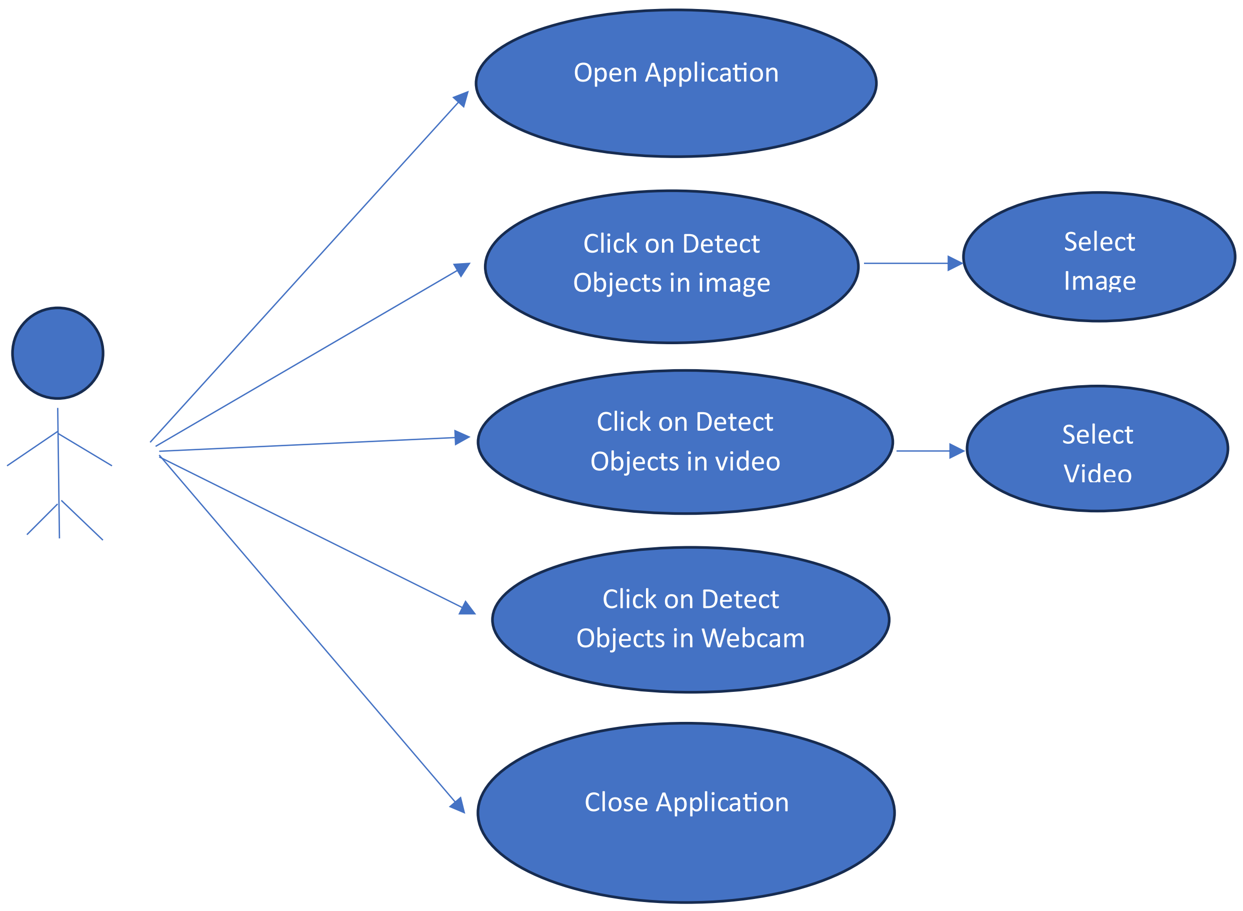
teddy bear

hair drier

toothbrush

CHAPTER 5: SYSTEM DESIGN

5.1 Use Case Diagram



Open Application
Click on Detect Objects in image
Select Image
Click on Detect Objects in video
Select Video
Click on Detect Objects in Webcam
Close Application

