REAL TIME OBJECT DETECTION

Mini Project Report

Submitted by

Harshika T

(FIT20MCA-2057)

Submitted in partial fulfillment of the requirements for the award of the degree of

Master of Computer Applications
Of
A P J Abdul Kalam Technological University

FEDERAL INSTITUTE OF SCIENCE AND TECHNOLOGY (FISAT)®
ANGAMALY-683577, ERNAKULAM(DIST)
MARCH 2022

DECLARATION

I, **Harshika T** hereby declare that the report of this project work, submitted to the Department of Computer Applications, Federal Institute of Science and Technology (**FISAT**), Angamaly in partial fulfillment of the award of the degree of Master of Computer Application is an authentic record of my original work.

The report has not been submitted for the award of any degree of this university or any other university.

Date : **Harshika T**

Place: Angamaly

FEDERAL INSTITUTE OF SCIENCE AND TECHNOLOGY (FISAT)®

ANGAMALY, ERNAKULAM-683577

DEPARTMENT OF COMPUTER APPLICATIONS

CERTIFICATE

This is to certify that the project report titled **"Real time object detection"** submitted by **Harshika T(FIT20MCA-2057)** towards partial fulfillment of the requirements for the award of the degree of Master of Computer Applications is a record of bonafide work carried out by her during the year 2022.

Project Guide **Head of the Department**

ACKNOWLEDGEMENT

I am deeply grateful to **Dr.Manoj George** , Principal, FISAT, Angamaly for providing me with adequate facilities, way and means by which i was able to complete my mini project work and I express my sincere thanks to **Dr. C. Sheela** ,Vice Principal FISAT, Angamaly.

My sincere thanks to **Dr.Deepa Mary Mathwes**, Head of the department of MCA, FISAT, who had been a source of inspiration. I express heartiest thanks to **Ms.Senu Abi** my project guide for their encouragement and valuable suggestion . I express my heartiest gratitude to my scrum master

Ms. Manju Joy and the faculty members in my department for their constant encouragement and never ending support throughout the project.i would also like to express my sincere gratitude to the lab faculty members for their guidance

Finally I express my thanks to all my friends who gave me wealth of suggestion for successful completion of this project.

ABSTRACT

Real-time object detection is a deep learning project to identify the real object in an image,Video, etc. It is a task of doing object detection in real-time with fast inference while maintaining a base level of accuracy. In this project, we detect each and every object also classify them into groups. The object is detected using a bounding box shown with accuracy and labeled by name. This is a python based program. When humans look at images or video, we can recognize and locate objects of interest within a mann er of moments. The goal of object detection is to replicate this intelligence using a computer. When we look at an image or video the project help to recognize and locate the object. The application of this project is people counting, self-driving cars, agriculture, the health sector, etc. This project is developed using python and Tensorflow along with other libraries.

This is an image-based object detection model. Object detection is a computer vision technique for locating instances of objects in images or videos. When humans look at images or video, we can recognize and locate objects of interest within a matter of moments. The goal of object detection is to replicate this intelligence using a computer. Our model is effective in detecting object from live camera data

Contents

Chapter 1

Introduction

To gain a complete image understanding, we should not only concentrate on classifying different images, but also try to precisely estimate the concepts and locations of objects contained in each image. The motive of object detection is to recognize and locate all known objects in a scene. The searching or recognition process in real-time scenario is very difficult. So far, no effective solution has been found for this problem. Object detection is relatively simpler if the machine is looking for detecting one particular object.

This is the process of finding and recognizing real-world object instances such as cars, bikes, TV, flowers, and humans out of images or videos. An object detection technique lets you understand the details of an image or a video as it allows for the recognition, localization, and detection of multiple objects within an image. Object detection in a video stream can be done by processes like pre-processing, segmentation, foreground and background extraction, feature extraction. Humans can easily detect and identify objects present in an image. The human visual system is fast and accurate and can perform complex tasks like identifying multiple objects with the availability of large amounts of data, faster GPUs, and better algorithms, we can now easily train computers to detect and classify multiple objects within an image with high accuracy.

