Turkish Journal of Computer and Mathematics Education Vol.12 No.6 (2021), 5199-5206

Research Article

Crowd Manager during Pandemic using Video Analytics

Anjana Seby[a], Annu Baby[b], Bismi Rose Thomas[c], Cerita Gerard[d], Dr.Varghese S. Chooralil[e]

[a,b,c,d] Rajagiri School Of Engineering and Technology Kochi,682030 India
[e]Asst. Professor, Department of Computer Science, Rajagiri School of Engineering and Technology, Kochi, 682030 India
[a]anjanaseby@gmail.com, [b]annubaby98@gmail.com, [c]bismi2999@gmail.com, [d]mariacerita@gmail.com,
[e]varghesesc@rajagiritech.edu.in

Article History: Received: 10 November 2020; Revised 12 January 2021 Accepted: 27 January 2021; Published online: 5 April 2021

Abstract: As the COVID-19 pandemic sweeps through the world, we are in need of a system that can manage the flow of people. With the increase of Covid-19 cases, crowds need to be monitored, be it in a public place or in a building. Human monitoring can be quite tiresome and expensive, making way for the upcoming of automated crowd monitoring systems. In this paper, a framework for detection and analysis of crowd flow is proposed. The automated crowd flow analysis and mask detection in buildings eliminates the need for manual monitoring

Keywords: Video analytics • Covid-19 • Convolutional Neural Network • Computer Vision • People Counting • Mask Detection

1. Introduction

During the pandemic situation, there is a need to control the crowd and flow of people/customers in public places like banks, hospitals, govt. buildings etc. In most places the government has imposed a limit on the number of people who can be inside the building at a time ,so as to prevent COVID-19 transmission. People trying to enter a building do not have the facility to know how many people are inside the building. Sometimes workers in the public places are assigned with the job of a crowd manager. But this is not an efficient or a productive method to control the crowd. The crowd manager software detects the number of customers by analyzing the body movements from live video data captured from surveillance cameras placed inside buildings. We can count the number of people who are heading "in" and "out" of a building in real-time. Mask detection is usually done manually which is time-consuming. Face Mask Detection System uses existing CCTV surveillance cameras combined with Computer Vision to detect people without masks.

The software displays the count of people inside the building and will also produce an alarm if it exceeds the occupancy limit. It is useful during these COVID-19 times, and also during any pandemic situation in the future. An additional feature such as mask detector further contributes to the prevention of the virus spread by detecting people without masks.

Counting people in a crowded surveillance environment is a challenge task due to low resolution, occlusion, illumination change, imaging viewpoint variability, background clutter, etc. The major assumptions we consider are that the employ- ees are already inside the building and only customers have an occupancy limit, all customers pass through the video surveillance area which is a sufficiently lit environment, no blind spots and a single exit/entry point.

2. Literature Survey

In recent years, detecting and tracking people is a challenging task in crowded environments such as theatres,banks,hospitals due to the global spread of SARS- CoV-2 virus[1].Monitoring the access of people who pass through a certain en- trance is necessary to maintain safety inside buildings. An efficient automated system to manage the crowd count is therefore essential[2].

The initial people counting approaches that were implemented are designed to process the video stream produced by traditional RGB cameras [3, 5]. A combi- nation of a foreground detection method and tracing techniques are employed by most of these approaches. But when more than one person passes, this approach tends to cause errors. Mainly two approaches can be used to implement people detection and counting. The first approach is using a single overhead mounted camera, the system counts the number of people going in and out of an observed area. This approach uses the line of interest (LOI)[3], in which a camera is fixed on a virtual line and people crossing this line are counted. Another approach that can be used is the region of interest(ROI),in which people in a certain area are counted[4].