Figure 5.1

5.2. Data Flow Diagram

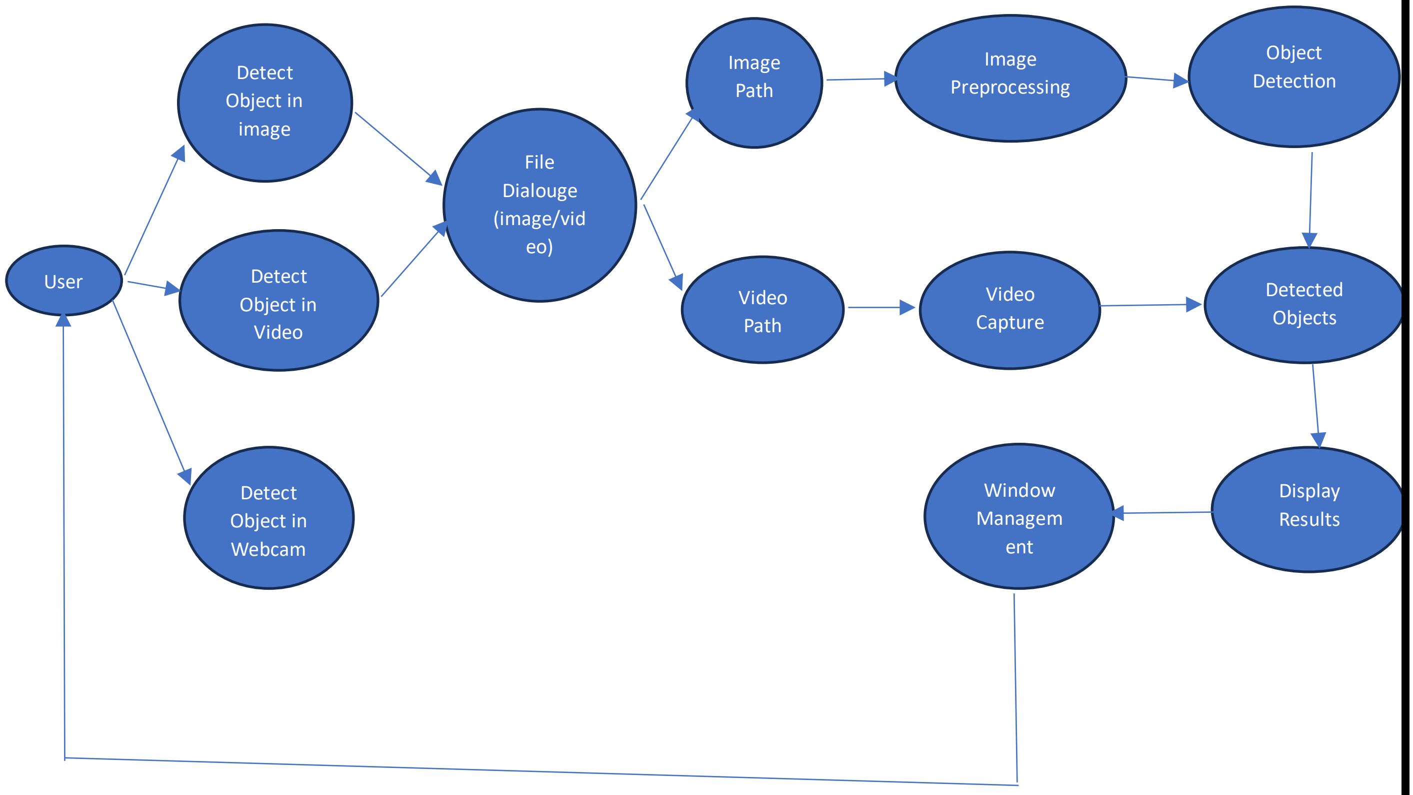


Figure 5.2

5.3. Sequence Diagram

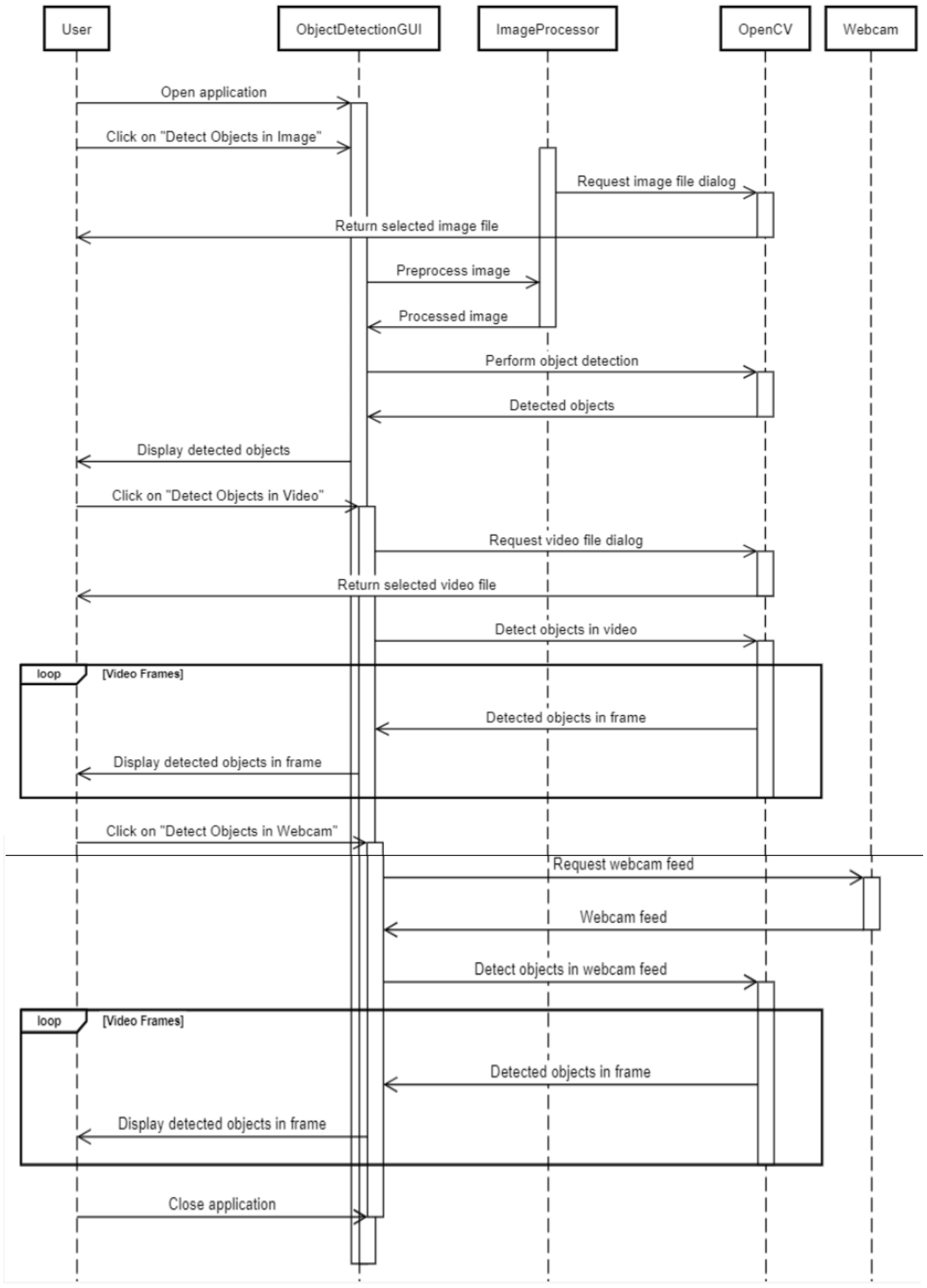

User
ObjectDetectionGUI
ImageProcessor
OpenCV
Webcam
Open application
Click on "Detect Objects in Image"
Request image file dialog
Return selected image file
Preprocess image
Processed image
Perform object detection
Detected objects
Display detected objects
Click on "Detect Objects in Video"
Request video file dialog
Return selected video file
Detect objects in video
loop
[Video Frames]
Detected objects in frame
Display detected objects in frame
Click on "Detect Objects in Webcam"
Request webcam feed
Webcam feed
Detect objects in webcam feed
loop
[Video Frames]
Detected objects in frame
Display detected objects in frame
Close application