Deep learning is subset of machine learning, which is essentially a neural net-

work with three or more layers. These neural networks attempt to stimulate the behavior of human brain. Neural network with a single layer can still make approximate prediction, help to organize and refine for accuracy. It drives many artificial intelligence application and services that improve automation performing analytical and physical task without human intervention. Deep learning technology lies behind everyday product and services.

The main purpose of object detection is to identify and locate one or more effective targets from still images or real video data. It comprehensively includes a variety of important techniques, such as image processing, pattern recognition artificial intelligence, and machine learning. Real-time object detection is the task of doing object detection in real-time with faster interference while maintaining a base level of accuracy. The goal of objection detection is to identify a certain defined object or thing in the live feed in order to search, locate, or count specific items desired as per the goal.

Chapter 2

PROOF OF CONCEPT

This project is designed to detect the real time objects from a image or video. Object detection is a computer vision technique in which it can detect, locate, and trace the object from a given image or video using the bounding box. The real-world applications are Logo detection, medical image analysis, autonomous driving, video surveillance, or robot vision. The motive of object detection is to detect and locate all known objects in a scene.

It is aiming to make the machine as fast and accurate as a human with the availability of large amounts of data, faster GPUs, and better algorithms.The project helps us to detect the object from images and real-time videos fastly and efficiently.Object detection is a computer vision technique in which it can detect, locate, and trace the object from a given image or video using the bounding box.The real-world applications are Logo detection, medical image analysis, autonomous driving,video surveillance,or robot vision.It is easy for training computers to detect and classify multiple objects within an image with high accuracy.Object detection helps us to understand and analyze the scenes in videos and images.This project enables us to achieve greater accuracy in tasks.Applications using the approach often require less expert analysis and fine-tuning.Deep learning also provides superior flexibility.This system increases accuracy and reduces complexity.

2.1 Objectives

The main objectives of real time object detection

- To determine the objects present in the images efficiently and effectively by using a convolutional neuron network.
- To find multiple different objects easily by bounding boxes with labeling their names and reduce input parameter then gives accurate result.
- To process and recognize the objects in images or videos.

Chapter 3

IMPLEMENTATION

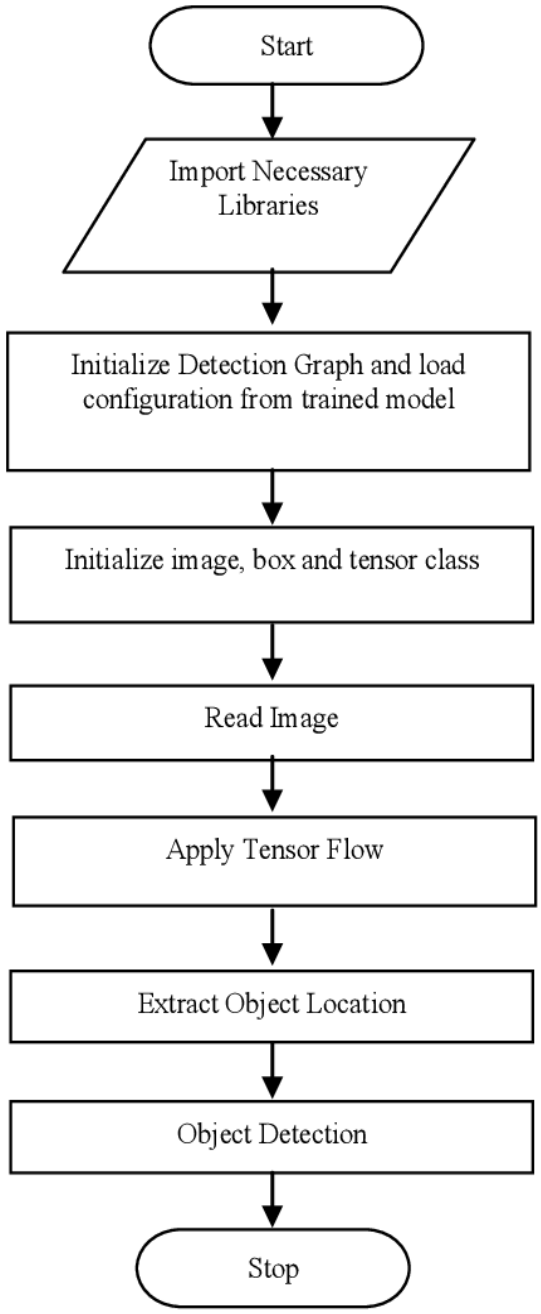
Real time object detection is a deep learning project. Object recognition is a general term to describe a collection of related computer vision tasks that involve identifying objects. Image classification involves predicting the class of one in an image. Object localization refers to identifying the location of one or more objects in an image and drawing a bounding box around their extent. Object detection combines these two tasks and localizes and classifies one or more objects in an image.This is a deep learning project which is python based. The frontend of the project is python and backend I have used many libraries. The IDE used is anaconda with jupyter notebook with TensorFlow other libraries used are NumPy, Matplotlib, OpenCV.