Del Pizzo et al.,in [6] elaborates on a visual based approach which processes the video stream acquired by traditional RGB cameras mounted in a zenithal position with respect to a virtual counting line. This method analyzes the video stream to count the number of people crossing the virtual line and determines the

direction in which each person crosses the line. Crowd counting methods on single images may result in incorrect or inconsistent counts for neighboring frames in the video crowd count.Liciotti et al. [7] introduced an approach to overcome this problem by tracking and detecting people, in case of heavy occlusions based on CNNs for semantic segmentation using top-view depth visual data.

The first step towards detection of face masks in people is face recognition. We would have to detect a face from an image that has several different attributes in it.D. Meena and R. Sharan[8],stated that research into face detection requires expression recognition, face tracking, and pose estimation. The main challenge is to correctly identify a face from a given image. Face detection is a difficult task because the faces change in shape, size, color, etc. and they are not immutable.

In [10],the authors used a method for the identification of faces using Gen- eralized Intersection over Union (GIoU) based on Mask R-CNN. The usage of bounding boxes may add noise to the face features and reduces the accuracy of detection. Instead of using bounding boxes, this method helps to reduce the background noise by correctly identifying the face.

Bosheng Qin,Dongxiao Li[9],have developed a method to identify how a per- son is wearing the face mask. They were able to classify three categories of facemask-wearing condition namely correct facemask-wearing, incorrect facemask-wearing, and no facemask-wearing.

3. Methodology

The framework used for crowd detection and tracking will be described in this section. People counting method based on detection and tracking to eval- uate the total number of people who pass through the surveillance camera and checks whether each person is wearing a mask or not. Python3 and OpenCV are used for implementing this proposed method.

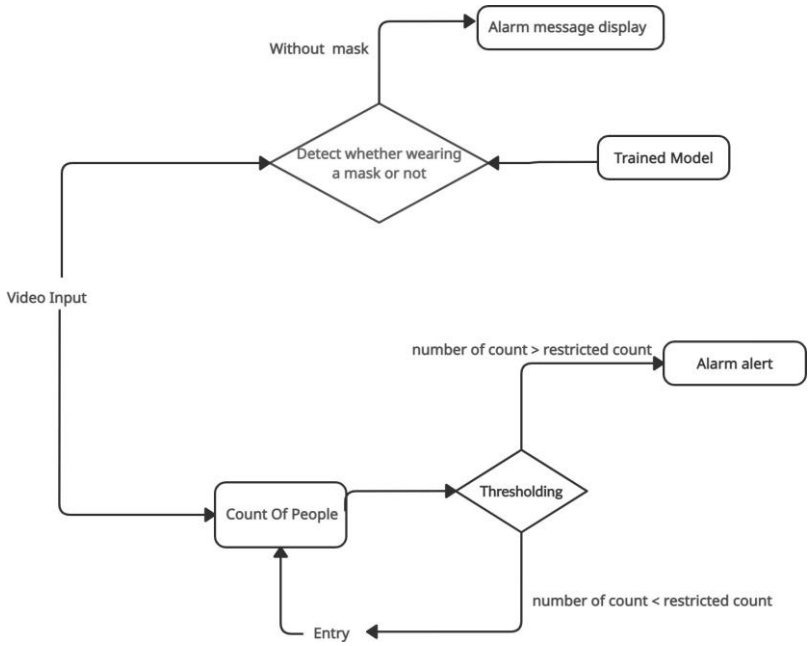

Fig. 1: System Flow Diagram

3.1 People Counter

To implement the people counter OpenCV and dlib are used. dlib is used for its implementation of correlation filters. The correlation filter uses a previously designed template to generate high response to regions that are similar to the target object while suppressing responses to distractors which are not similar to the target object. The VideoStream and FPS modules from imutils.video helps to work with a webcam and to calculate the estimated Frames Per Second (FPS) throughput rate.

In people counting we mainly go through two algorithms :

– Object Detection

– Object Tracking

When **Object Detection** is applied, it determines where in an image/frame an object is. Here we mainly use the SSD(single shot detectors) for object de- tection, which is a fast approach for object detection.