Figure 5.3

CHAPTER 6: CODING

6.1 Coding

```
“import tkinter as tk
from tkinter import filedialog
import cv2
import numpy as np
from PIL import Image, ImageTk
import threading

class ObjectDetectionGUI:
    def __init__(self, root):
        self.root = root
        self.root.title(“Object Detection”)
        self.window_width = 500
        self.window_height = 314

        # Load background image
        self.load_background_image()

        # Create frame to hold buttons
        self.button_frame = tk.Frame(self.root, bg=”white”)
        self.button_frame.pack(pady=10)

        # Create buttons for different functionalities
        self.image_button = tk.Button(self.button_frame, text=”Detect
Objects in Image”, command=self.detect_objects_in_image)
        self.image_button.pack(side=”left”, padx=10)

        self.video_button = tk.Button(self.button_frame, text=”Detect
Objects in Video”, command=self.detect_objects_in_video)
        self.video_button.pack(side=”left”, padx=10)
        self.webcam_button = tk.Button(self.button_frame, text=”Detect
Objects in Webcam”, command=self.detect_objects_in_webcam)

        self.webcam_button.pack(side=”left”, padx=10)


        # Initialize variables

        self.net, self.output_layers, self.classes =
self.load_yolov3_model()


    def load_background_image(self):

        # Load background image

        img = Image.open(“yolologo2.png”)


        # Resize image to fit the window

        img = img.resize((self.window_width, self.window_height),
Image.LANCZOS)


        # Convert image to PhotoImage

        self.background_image = ImageTk.PhotoImage(img)


        # Create canvas to display background image

        self.canvas = tk.Canvas(self.root, width=self.window_width,
height=self.window_height)

        self.canvas.pack(fill=”both”, expand=True)

        self.canvas.create_image(0, 0, anchor=tk.NW,
image=self.background_image)


    def load_yolov3_model(self):

        # Load YOLOv3 model and classes

        net = cv2.dnn.readNet(‘yolov3.weights’, ‘yolov3.cfg’)

        layer_names = net.getLayerNames()

        output_layers = [layer_names[i – 1] for i in
net.getUnconnectedOutLayers()]


        with open(“coco.names”, “r”) as f:

            classes = [line.strip() for line in f.readlines()]
        return net, output_layers, classes


    def preprocess_image(self, image_path):
        # Preprocess image
        img = cv2.imread(image_path)
        if img is None:
            raise FileNotFoundError(f"Image not found: {image_path}")
        blob = cv2.dnn.blobFromImage(img, 1 / 255.0, (416, 416),
swapRB=True, crop=False)
        return img, blob


    def perform_detection(self, frame):
        # Perform object detection
        height, width, _ = frame.shape
        blob = cv2.dnn.blobFromImage(frame, 1 / 255.0, (416, 416),
swapRB=True, crop=False)
        self.net.setInput(blob)
        outs = self.net.forward(self.output_layers)


        class_ids = []
        confidences = []
        boxes = []


        for out in outs:
            for detection in out:
                scores = detection[5:]
                class_id = np.argmax(scores)
                confidence = scores[class_id]
                if confidence > 0.5:
                    center_x = int(detection[0] * width)
                    center_y = int(detection[1] * height)
                    w = int(detection[2] * width)
                    h = int(detection[3] * height)
                x = int(center_x - w / 2)
                y = int(center_y - h / 2)
                boxes.append([x, y, w, h])
                confidences.append(float(confidence))
                class_ids.append(class_id)

        indexes = cv2.dnn.NMSBoxes(boxes, confidences, 0.5, 0.4)

        colors = np.random.uniform(0, 255, size=(len(self.classes), 3))

        for i in range(len(boxes)):
            if i in indexes:
                x, y, w, h = boxes[i]
                label = str(self.classes[class_ids[i]])
                color = colors[class_ids[i]]
                cv2.rectangle(frame, (x, y), (x + w, y + h), color, 2)
                cv2.putText(frame, label, (x, y + 30),
cv2.FONT_HERSHEY_SIMPLEX, 1, color, 2)

        return frame

    def detect_objects_in_image(self):
        # Detect objects in an image
        image_path = filedialog.askopenfilename()
        if image_path:
            img, _ = self.preprocess_image(image_path)
            frame = self.perform_detection(img)
            cv2.imshow("Object Detection", cv2.resize(frame, (1280, 780)))
# Adjust window size here
            cv2.waitKey(0)
            cv2.destroyAllWindows()

    def detect_objects_in_video(self):
        # Detect objects in a video
        video_path = filedialog.askopenfilename()
        if video_path:
            video_capture = cv2.VideoCapture(video_path)
            while True:
                ret, frame = video_capture.read()
                if not ret:
                    break
                frame = self.perform_detection(frame)
                cv2.imshow("Object Detection", cv2.resize(frame, (1280,
780)))  # Adjust window size here
                if cv2.waitKey(1) & 0xFF == ord('q'):
                    break
            video_capture.release()
            cv2.destroyAllWindows()


    def detect_objects_in_webcam(self):
        # Detect objects in webcam feed
        video_capture = cv2.VideoCapture(0)
        while True:
            ret, frame = video_capture.read()
            if not ret:
                break
            frame = self.perform_detection(frame)
            cv2.imshow("Object Detection", cv2.resize(frame, (1280, 780)))
# Adjust window size here
            if cv2.waitKey(1) & 0xFF == ord('q'):
                break
        video_capture.release()
        cv2.destroyAllWindows()


if __name__ == "__main__":
    root = tk.Tk()
    app = ObjectDetectionGUI(root)
    root.mainloop()"
```

6.2. Description of The Code

1-__init__(self, root): This is the constructor method that initializes the ObjectDetectionGUI class. It takes root as an argument, which is the root window of the tkinter application.

2-load_background_image(self): This method loads the background image for the GUI window. It opens the image file, resizes it to fit the window dimensions, converts it to a PhotoImage object, and creates a canvas to display the background image.

3-load_yolov3_model(self): This method loads the YOLOv3 model and associated classes. It reads the YOLOv3 weights and configuration files, extracts layer names, and reads class names from the COCO dataset.

4-preprocess_image(self, image_path): This method preprocesses an image for object detection. It loads the image, creates a blob for the YOLOv3 model input, and returns the image and blob.

5-perform_detection(self, frame): This strategy performs protest location on a given outline utilizing the YOLOv3 demonstrate. It takes a outline as input, recognizes objects in the outline, draws bounding boxes around them, names them with their lesson names, and returns the prepared frame.

6-detect_objects_in_image(self): This method detects objects in an image selected by the user. It opens a file dialog to select an image, preprocesses it, performs detection, and displays the result in a window.

7-detect_objects_in_video(self): This method detects objects in a video selected by the user. It opens a file dialog to select a video, captures frames from the video, performs detection on each frame, and displays the result in real-time.

8-detect_objects_in_webcam(self): This method detects objects in the live webcam feed. It captures frames from the webcam, performs detection on each frame, and displays the result in real-time.