Object detection is used to detect, locate, and trace the object for a given image, we can also detect in real time. But the main question is how we can do this? So to implement this we will be using the Object Detection API provided by Tensorflow. The main purpose of this is that it identifies the class of objects (person, table, chair, etc.) and their specific coordinates in the given image, and recognizes it. Then this specific coordinates are traced out by drawing a box around those specific objects, and it depends on our model how accurately it locates the position of these objects.So the ability to locate the object inside an image defines the

performance of the algorithm used for detection.The object detection algorithms may be pre-trained or you can train it from scratch. In this we will be using pre-trained models and changing them as per our requirements.

3.1 System Architecture

The region proposal network starts with the input image being fed into the backbone CNN. For every point in the output feature map, the network has to learn whether an object is present in the input image at its corresponding location and estimate its size. This is done by placing a set of "Anchors" on the input image for each location on the output feature map from the backbone network. As the network moves through each pixel in the output feature map, it checks whether the corresponding anchors spanning the input image actually contain objects, and refine these anchors' coordinates to give bounding boxes as "Object proposals" or regions of interest this output is used to give probabilities of whether or not each point in the backbone feature map. The architecture diagram for the whole application is given in figure 3.1
Object detection generally is categorized into 2 stages:
Single-stage object detectors. Two-stage object detectors. two-stage detectors divide the object detection task into two stages: extract RoIs (Region of interest), then classify and regress the RoIs. Examples of object detection architectures that are 2 stage oriented include R-CNN, Fast-RCNN, Faster-RCNN, Mask-RCNN and others. Let's take a look at the Mask R-CNN for instance.And this which i am using in this model.



2. TensorFlow Based Object detection flowchart

Figure 3.1: Architecture diagram

3.2 Dataset

The data requirements is very high for the project. The data set here used are COCO dataset. Which is mostly used for the object detection .COCO is large-scale object detection, segmentation, and captioning dataset. COCO dataset, is high-quality datasets for computer vision, mostly state-of-the-art neural networks. The dataset consists of 328K images.It is the labeled data.

3.3 Algorithm

- CNN:

  In deep learning, a convolutional neural network (CNN/ConvNet) is a class of deep neural networks, most commonly applied to analyze visual imagery. Now when we think of a neural network we think about matrix multiplications but that is not the case with ConvNet. It uses a special technique called Convolution.CNNs are used for image classification and recognition because of its high accuracy. It was proposed by computer scientist Yann LeCun in the late 90s, when he was inspired from the human visual perception of recognizing things.

  The big idea behind CNNs is that a local understanding of an image is good enough. The practical benefit is that having fewer parameters greatly improves the time it takes to learn as well as reduces the amount of data required to train the model.

- Faster RCNN:

  Faster RCNN is an object detection architecture presented by Ross Girshick, Shaoqing Ren, Kaiming He and Jian Sun in 2015, and is one of the famous object detection architectures that uses convolution neural networks.The Faster R-CNN has the same number of hidden layers as the

Fast R-CNN, the RPN has no hidden layers and is only used as a feature extractor. The Fast R-CNN has three fully connected layers.

