INTERNATIONAL JOURNAL OF ADVANCE RESEARCH, IDEAS AND INNOVATIONS IN TECHNOLOGY

ISSN: 2454-132X
Impact Factor: 6.078
(Volume 7, Issue 3 - V7I3-1976)
Available online at: https://www.ijariit.com

Social distance monitoring using HOG and DNN

N. Sai Karthik
karthikn9246@gmail.com
SRM University, chennai

G.Dinesh kumar
dineshkumar7997@gmail.com
SRM University, chennai

P.Ponnammal
ponnammp@srmist.edu.in
SRM University, chennai

ABSTRACT

As of now, the world is facing its greatest pandemic COVID-19. Social distancing has become a part of our lives more than ever during the recent times .From our project, we would like to analyze a video which displays people walking around and provide the output in the same frame whether or not the people shown in the video are maintaining a social distance between them. The code will then be generalized in order to take any sample video as input and to provide the output. This will potentially help authorities to monitor public areas for practicing social distance, which will eventually help to curb the spread of a contagious disease.

Keywords: *HOG, DNN*

Tools and Packages

- Language – Python
- Packages –

OpenCV
TensorFlow
Numpy
Imutils
Object_detection

Two methodologies included in this project to detect the social distance violation

1. HOG – Histogram of Oriented Gradients
2. DNN – Deep Convolution Neural Network

1. INTRODUCTION

Coronavirus is known as an ailment because of a novel Covid, otherwise called extreme intense respiratory disorder Covid- 2. It was accounted for on 31 -dec-20l9. WHO reported this flare-up as a worldwide crisis on 30-jan-2020, and later declared as worldwide pandemic on March 1 1 2020.Many countries declared lockdown to protect their citizens and stop spreading the virus. As the nation's economy is falling, many countries are forced to lift lockdown and are re-building their economy to survive. Now, governments across the globe have new challenges to monitoring the people for maintaining social distancing to stop spreading of the virus, which is a tedious one.

2. LITERATURE SURVEY

[1] Histogram of bound Gradient (HOG) along with SVM was utilized as partner prudent procedure for location of item typically and identification of people particularly. Human identification misuse HOG-SVM furnishes us higher grouping yield offset with profound learning ways. Notwithstanding, data conditions and complex number-crunching in HOG include age and SVM arrangement edge the most extreme yield of those applications. During this exploration, we will in general start a totally one of a kind high-throughput equipment plan for location of people by improving HOG attributes and Support Vector Machine grouping. The yield has highlights of a quick, exceptionally improved and limited expense of HOG-age along with a changed information way for side-side activities of SVM and HOG include social control. The arranged plan has been upheld in TSMC 65-nanometer innovation with a most employable recurrence of 500MegaHertz with yield of 139-outlines per-second for Full-High-Definition goal. The equipment space cost is concerning 145kiloGEs along the edge of 242kb SRAMs.

Published in:2019 IEEE Asia Pacific Conference on Circuits and Systems (APCCS)
Date of Conference: 11-14 Nov. 2019
Date Added to IEEE Xplore: 09 January 2020
ISBN Information:
INSPEC Accession Number: 19264145
DOI: 10.1109/APCCAS47518.2019.8953123
Publisher: IEEE
Conference Location: Bangkok, Thailand

[2] Generally, HOG along with SVM classifier uses rectangular pictures for HOG extraction of function descriptors and coaching. this suggests important further work must be done to method digressive pixels happiness to the framework close the thing of interest. whereas some entites might so be sq. or rect. in shape, mostly entites don't seem to be simply expressible by easy geometric shapes. In Bitmap-Histogram Of Gradients method we tend to propose during this research ,the unpredictable form of entity is painted by a image to neglect process of additional framework pixels. Bitmap, is developed from the coaching data-set , encodes those parts of a picture to be want to be developed from a classifier. Final outputs and practical outputs show that not solely the planned formula decreases the work related to HOG along with SVM classifiers by seventy fifth equaled to the progression, however conjointly it shows a median increase concerning five-hitter in recall and a decrease concerning a pair of in exactitude compared with normal HOG.