CHAPTER 7: OUTPUT SCREENS

7.1 GUI Window Output

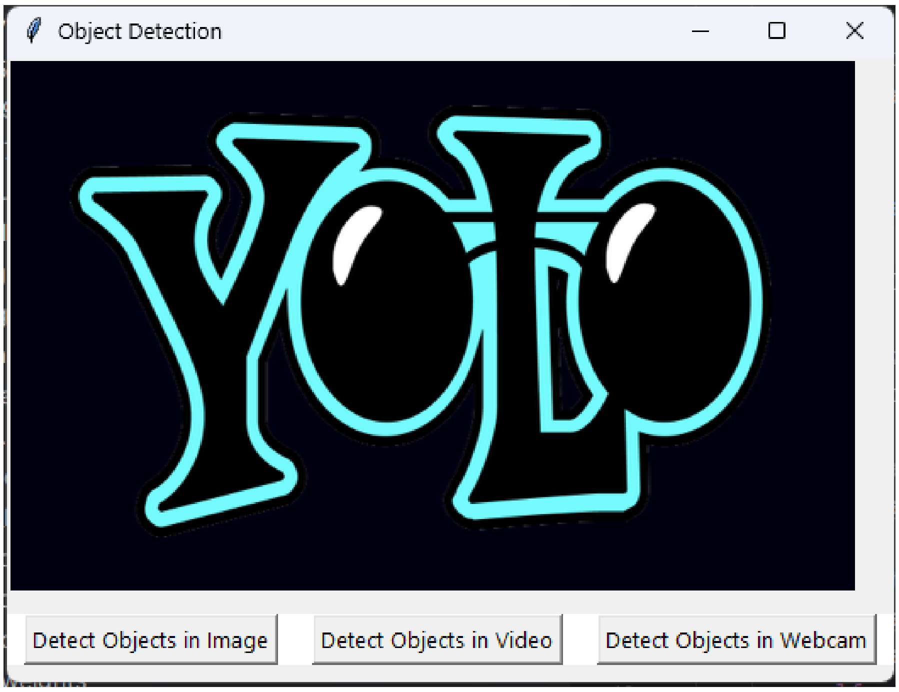


Figure 7.1

7.2 Object Detection in Images Output

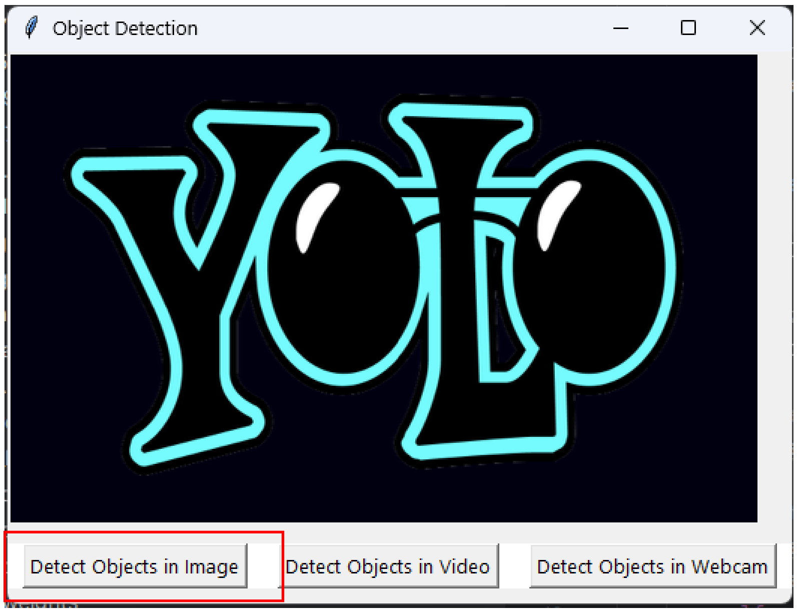
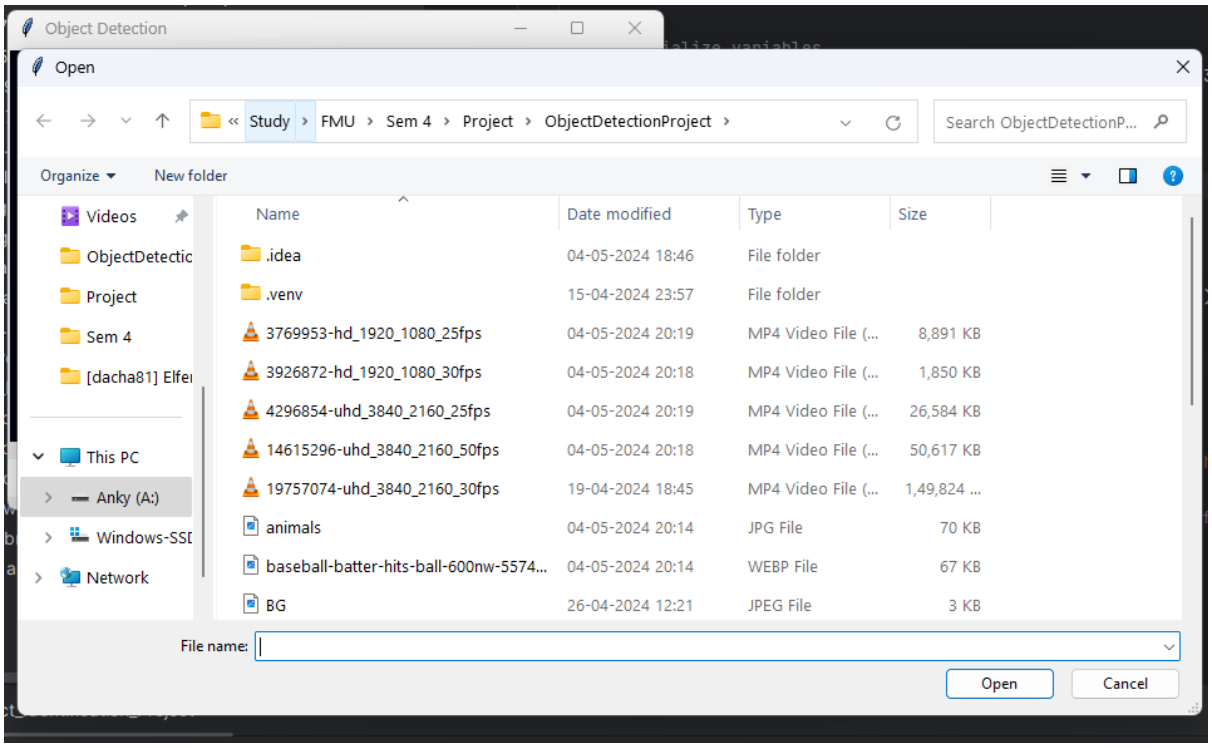


Figure 7.2

Now select image



Object Detection

Open

« Study › FMU › Sem 4 › Project › ObjectDetectionProject ›

Search ObjectDetectionP...

Organize ▾ New folder

Videos
ObjectDetectic
Project
Sem 4
[dacha81] Elfe
This PC
Anky (A:)
Windows-SSI
Network

Name	Date modified	Type	Size
.idea	04-05-2024 18:46	File folder	
.venv	15-04-2024 23:57	File folder	
3769953-hd_1920_1080_25fps	04-05-2024 20:19	MP4 Video File (...	8,891 KB
3926872-hd_1920_1080_30fps	04-05-2024 20:18	MP4 Video File (...	1,850 KB
4296854-uhd_3840_2160_25fps	04-05-2024 20:19	MP4 Video File (...	26,584 KB
14615296-uhd_3840_2160_50fps	04-05-2024 20:18	MP4 Video File (...	50,617 KB
19757074-uhd_3840_2160_30fps	19-04-2024 18:45	MP4 Video File (...	1,49,824 ...
animals	04-05-2024 20:14	JPG File	70 KB
baseball-batter-hits-ball-600nw-5574...	04-05-2024 20:14	WEBP File	67 KB
BG	26-04-2024 12:21	JPEG File	3 KB

File name:

