**CS225-226 Project:-**

**Topic:**

***“Object Detection Model” using Tensorflow***

# Motivation:-

When it comes to identifying images, we humans can clearly recognize and distinguish different features of objects. This is because our brains have been trained unconsciously with the same set of images that has resulted in the development of capabilities to differentiate between things effortlessly. Contrary to human brains, computer views visuals as an array of numerical values and looks for patterns in the digital image, be it a still, video, graphic, or even live, to recognize and distinguish key features of the image. Training computers to detect objects is helpful in various areas including security, as well as in automation. This project is a demonstration of such a model.

# Overview:-

The complete code is written in python language and uses various python libraries including OpenCV. It also makes the use of TensorFlow API.

The whole code is distributed in two Jupyter notebooks :

The first Jupyter notebook (image\_collection.ipynb) contains the code for collection of images.

Graphical user interface, text, application, email

Description automatically generated

The second notebook (training\_and\_detection.ipynb) contains the code for training our model, as well as to detect objects through our v380 Wi-Fi camera.

Text

Description automatically generated

For this project, I have created a separate virtual environment in the system (‘spro’) to install all the packages and libraries that we need, and then import all the header files according to their need.

A screenshot of a computer

Description automatically generated with medium confidence



## Components and Supplies:-

* v380 Wi-fi Camera:- A v380 Wi-fi camera is an IP camera that can be connected to any mobile, or a computer system over the same Wi-fi network. It is used in this project to give us the live-feed of the object which we have to detect.

A picture containing text, wall, indoor, green

Description automatically generated

* A to mini B USB data cable:-Used to connect v380 Wi-Fi camera to a power source.



* Apps and Online Services:-

Logo, company name

Description automatically generated

The Jupyter Notebook is the original web application for creating and sharing computational documents. It offers a simple, streamlined, document-centric experience.

## Making the Model:-

The execution of the code is as follows:

1. Create a virtual environment ‘spro’ to install all the libraries and packages.

2. Importing various libraries as given :

i) opencv - reading our sample images,

ii) uuid - for giving unique number to our image file name,

iii) os - to manipulate the storage of images in the system,

iv) time - to introduce delay in capturing images are included.

3. The dataset that is to be fed to our model is loaded. This can be done in two ways :

i) import a standard dataset

ii) capture images using our Wi-fi camera.

4. Images are labelled. This generated .xml extension files which can be fed to our model directly.

5. The dataset is then classified into a training dataset and a testing dataset.

6. Various directories are created in order to load our model, which is a pre-trained model from TensorFlow Zoo.

7. Installation of the model from TensorFlow Zoo and verification of the installation.

8. The dataset is then converted into a format called TFRecords which is the format that is recognised by the model.

9. The model is then configured according to our dataset, such as specifying appropriate directories to the model.

10. Training and testing of the model is done.

11. Finally, the model is ready for object detection.

12. The object detection is done in two ways :

i. Detecting from an image :

A picture containing text, person

Description automatically generated

ii. Detecting from real-time video feed :

Graphical user interface

Description automatically generated

## Code:-

The code snippet is given below:-

Graphical user interface, text, application, email

Description automatically generatedGraphical user interface, text, application

Description automatically generatedText

Description automatically generatedGraphical user interface, text, application, email

Description automatically generatedText, application

Description automatically generatedGraphical user interface, text, application, email

Description automatically generatedGraphical user interface, text, application, email

Description automatically generatedGraphical user interface, text, application, email

Description automatically generatedText

Description automatically generatedText

Description automatically generated

## Challenges faced (Lessons Learnt):-

During the installation of the model, several libraries were missing. Thus, I was forced to download the missing libraries and then configure them into my model. Also, sometimes the model gives wrong output such as detecting a wrong object, or detecting an object even if it is not present. I trained the model by giving it more dataset but a huge amount of dataset is required for the model training. glow.

## Alternatives:-

1. From the TensorFlow Zoo, more efficient models can be used which will give us correct results.

2. More dataset can be fed to our model. This can be done using a standard dataset.

3. The number of iterations for training our model in this case was 2000. It can be increased to say 10000 to improve the efficiency of our model.

THANK YOU

# SAURAV DUDHATE

# B.Tech 2nd Year

# 2001CS62