An **Object Tracker**, on the other hand, accepts the input (x, y)-coordinates of the location of an object in an image and:

1. Assigns a unique ID to that particular object,
2. Track the object as it moves around in a video stream.

To obtain highly accurate object detection and tracking we create a single algo- rithm by combining the concept of object detection and object tracking, typically divided into two phases:

Phase 1 — Detecting: During the detection phase we run the object tracker to (1) detect if new objects have entered the view, and (2) checks whether if we can find objects that were lost during the tracking phase. For each object that we detect, we create or update an object tracker with the new bounding box coordinates. We run this phase only once every N frames due to the computationally expensive nature of the object detector.

Phase 2 — Tracking: When not in the "detecting" phase its in the "track- ing" phase. For each of the detected objects, create an object tracker to track the object as it moves around the frame. Continue tracking until reachesthe N-th frame and then re-run the object detector. The entire process then repeats.

Centroid tracking algorithm is also used to track, register and deregister objects.

The steps involved in centroid tracking are :

Step 1: Accept a set of bounding boxes and compute their corresponding cen- troids (i.e., the center of the bounding boxes)

Step 2: Compute the Euclidean distance between any new centroids (yellow) and existing centroids (purple).

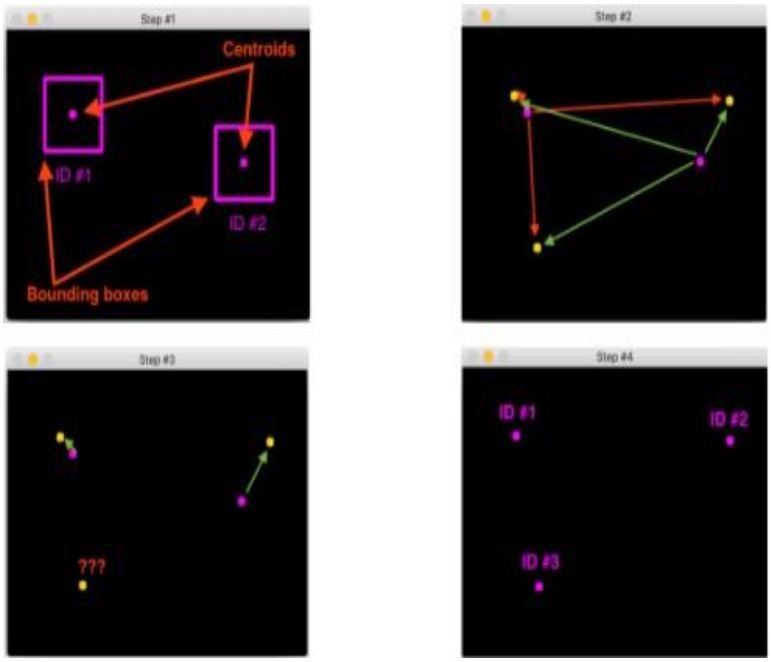

Fig. 2: Centroid Tracking

Step 3: Update (x, y)-coordinates of existing objects.

Step 4: Register new objects

Step 5: Deregister old objects

The bounding boxes themselves can be provided by either an object detector or an object tracker. In the

above image, ie, Figure 2 there are two objects to track in this initial iteration of the algorithm. To build a simple object track- ing via centroids script with Python, the first step is to accept bounding box coordinates and use them to compute centroids. During Step 2 in the figure com- pute the Euclidean distance between any new centroids (yellow) and existing centroids (purple). In the centroid tracking algorithm, we assume that pairs of centroids with minimum Euclidean distance between them can be considered as the same object ID. There are two existing centroids (purple) and three new centroids (yellow), implying that a new object has been detected (since there is one more new centroid vs. old centroid). The arrows between the centroids represent the computation of the Euclidean distances between all purple centroids and all yellow centroids. Once the Euclidean distances are calculated then attempt to associate object IDs in the next step. In step 3, Update (x, y)-coordinates of existing objects. In Step 4, registering new objects happens. Registering the object simply means adding the new object to the list of tracked objects by assigning it a new object ID to it and then storing the centroid of its bounding box coordinates. In the event that an object has been lost or has left the field of view, simply deregister the object (Step 5). For the people counter, we deregister people IDs when it cannot be matched to any existing person objects consecutively for 40 frames.