Open Cancel

Figure 7.3

OUTPUT-1

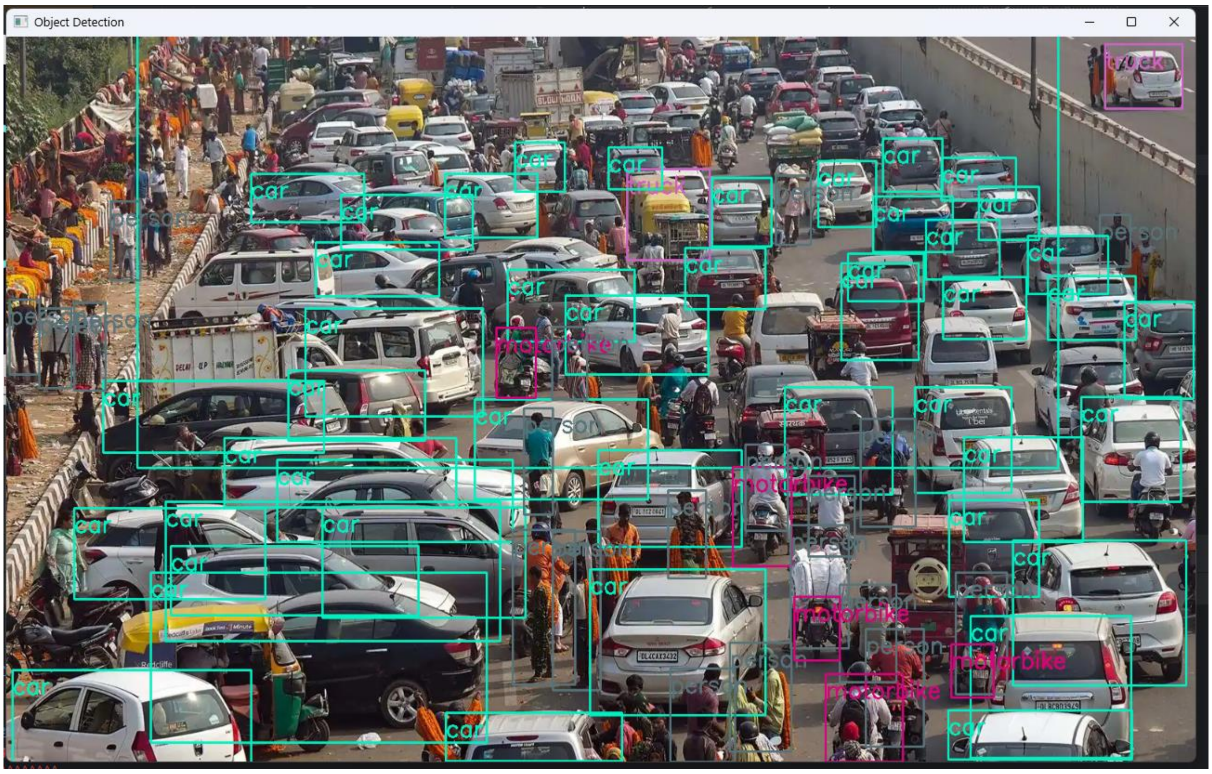


Figure 7.4

OUTPUT-2

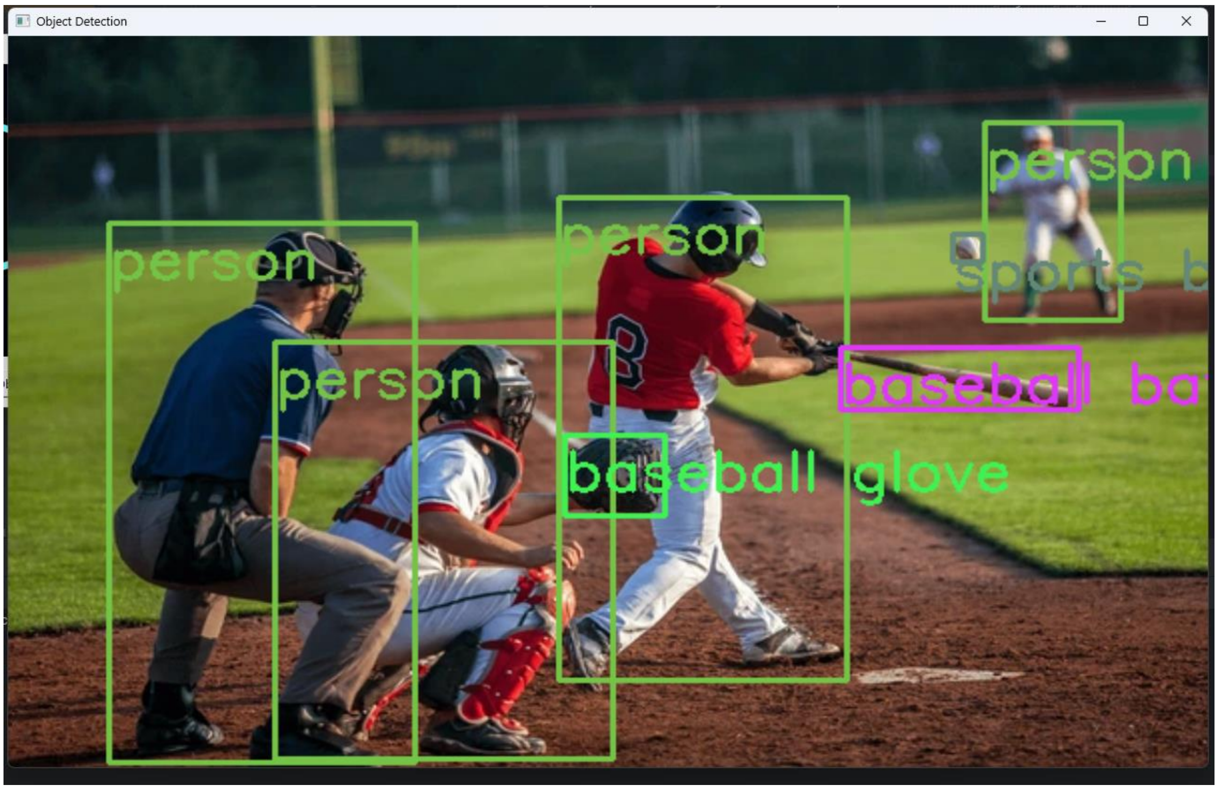


Figure 7.5

OUTPUT-3

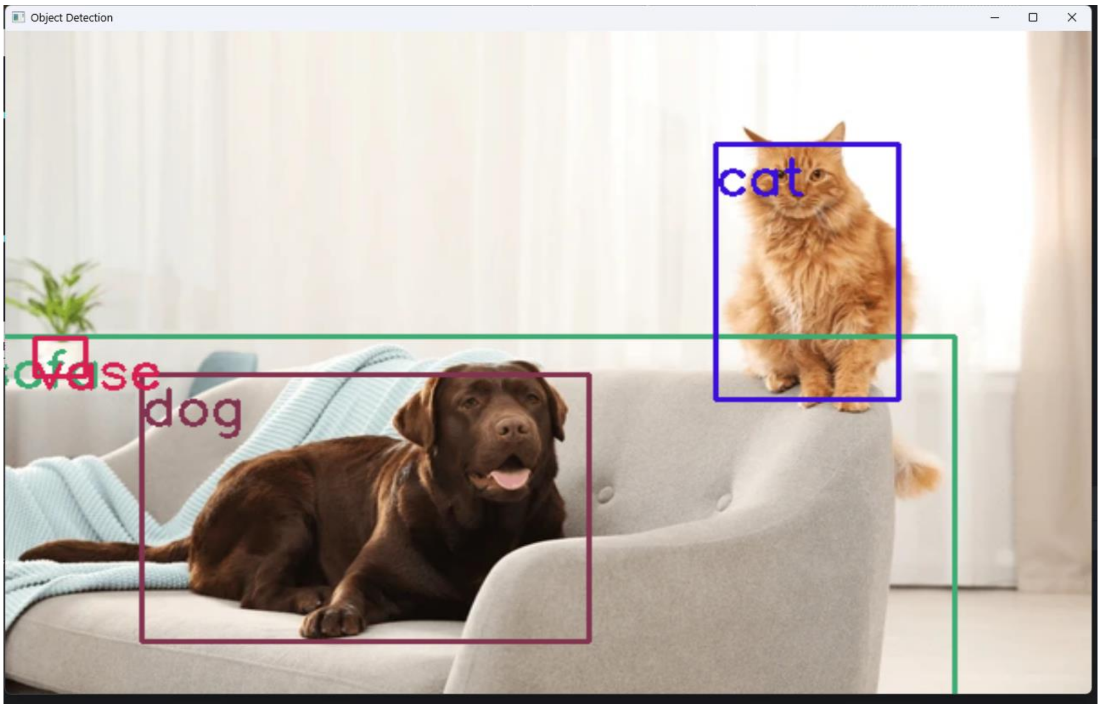


Figure 7.6

7.3 Object Detection in video Output

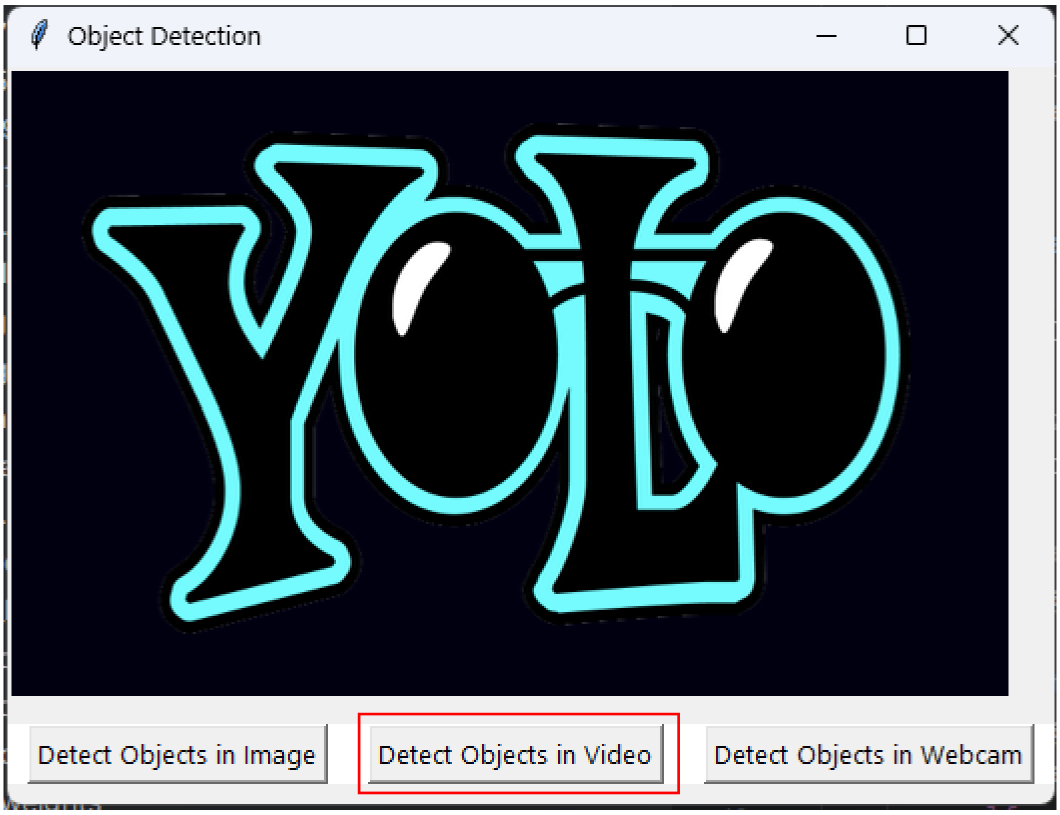


Figure 7.7

Now select Video

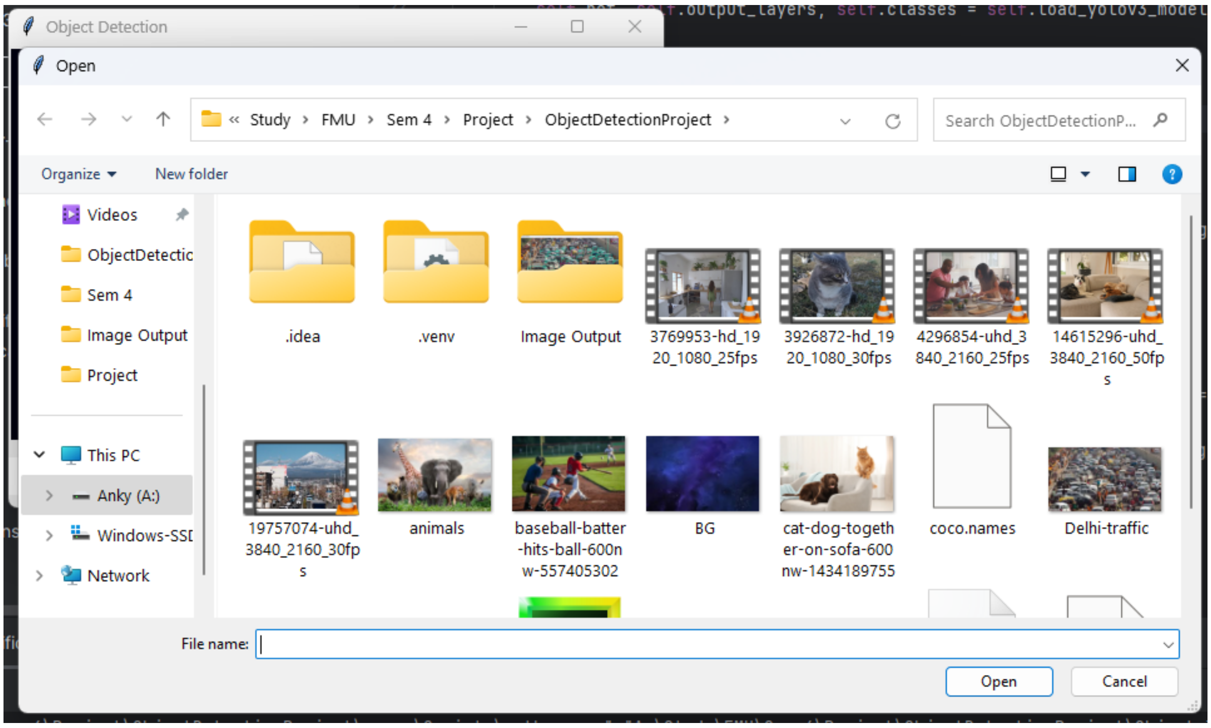

Object Detection
Open
« Study › FMU › Sem 4 › Project › ObjectDetectionProject ›
Search ObjectDetectionP...
Organize
New folder
Videos
ObjectDetectic
Sem 4
Image Output
Project
This PC
Anky (A:)
Windows-SSD
Network
.idea
.venv
Image Output
3769953-hd_1920_1080_25fps
3926872-hd_1920_1080_30fps
4296854-uhd_3840_2160_25fps
14615296-uhd_3840_2160_50fps
19757074-uhd_3840_2160_30fps
animals
baseball-batter-hits-ball-600nw-557405302
BG
cat-dog-together-on-sofa-600nw-1434189755
coco.names
Delhi-traffic
File name:
Open
Cancel