Faster R-CNN is an object detection model that improves on Fast R-CNN by utilising a region proposal network (RPN) with the CNN model. The RPN shares full-image convolutional features with the detection network, enabling nearly cost-free region proposals. It is a fully convolutional network that simultaneously predicts object bounds and objectness scores at each position. The RPN is trained end-to-end to generate high-quality region proposals, which are used by Fast R-CNN for detection. RPN and Fast R-CNN are merged into a single network by sharing their convolutional features: the RPN component tells the unified network where to look.

3.4 Modules

There are 3 modules in this model

- Detect object in the frame
- classification of the objects
- Detecting objects with its accuracy.

3.4.1 Detect object in the frame

The project is using CNN, Algorithm is used to generate a large set of bounding boxes spanning the full image. Faster R-CNN used for detecting from the videos.This detect the live object from the frame accurately.By classifing the objects.

Algorithm

Algorithm 1: image processing
Input: Image
Output: Image with labeled name
Step 1: Start
Step 2: Get the image needed to classify
Step 3: Perform the operation and extract features from the image.
Step 4: Get the possible object from the input
Step 5: Display the result
Step 6: Stop.

flowchart

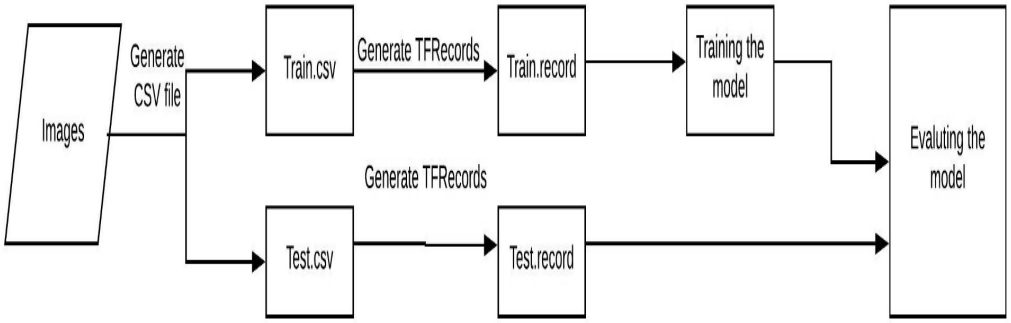


Figure 3.2: test/train flow

3.4.2 Classification of the objects

Visual features are extracted for each boxes. They are evaluated and it is determined whether and which objects are present in the boxes based on visual features.

Algorithm

Algorithm 2: object pre processing
Input: live video stream
Output : processed frames .
Step 1: Start
Step 2: Get video stream
Step 3: convert video into frames
Step 4: Resize the frame apply to convert it into object and classify them.
Step 5: Return processed frame.

flowchart

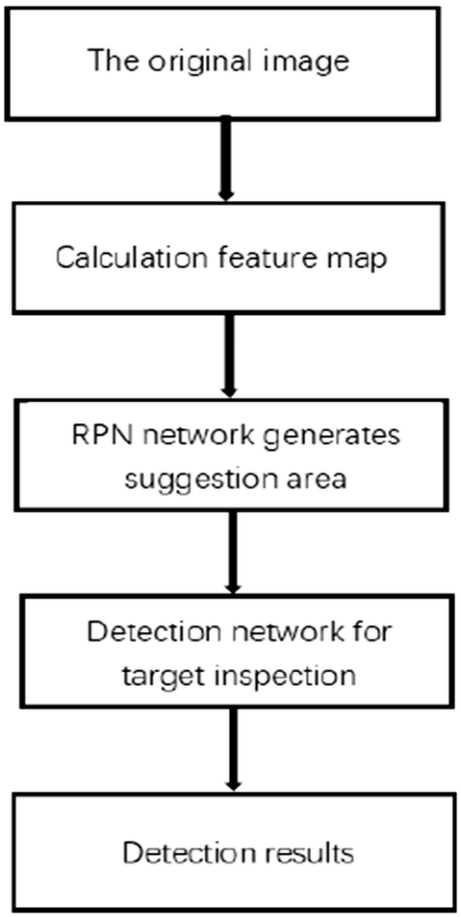


Figure 3.3 Diagram for RCNN

3.4.3 Detecting object with accuracy

In the final post-processing step overlapping boxes are combined into a Single bounding box.