Published in: 2016 IEEE International Conference on Image Processing (ICIP)
Date of Conference: 25-28 Sept. 2016
Date Added to IEEE Xplore: 19 August 2016
ISBN Information:
Electronic ISSN: 2381-8549
INSPEC Accession Number: 16521645
DOI: 10.1109/ICIP.2016.7532439
Publisher: IEEE
Conference Location: Phoenix, AZ, USA

[3] Image processing and computer vision have gained an enormous advance in the field of machine learning techniques. Some of the major research areas within machine learning are Object detection and Scene Recognition. Though there are numerous existing works related to the specified fields object detection still encounters numerous challenges when it comes to implementing in the real-time scenario. The problem occurs in the detection due to various objects present in the background. Object detection mechanism detects a specified object when a particular scene is given. Classifiers like SVM and Neural Networks are used to train the classifier in such a way they are able to detect an object when a new image is given. In this paper, we have proposed a model which detects texts from an image. Bounding boxes are used to detect the texts and localize it.
The neural network is used to train the model where numerous images having texts are given as the training set. The performance evaluation is done on the model and it is observed that it detects the texts when a new image is given. Object detection is a fundamental problem in computer vision, which aims to detect general objects in images.

Published in: 2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI)
Date of Conference: 23-25 April 2019
Date Added to IEEE Xplore: 10 October 2019
ISBN Information:
INSPEC Accession Number: 19046613
DOI: 10.1109/ICOEI.2019.8862698
Publisher: IEEE
Conference Location: Tirunelveli, India

[4] Object detection is widely used in a variety of fields, including smart security surveillance, smart cities, power inspection, and many more. The thing detection formula with deep learning support might be a fairly storage and computing intensive formula, which would be difficult to implement on an embedded platform with limited storage resources and computing resources. In addition, we prefer to compress the network model using a model compression technique that supports channel pruning. On an embedded ARM platform, this technique compresses the model to detective work specimens. The argonon instruction , OpenMP technologies are more likely to optimize and improve convolutional network intensive computation, and finally come through a time embedded object detection.

Published in: 2020 13th International Congress on Image and Signal Processing, Bio-Medical Engineering and Informatics (CISP-BMEI)
Date of Conference: 17-19 Oct. 2020
Date Added to IEEE Xplore: 25 November 2020
ISBN Information:
INSPEC Accession Number: 20227265
DOI: 10.1109/CISP-BMEI51763.2020.9263648
Publisher: IEEE
Conference Location: Chengdu, China

3. METHODOLOGY

We have implemented this project using two different algorithms.
1. HOG – Histogram of Oriented Gradients
2. DNN – Deep Convolution Neural Network

HOG
It is an element descriptor utilized in PC vision and picture handling to identify the items. Algorithm implementation includes following steps:
Algorithm implementation includes following steps:
a. Gradient computation
b. Orientation binning
c. Descriptor blocks
d. Block normalization
e. Object recognition

This strategy will tally the recurrence of angle direction m a confined extent of the picture. HOG descriptor identifies whether or not each pixel is an edge. It extracts gradient and orientation by calculating both direction and magnitude by breakingdown the image into smaller regions. Histogram with 9 bins will then be created by categorizing themagnitude or the gradient into 9 different directions from O to 180.Another critical point to note is that the descriptor scales down the image to 1:2 ratio (width to height). For the ease of calculations, the size of 64x128 is preferred in almost all of the cases since the image will be broken down into blocks or cells of 8x8 to extract the features. These small changes in the x and y directions are then calculated to form the gradients. This process will be iterated across all pixels of the image. This can be simply thought of the change in the intensity across x and y directions for each individual pixel. By taking the square root of the summation of the square of gradient values across x and y axes, the magnitude will be obtained. Direction for this magnitude is obtained by an inverse tangent. As mentioned earlier, each 8x8 cell is then represented as a histogram, which is divided into 9 bins, which covers the angle from 0 to 180 degrees. These bins are equally distanced by a difference of 10 degrees. The magnitude of the 8x8 cells for all pixels in the image conveys the length that needs to be considered for each bin in the histogram.

In order to partially remove the noise or the unwanted dense areas with high intensities, we will smooth or normalize the data. This brings us to the topic of block normalization. This normalizing of variation in the light is obtained by considering the 16x16 cells.By combining all the features of 16x16 cells that we have obtained so far, features of the final image will be obtained. Thus the HOG feature descriptor uses the intensity or rather changes in the intensity as the main criteria in finding out the features of the image.