Figure 7.8

OUTPUT-1

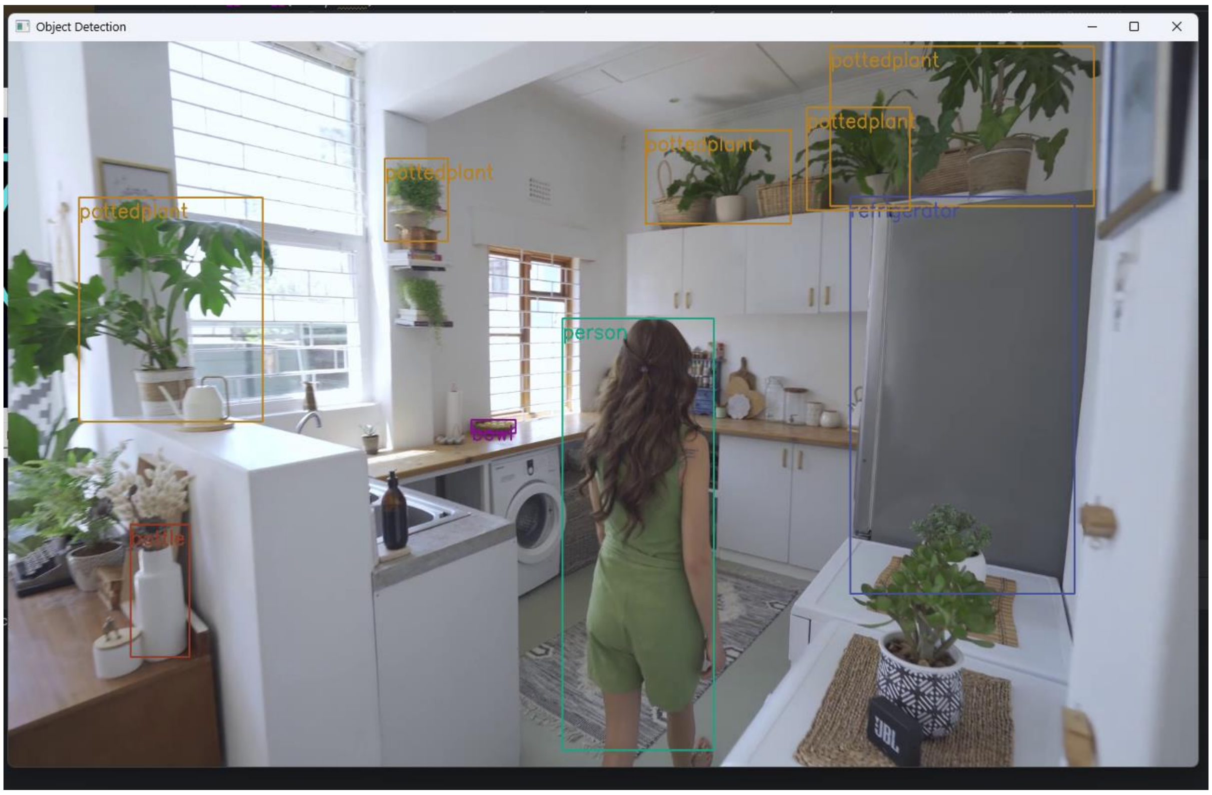


Figure 7.9

OUTPUT-2

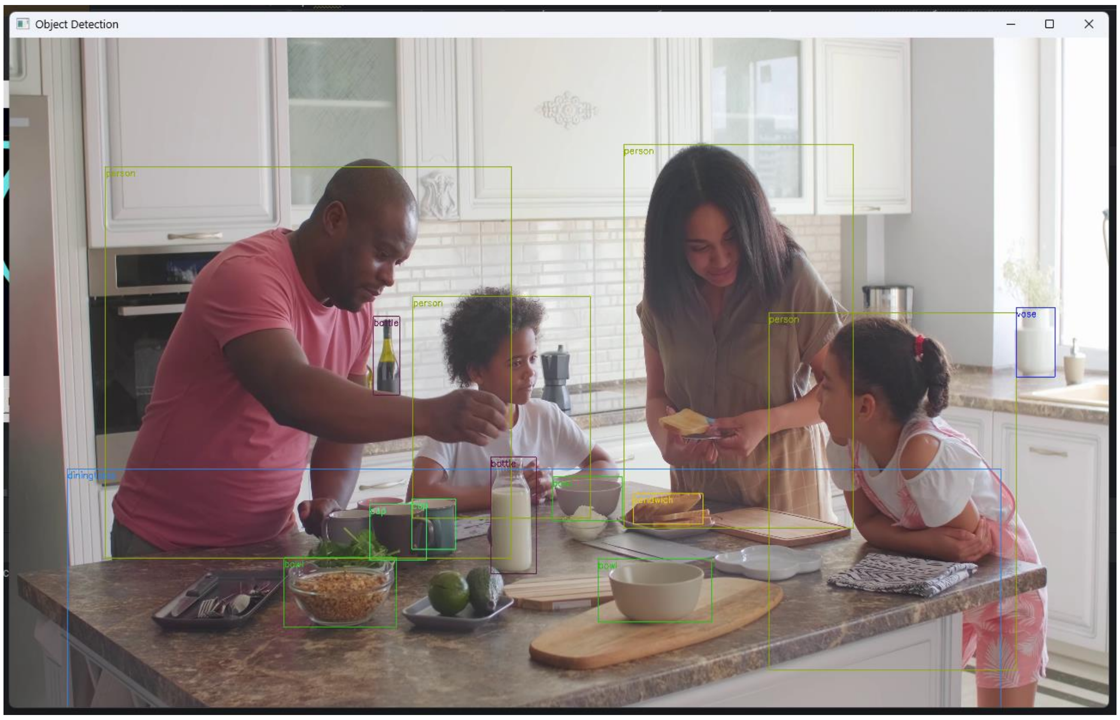


Figure 7.10

OUTPUT-3

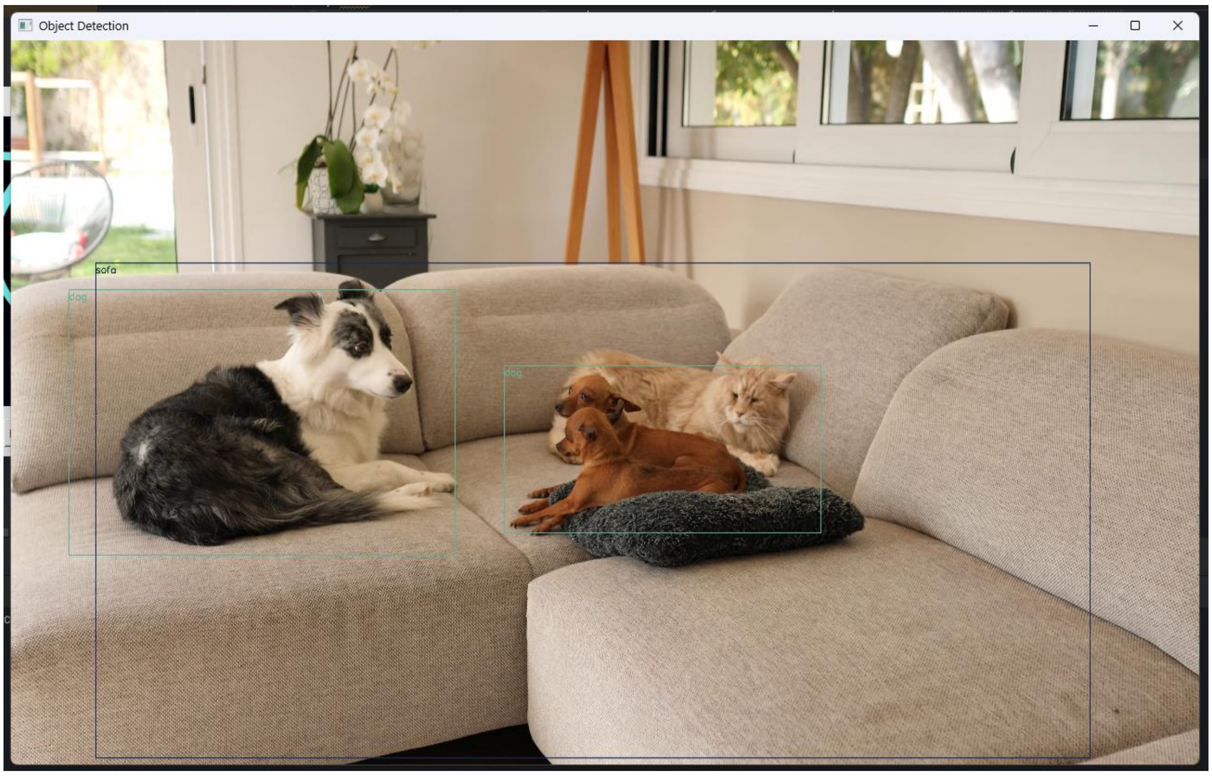


Figure 7.11

7.4 Object Detection in Webcam Output

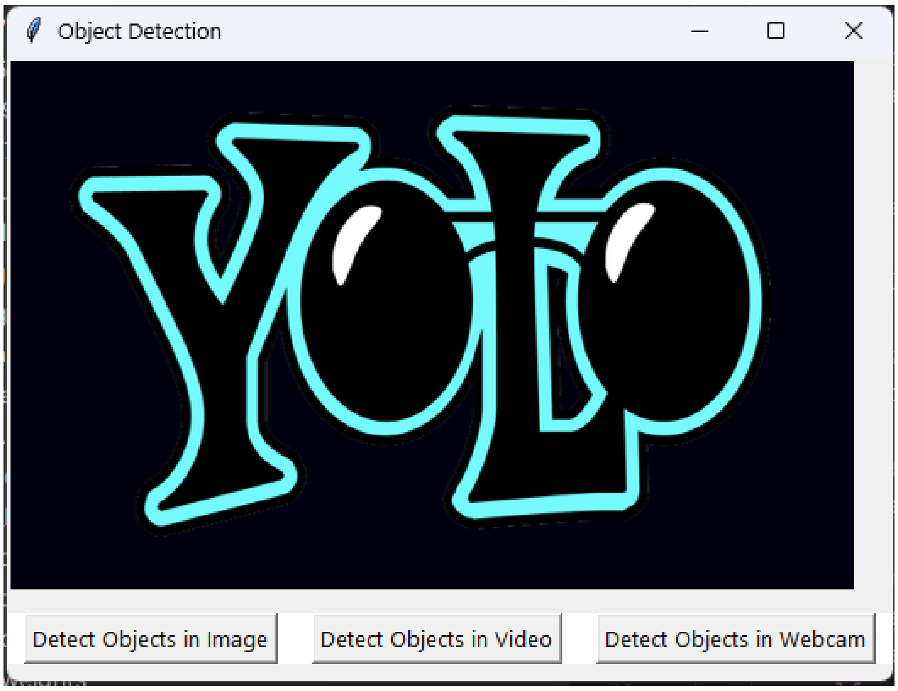


Figure 7.12

OUTPUT-

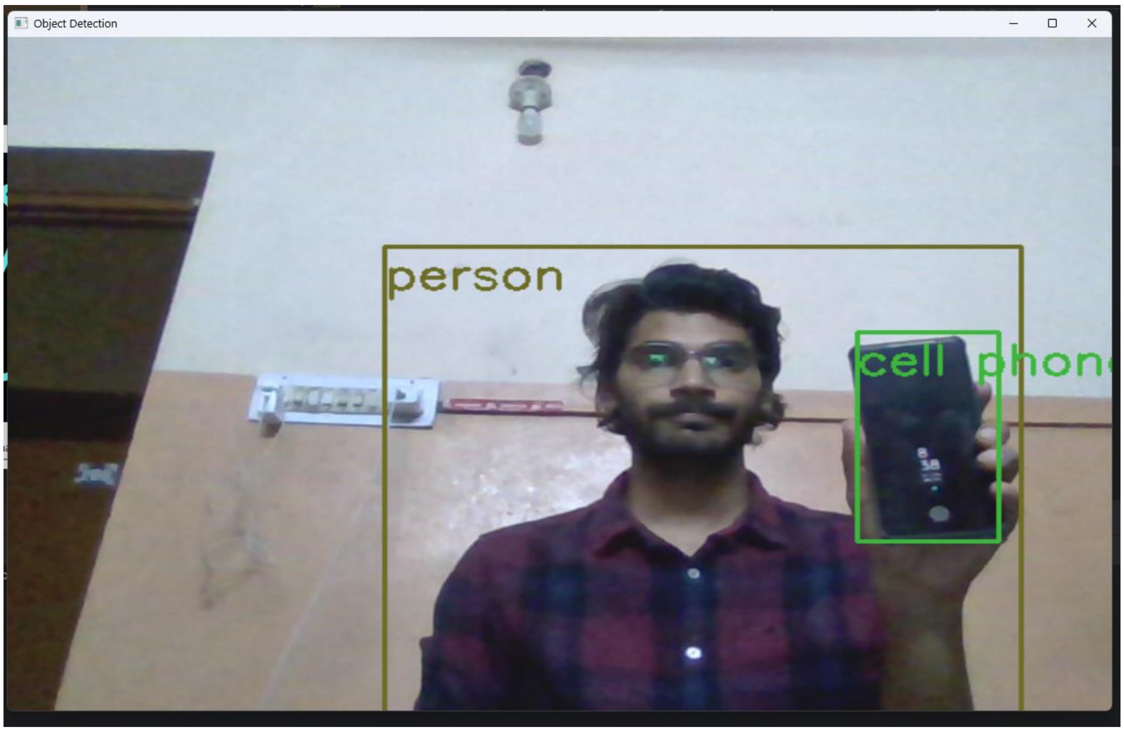


Figure 7.13

CHAPTER 8: SCOPES OF IMPROVEMENT AND CONCLUSION

8.1 Scopes of Improvement

This Object Identification system is identifying objects in images , videos and in webcam quite good but it is not able to identify all the objects in a image or video or in the webcam.

This is because the pre-trained dataset and weights we have used doesn't contain all the elements. In the future we can use a much more larger dataset to identify objects more accurately.

Another scope of improvement can be the smoothness of video and webcam output.

8.2 Conclusion

By and large utilizing this proposition and based on the comes about of the try, we can assist examine the require and distinguish the protest exclusively by its position on the x, y hub in the picture. This paper moreover presents test comes about of different strategies for question discovery. and analyzes and compares each strategy concurring to their performance.We too included a Graphical Client Interface to make it simple to utilize so that everybody can utilize this framework without any programming knowledge.

BIBLIOGRAPHY

- Wikipedia
- https://www.mygreatlearning.com/blog/yolo-object-detection-using-opencv/
- https://git-scm.com/downloads
- https://github.com/pjreddie/darknet
- https://pjreddie.com/media/files/yolov3.weights
- https://www.researchgate.net/publication/340145010_Object_Detection_Algorithm_Based_on_Improved_YOLOv3
- https://towardsdatascience.com/object-detection-using-yolov3-and-opencv-19ee0792a420
- https://www.mygreatlearning.com/blog/yolo-object-detection-using-opencv/

Reference Books

- OpenCV with Python By Example by Prateek Joshi