working

Every object class has its own special features that helps in classifying the class. Object class detection uses these special features. For this project we chose the deep learning approach as we can get an end-to-end object detection without specifically defining the features. There are several ways for implementing object detection using the Deep Learning approach some of them are using Single Shot MultiBox Detection (SSD), You Only Look Once (YOLO),and Faster Region Proposal Convolutional Neural Network. For this, I used faster – RCNN for object detection because it seemed to be more efficient in our model Training / Testing.The working of this is while the webcam is on it will detect the objects with the help of CNN, faster R-CNN algorithm. The CNN algorithm extracts the features and detects the objects. The faster R-CNN helps in the selection of a specific region, it helps forgetting objects from the real-time video.

3.5 Issues Faced and Remedies Taken

3.5.1 Issues

- Slow detection while using Flask framework.
- Required new classes.
- High computation power needed.

3.5.2 Remedies

Instead of Flask framework I have used TKinter to create GUI.It is easy and convenient to use.Check the accuracy of algorithms.

Chapter 4

RESULT ANALYSIS

The project was completed successfully in time.The result of the proposed project Real time object detection with deep learning lies in developing a model for detecting an object from the image or video effectively.This proposed system help in searching for the object in the frame easily with a labeled bounding box.

Chapter 5

CONCLUSION AND FUTURE SCOPE

5.1 Conclusion

Deep-learning based object detection has been a search hotspot in recent years. This project starts on generic object detection pipelines which give base architectures for other related tasks.Object detection with deep learning and OpenCV and Efficient, threaded video streams with OpenCV. The camera sensor noise and lightening condition can change the result because it can create problem in recognizing the objects. generally, this whole process requires GPU.

Object detection is a key ability for most computer and robot vision system. It should be noted that object detection has not been used much in many areas where it could be of great help. As mobile robots, and in general autonomous machines, are starting to be more widely deployed, the need for object detection systems is gaining more importance. Finally, we need to consider that we will need object detection systems for nano-robots or for robots that will explore areas that have not been seen by humans, such as depth parts of the sea or other planets, and the detection systems will have to learn to new object classes as they

are encountered. In such cases, a real-time open-world learning ability will be critical.

The goal of this project is to develop a user-friendly way for object detection. That will make us easy to detect objects from a image or real video. The scope of the project lies high. The use of object detection is more useful in many fields today. It is more useful to identify objects inthe image, face detection, automatic car, etc.

5.2 Future Scope

The project has got a lot of future scopes. self-driving cars, Tracking objects, face recognition,medical imaging, object counting, object extraction from an image or video, person detection.We can add noise detection also.

.

Chapter 6

CODING

6.1 ui.py

This is the GUI code for my project.It have used Tkinter and has 2 button for selecting image or video.

```
import sys
import os
from tkinter import *
from tkinter import filedialog
window = Tk()
window.title('Object Detection')
window.geometry('600x500')

def loadfilename():
filetype = (('Image Files', '*.jpeg;*.jpg;*.png'), ('All files', '*.*'))
filename = filedialog.askopenfilename(filetypes=filetype)
script1(filename)
```

def script1(filename):
os.system('python detect$_{i}mage.py' + filename)

$defscript2() :$
$os.system('pythondetect_{v}ideo.py')$

$btn = Button(window,text = "DetectFromImage", bg = "black", fg =$
"white",
$command = loadfilename)$

$btn.grid(column = 0, row = 0, padx = 100, pady = 200)$

$btn = Button(window,text = "DetectFromVideo",$
$bg = "black", fg = "white", command = script2)$