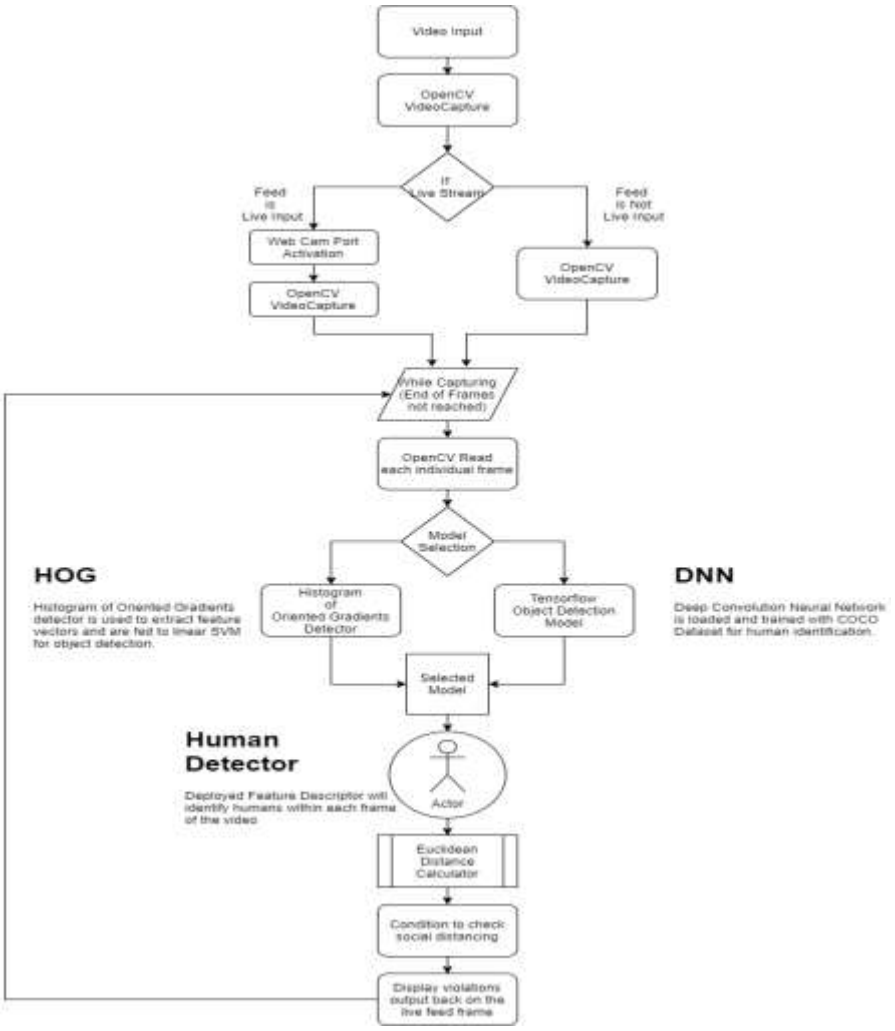
DNN
For our project, we have considered the model as ssd_mobilenet_v1_coco_2017_11_17. This is a pre-trained model on the novel data set from Common Objects in Context (COCO). This single convolution network predicts bounding box locations by classifying these locations in just one pass. This architecture makes single-shot detection (SSD) quick and efficient. By using the feature maps, SSD identifies the coordinates of bounding boxes. The exact shape of the object which is to be detected is not predicted by the SSD model, but rather the outline of where the box is located will be displayed. Mobilenet architecture initially filters the input frame of the video in the depth-wise convolution layer, and then it combines all these filtered outputs from the depth-wise convolution layer to generate new features for object detection. Input to this model should be a tensor created from the image. The output of this ssd_mobilenet_v1_coco_2017_11_17 model always gives us a dictionary object with four items namely

- Detection_scores with Float Datatype and with a size of 100
- Detection_classes with Float Datatype and with a size of 100
- Num_detections talks about the number of objects that have been identified.
- Detection_boxes if shape (, 100, 4) which specifies the location of the bounding box.

Detection_scores is the score given by the model for an object that it has identified. It can range up to 100 as percentage which implies the accuracy or confidence.
Detection_classes is the label provided by the model for the object that it has identified. These labels will be used in training of the model.
Num_detections will provide the exact number of objects identified by our model in one frame.
Detection_boxes provides us the location of the bounding box for every object in the frame. The results from detection_boxes are in normalized form ranging from 0 to 1.

Working
The input video is processed by iterating over the frames. Each frame is fed into the SSD Object detection model. Firstly, the frame is converted into a NumPy array and then converted to a tensor. A new dimension needs to be added to this frame as the model expects more than one image. Hence a new empty dimension is added. When this frame is passed to the model, we will receive a dictionary object with four items, as mentioned above. We will then be filtering out the objects only with the class of value '1', which implies a "Person." A threshold needs to be set based on the detection_scores to further filter out the boxes. In our case, this is set to a default of 0.35. These filtered out boxes are in normalized form and needs to be scaled to the frame size by multiplying with frame height and width. This information of the location of bounding boxes is then fed to OpenCV to plot the rectangles to further process the flow.