In order to identify only the person class from the video input we use the Caffe pre-trained Convolutional Neural Network(CNN) model. Then loads the pre-trained MobileNet SSD used to detect objects. The network which has been chosen for the detection task is the MobileNet, which has proven to be the most effective in this kind of situation for its suitability to perform object detection, fine grain classification, face attributes etc. The minimum threshold to filter out weak detections is set to the value of 0.4. Then skips 30 frames between detecting objects with the OpenCV DNN module and CNN single shot detector model. DNN module of openCV's enables to pass the frames of video through the network and obtain the output bounding box(x,y)-coordinates of each object in the frame.

We loop over frames from the video stream. Preprocessing of data is done by resizing the frame to have a maximum width of 500 pixels for faster processing, then convert the frame from BGR to RGB color format for dlib implementation. To avoid running the object detector on every frame, skips every N frames. N is set to a value of 30. Skipping frames can increase the speed of tracing pipeline. Only every N frames will we use our SSD for object detection. Otherwise, simply tracking moving objects in-between. For object detection, begin by creating a blob from the image, followed by passing the blob through the network to obtain detections. By setting the input to the network and compute the forward pass for the input and store the result as detections. By looping over our detections, apply a probability check with each detection .Filter out weak results and those that don't belong to the "person" class by requiring a minimum threshold. Compute the bounding box around the detected object and extract the (x, y)-coordinates of the box. After constructing a dlib rectangle object from the bounding box coordinates and then start the dlib correlation tracker. Add the tracker to the list of trackers so it utilizes it during skip frames. Most of the time, we are not landing on a skip-frame multiple. During this time, we'll utilize our trackers to track the object rather than applying detection. The crossing-line judgment is used to determine whether the particular person will get counted or not to be counted. Draw a horizontal line in the center of the frame - once an object crosses this line, determine whether they are moving 'in' or 'out' of a building. CentroidTracker is utilized to accept the list of objects, both generated via object detection or object tracking. CentroidTracker will associate object IDs with object locations.

To count the number of people that has moved in or out through the frame by looping over the updated bounding box coordinates of the object IDs. Then attempt to fetch a TrackableObject for the current objectID.If the TrackableOb- ject doesn't exist for the objectID, create a new one. Otherwise, there is already an existing TrackableObject, so determine whether the person is moving in or out. To do so, grab the y-coordinate value for all previous centroid locations for the given object. Then compute the direction by taking the difference between the current centroid location and the mean of all previous centroid locations. Then we check to see if the object has been counted or not. If the TrackableOb- ject has not been counted determine if it's ready to be counted yet by, checking if the direction is negative which indicates that the object is moving out and the centroid is above the center-line. In this case we decrement the people count. If the direction is positive which indicates that the object is moving in and the centroid is below the centerline then increment the people count. Finally, store the TrackableObject in trackableObjects dictionary so we can grab and updateit when the next frame is captured.

3.2 Mask Detector

We proposed an automated smart framework for screening persons who are not using a face mask. To implement this, we use a Haar cascades classifier to detect faces from images and a CNN model to classify

the detected faces as with mask and without mask.

A. Image Preprocessing

The images captured by the CCTV cameras require preprocessing before going to the next step. During this step, the image is converted into a grayscale image since the RGB color image contains so much information about the features that is not necessary for face mask detection. We use the statement *'rescale=1./255'* for this conversion. *rescale* is a value, that we will multiply the data before any other processing. The original images consist of RGB coefficients in the range 0-255, but such values might be too high for our model to process so we take target values between 0 and 1 instead of scaling with a 1./255 factor. *shear range* is for applying shearing transformations. We set *shear range=0.2.zoom range* is for randomly zooming inside pictures and it is set as *zoom range=0.2*. *horizontal flip* is for randomly flipping half of the im- ages horizontally and it is set as *True*.