$btn.grid(column = 4, row = 0, padx = 100, pady = 200)$
$btn.grid(column = 0, row = 0, padx = 100, pady = 200)$

$btn = Button(window,text = "DetectFromVideo",$
$bg = "black", fg = "white", command = script2)$

$btn.grid(column = 4, row = 0, padx = 100, pady = 200)$

$window.mainloop()$
$defscript3() :$

$os.system('pythondetect_{v}ideo.py')$

$btn.grid(column = 4, row = 0, padx = 100, pady = 200)$

$defscript4() :$
$os.system('pythondetect_video.py')$

$btn = Button(window, text = "DetectFromImage", bg = "black", fg =$
$"white",$
$command = loadfilename)$

$window.mainloop()$

6.2 detectimage.py

```
import os
import cv2
import matplotlib.pyplot as plt
import matplotlib
from PIL import Image
import numpy as np
import tensorflow as tf
import pathlib
import time

from object_detection.utils import label_map_util
from object_detection.utils import visualization_utils as viz_utils
import warnings
matplotlib.use('TkAgg')
gpus = tf.config.experimental.list_physical_devices('GPU')

for gpu in gpus:

tf.config.experimental.set_memory_growth(gpu, True)

PATH_TO_LABELS = "mscoco_label_map.pbtxt"

PATH_TO_SAVED_MODEL =
"centernet_resnet50_v1_fpn_512x512_coco17_tpu-8_model"

detect_fn = tf.saved_model.load(PATH_TO_SAVED_MODEL)

category_index =
```

$$label_map_util.create_category_index_from_labelmap(PATH_TO_LABELS,$$
$$use_display_name = True)$$

$$image_np = np.array(Image.open(os.sys.argv[1]))$$

$$input_tensor = tf.convert_to_tensor(image_np)$$

$$input_tensor = input_tensor[tf.newaxis, ...]$$

$$detections = detect_fn(input_tensor)$$

$$num_detections = int(detections.pop('num_detections'))$$

$$detections =$$
$$key : value[0, : num_detections].numpy()\ for\ key, value\ in\ detections.items()$$
$$detections['num_detections'] = num_detections$$

$$detections['detection_classes'] = detections['detection_classes'].astype($$
$$np.int64)$$

$$image_np_with_detections = image_np.copy()$$

$$viz_utils.visualize_boxes_and_labels_on_image_array($$
$$image_np_with_detections,$$
$$detections['detection_boxes'],$$
$$detections['detection_classes'],$$
$$detections['detection_scores'],$$
$$category_index,$$
$$use_normalized_coordinates = True,$$
$$max_boxes_to_draw = 200,$$

$min_score_thresh = .30,$
$agnostic_mode = False)$

$detections =$
$key : value[0, : num_detections].numpy() for key, value in detections.items()$
$detections['num_detections'] = num_detections$

$detections['detection_classes'] = detections['detection_classes'].astype($
$np.int64)$
$plt.figure()$

$viz_utils.visualize_boxes_and_labels_on_image_array($
$image_np_with_detections,$
$detections['detection_boxes'],$
$detections['detection_classes'],$
$detections['detection_scores'],$
$category_index,$
$plt.imshow(image_np_with_detections)$

$plt.show()$

$if cv2.waitKey(25)0xFF == ord('q') :$

$exit()$

6.3 detectvideo.py

```
from PIL import Image
import matplotlib.pyplot as plt
import tensorflow as tf
import numpy as np
import os
import cv2

from object_detection.utils import label_map_util
from object_detection.utils import config_util
from object_detection.utils import visualization_utils as viz_utils
from object_detection.builders import model_builder

center_net_path = './centernet_resnet50_v1_fpn_512x512_coco17_tpu-8/'
pipeline_config = center_net_path + 'pipeline.config'
model_path = center_net_path + 'checkpoint/'
label_map_path = './mscoco_label_map.pbtxt'

cap = cv2.VideoCapture(0)

configs = config_util.get_configs_from_pipeline_file(pipeline_config)
model_config = configs['model']
detection_model = model_builder.build(
model_config = model_config, is_training = False)

ckpt = tf.compat.v2.train.Checkpoint(model = detection_model)
ckpt.restore(os.path.join(model_path, 'ckpt-0')).expect_partial()
```