Design Flow



Input video provided to our algorithm is captured by using the VideoCapture module in OpenCV. The system will check whether the provided input video is a live feed of data or rather a previously recorded data. If the feed that we need to process is a live video, then we need to initialize the webcam port before we start to capture the video. Otherwise, if the feed is provided as input from a specific path, we don't need to initialize the webcam port. The captured video will then be divided into frames for further analysis. Each frame will be read by using the OpenCV.read() method before providing it to the feature descriptors. Based on the user input, analysis of Individual frames will be carried out by either Histogram of Oriented Descriptor or the TensorFlow's Deep Convolution Neural Network descriptor to identify the humans in the frame. Each individual human will then be outlined by a rectangular box before proceeding to calculate the Euclidean distance amongst all the individuals in the provided frame.

The centroid of each rectangular box outlining each individual is considered for the distance calculation. Pairwise Euclidean distance is then computed and checked for a threshold of approximately 6 feet, which is the height of the rectangular box that we have. If a person is found violating this distance with any one individual, he will then be outlined by a red rectangular box in the output video. However, if a person is maintaining a reasonable distance from others throughout the video, a green rectangular box will be displayed outlining that person in the output video. This process will be followed and repeated over all the frames within the video and will simultaneously be displayed with green or red rectangular boxes conveying the message of social distance practice. At any given instance, we are also calculating the number of violators by analyzing each frame, and the same is being displayed in the output. Along with this, we are also providing the percentage of people who are abiding by this social distancing practice in the output video.

Once the processing is done over entire frames of the video, we will be saving the output video, which can be played anytime in the future to evaluate how well social distancing is being practiced in that particular location.

Testing

Testing is one of the most important section of the project. We have performed below test cases using already recorded videos and also using live feed captured through system camera. We have illustrated results below.

1. HOG – Input a recorded video
2. DNN – Input a recorded video
3. HOG – Input live feed
4. DNN – Input live feed

Test case with HOG

Test results for recorded video or live feed using the HOG algorithm will be of similar type. In HOG, pixels with high intensity are highlighted and detected as objects. In certain instances, even empty places with high light intensity were treated as humans, and a rectangular box was positioned, which is incorrect. Using the centroid point of the objects, we are calculating Euclidean distance. If the Euclidean distance is less than the threshold value, the distance between those two objects or humans is less than 6 feet, and social distancing is being violated. For those objects which are violating social distance are highlighted with red rectangular box and centroid is also turned to red. If the distance is greater than the threshold, then those objects are practicing social distancing and are highlighted with a green rectangular box and with green centroids. If an object with another object is detected as violating social distance, then both objects are highlighted with a red rectangular box and red centroids. Then both objects are not verified again for violation in that frame. If an object is not violating with another object, then it is marked as green and then further verified for violation with other persons. The same process is repeated for every frame, and results are displayed simultaneously. We have also calculated the number of persons & percentage of violating social distance for each frame and displayed the same. As said above in HOG, objects are detected based on the intensity of pixels. So those kinds of objects are also considered when evaluating and calculating violation.

Test case with DNN:
Test results for recorded video or live feed using the DNN algorithm will be of similar type. Highlighting a violated and non-violated object using DNN will be the same as the HOG algorithm. The major difference comes while detecting the object. In the Deep Convolution Neural Network algorithm, all objects are detected rather than just detecting humans. Unlike in the HOG descriptor, this algorithm accurately detects humans. But if in the frame, if two or more objects are together or very near, then all the objects are together considered as a single object, and a rectangular box is highlighted over the whole group rather than one individual. But this is rarely observed as most of the time, and the algorithm proved to be accurate. Here we have also calculated the number of persons &
percentage of violating social distance for each frame and displayed the same.
We were unable to identify metrics to evaluate the performance of both these object detectors without actually avoiding human intervention. Had there been already a dataset with all possible humans in all the frames of the video, models could have been evaluated efficiently by using the labels. As this wasn't the case, human intervention was unavoidable when it comes to evaluating the models for accuracy on our code.

4. RESULTS

Below are outputs displayed when we execute the both models.
1. Highlight objects with Red color if violate social distance.
2. Highlight objects with Green color if no social distance violation.
3. Highlight objects center point with Red color if violate social distance.
4. Highlight objects center point with Green color if no social distance violation.
5. Display number of violations means number of objects violate social distance in frame.
6. Display percentage of violation in frame.
7. Save processed video of HOG and DNN algorithm for given video input.
8. Save processed video of live feed.

5. ACKNOWLEDGEMENT

We would like to extend our thanks to ECE Department of SRM Institute of Science and Technology for providing help to us in this pandemic situation.

6. REFERENCES

[1] https://docs.opencv.org/2.4/
[2] https://www.tensorflow.org
[3] https://www.pyimagesearch.com/2015/11/16/hog detectmultiscale-parameters-explained/
[4] https://www.programcreek.com/python/example/84776/cv2.HO GDescriptor
[5] https://www.analyticsvidhya.com/blog/2020/04/build-your-own object-detection-model-using-tensorflow-api