Dataset consisting of 1376 images is used for training and testing the model. We collected a total of 690 images of people with masks and 686 images of people without a mask. For training purposes, 80% images of each class are used andthe rest of the images(i.e. 20%) are utilized for testing purposes.

B. Architecture Of CNN Model

The current method makes use of Sequential CNN. The First Convolution layer is followed by Rectified Linear Unit (ReLU) and MaxPooling layer. The CNN comprises an input layer, several hidden layers and an output layer. The hidden layers consist of multiple convolution layers that learn suitable filters for important feature extraction from the given samples. The features that are extracted by CNN are used by multiple dense neural networks for classificationpurposes.

The architecture contains three convolution layers each followed by one max pooling layer. This layer decreases the spatial size of the representation and thereby reduces the number of parameters. As a result, the computation is simplified for the network. Then, a flatten layer reshapes the information into a vector to feed into the dense network. A pairs of dense layers learn parameters for classification. The first dense layer comprises a series of neurons each of them learn nonlinear features and uses the ReLU activation function. The second denselayer contains two input neurons and uses the sigmoid activation function.

The learning process is configured first with the compile method. Here we use "adam" optimizer. Binary crossentropy is used as a loss function. As we are dealing with a classification problem, metrics is set to "accuracy". The above model is trained for 10 epochs and is saved at the end.

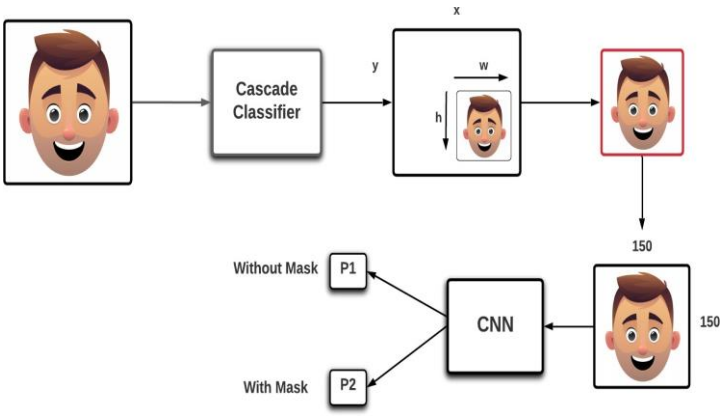

Fig. 3: Overview of Mask Detector

C. Mask Detection Using The Trained Model

The video stream is captured from the CCTV cameras. The Haar cascades classifier is used to detect faces from the input video stream. Each face detected is then passed on to the already trained model, which then classifies the image into 'with-mask' or 'without-mask'. The architecture identifies whether any input image contains persons without a face mask.

4. Experimental Results

The people counting and mask detection features were separately tested and the experimental results for

the same are shown in the figures.

	Actual Count	Output Count	Error percentage
Morning	112	110	1.78%
Evening	112	107	4.46 %
Night	112	105	6.25%

Table 1: People counter experimental observations based on lighting conditions

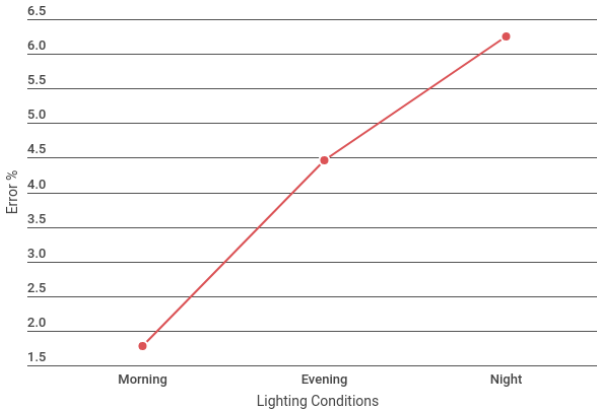

Fig. 4: People Counter experimental observations based on lighting conditions Table 1. represents the experimental observations of people counter based on