$def get_model_detection_function(model):$
$@tf.function$
$def detect_fn(image):$
$image, shapes = model.preprocess(image)$
$prediction_dict = model.predict(image, shapes)$
$detections = model.postprocess(prediction_dict, shapes)$

$return detections, prediction_dict, tf.reshape(shapes, [-1])$
$return detect_fn$

$detect_fn = get_model_detection_function(detection_model)$

$def get_model_detection_function(model):$
$@tf.function$
$def detect_fn(image):$
$image, shapes = model.preprocess(image)$
$prediction_dict = model.predict(image, shapes)$
$detections = model.postprocess(prediction_dict, shapes)$

$label_map_path = label_map_path$
$label_map = label_map_util.load_labelmap(label_map_path)$
$categories = label_map_util.convert_label_map_to_categories($
$label_map,$
$max_num_classes = label_map_util.get_max_label_map_index(label_map),$
$use_display_name = True)$
$category_index = label_map_util.create_category_index(categories)$

$while1:$
$ret, image_np = cap.read()$
$image_np_expanded = np.expand_dims(image_np, axis = 0)$
$input_tensor = tf.convert_to_tensor($
$np.expand_dims(image_np, 0), dtype = tf.float32)$
$detections, predictions_dict, shapes =$
$detect_fn(input_tensor)$
$label_id_offset = 1$
$image_np_with_detections = image_np.copy()$

cv2.imshow('object detection', cv2.resize(
$\text{image}_np_with_detections, (800, 600)))$

