**A**

**Project Report**

**On**

**“Human Detection & Counting”**

**For the Course**

**Machine Learning & Neural Networks(CS401)**

**SUBMITTED BY**

**SAHIL SHASHIN SHAH - 21251000**

**PROFESSOR: MR. BARAK PEARLMUTTER**

**ABSTRACT:**

* The objective of this project is to detect and count peoples from images and video for safety purpose.
* Counting people in visual surveillance is a difficult task.
* So in this project we are using OpenCV and HOG algorithm.

**INTRODUCTION:**

* In this project, we are trying to detect and count the number of humans in a video or in a photograph in a particular frame.
* In this we are using haarcascade\_frontalface\_default.xml to detect the number of faces in a frame.
* OpenCV which is a strong machine learning library is used, in this project.
* For feature descriptor in computer vision and image processing we are using HOG(Histogram Oriented Gradient)
* We are also using “haarcascade\_frontal\_face.xml” file, for object detection.

**PREREQUISITES**

* **Library Used:**

Cv2: This is used for computer vision.

Imutils: This library is used for processing image.

Numpy: Used to store image in numpy array.

Argparse: Used as input for command line.

* **Download**: haarcascade\_frontal\_face.xml file.

**DESCRIPTION:**

* Imported libraries which are mentioned above.
* Step 1 Detect() Method:

1. First we create a “Detect()” method, in this method were all the process is taking place. As we know that video is a frame of images, so what we tried to do is we will detect the person in the frame and count it one after another.
2. So in this method we are using haarcascade\_frontal\_face.xml file, to identify faces from the image and video, because it uses object algorithm for detecting features of edge and line.
3. Once the person is detected will make a box around that person and show it as a frame and return the frame with bounded green box around that particular person.
4. For creating a box it is done detectmuliscale() method. List containing of bounding box they are in the form of “x,y,w,h”
5. Starting coordinates of the box are x, y and w,h is used for height and width of the box.
6. So our detect() method is created.

* Step 2 HumanDetector() Method:

1. In this method first we have to give the path of the image and video.
2. Once the path is given then the method will go to that directory and perform the task.
3. We are using video.read() to read each frame one by one.
4. For each frame we will call detect() method.

* Step 3 DetectByPathVideo() Mehtod:

1. This method is more or less similar to the above method only change in this method is we are trying to give path to the video, for that we have to check whether the path provided has video or not.
2. After that we will just check that the frame has been successfully read or not. If there is not frame then we will end the loop.

* Step 4 DetectBypathImage():

1. This method is used to detect human beings from the image.
2. For this we are reading the image first using cv2 library.
3. Then to process the given image we are using imutils and then detect method is used for detecting.

* Step 5 Argparse() Mehtod:

1. This function parses and returns the arguments supplied to the script through the terminal as a directory.
2. In the parse we are passing 2 arguments for image and video.

* Step 6 Main function:

1. Here we are declaring the model in main function.
2. We are using HOGDescriptor with SVM implementing in OpenCV.
3. **Using cv2.HOGDescriptor\_getDefaultPeopleDetector()** it calls the pre-trained model for detecting human using OpenCV and then we are trying to feed the support vector machine with it.

* Step 7 Run the python file:

1. If we are dealing with video file. Run the below command in terminal

python filename.py –v “Path of video”

1. If we are dealing with image file. Run the below command in terminal

python filename.py –v “Path of image”

**CONCLUSION:**

* In this project we were successfully able to detect human beings in video and image, this kind of project can be very helpful during this COVID-19 pandemic as we can easily monitor the crowded place.