$if cv2.waitKey(25)0xFF == ord('q'):$

$break$

$cap.release()$

$cv2.destroyAllWindows()$

Chapter 7

SCREEN SHOTS

Figure 7.1: UI Screen

Figure 7.2: image detection

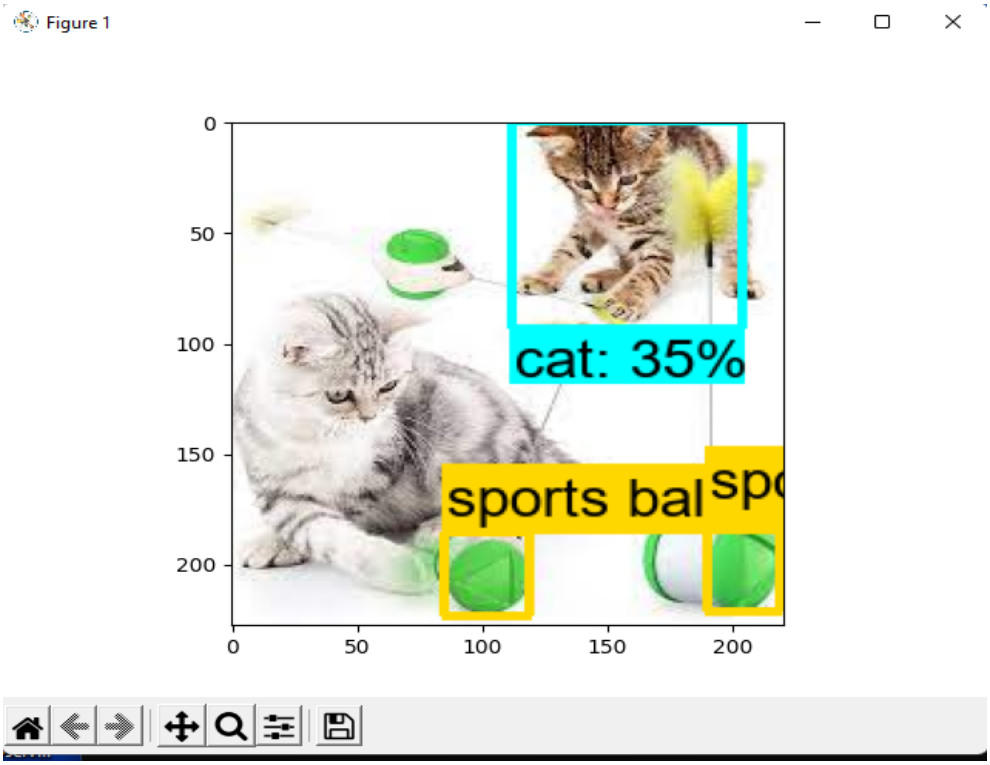


Figure 1
cat: 35%
sports bal
0
50
100
150
200

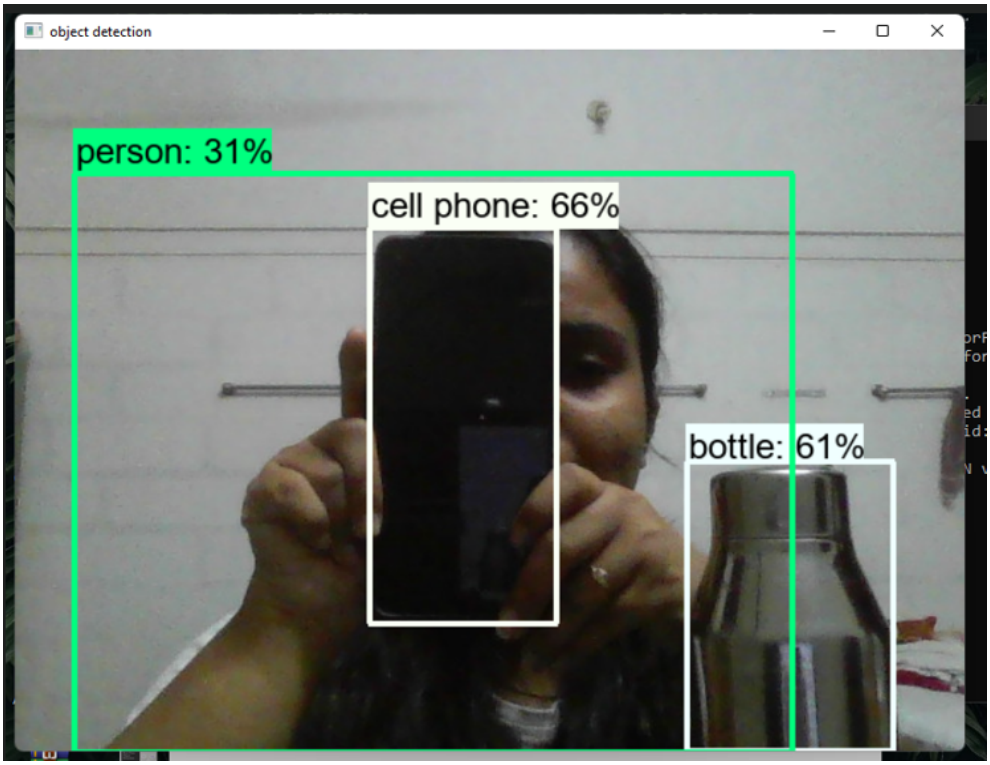


Figure 7.3: video detection

Chapter 8

REFERENCES

1. https://ieeexplore.ieee.org
2. Abdul Vahab, Maruti S Naik, Prasanna G Raikar an Prasad S R4, “Applications of Object Detection System”, International Research Journal of Engineering and Technology (IRJET)
3. Zhong-Qiu Zhao,Peng Zheng, Shou-tao Xu, and Xindong Wu-Object Detection with Deep Learning
4. Priyal Jawale, Hitiksha Patel Chaudhary2, Nivedita Rajput3. Real-Time Object Detection using TensorFlow (IRJET)
5. 1Rinkesh U Patel, 2Meet S Patel, 3Dev A Thakkar, 4Bhumika Bhatt Real-Time Object Detection using TensorFlow (IJCRT)