different lighting conditions. We tested the people counter for a total of 112 people passing through the entry of a building during 3 lighting conditions namely, morning, evening and night. The highest accuracy was achieved during the morning due to optimal lighting conditions with an error of 1.78%. A slight increase in error can be observed when the lighting condition switches to evening. The lowest accuracy was observed during the night due to poor lighting conditions with an error of 6.25%. The error percentage is plotted against the different lighting conditions as shown in Fig. 4. A slight increase in error can be observed with decreased lighting.

	Actual Count	Output Count	Error percentage
Dense crowd	209	188	10.04%
Medium dense crowd	115	109	5.21%
Sparse crowd	56	55	1.78 %

Table 2: People counter observations based on density of crowd

Similarly Table 2 shows experimental observations of people counter based on different crowd densities. We tested people counter for three

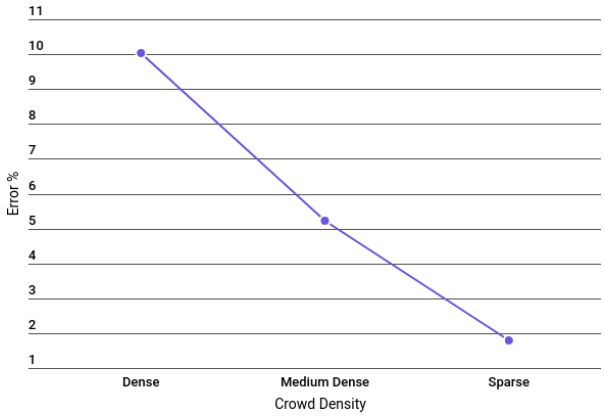

Fig. 5: People Counter experimental observations based on density of crowd

different crowd densities such as dense, medium dense and sparse having crowd counts, 209,115 and 56 respectively. Detecting people in dense crowds produced a greater error of 10.4% when compared to sparse crowds which generated the least error. The error percentage is plotted against different crowd densities as shown in Fig. 5. A similar increase in error can be seen as crowd density increases.

		Predicted Class	
		Without Mask	With Mask
True Class	Without Mask	134	1
	With Mask	3	170

Table 3: Confusion matrix for Mask Detector

Table 3 represents the confusion matrix of the mask detector. The model misclassifies only 4 samples out of 308 samples. It classifies 1 sample as 'with mask' while it is in 'without mask' class and classifies 3 samples as 'without mask' while these were in 'with mask' class. The main aim of the mask detection model is to identify samples within 'without mask' class and this model only misclassified 1 sample of this class that shows the reliability of the developed model.

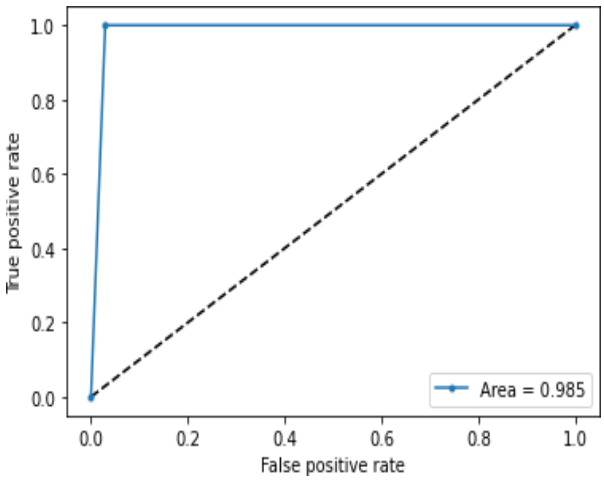

Fig. 6: ROC of the classification network

Fig.6 depicts the receiver operating characteristic (ROC) curve of the pro- posed mask detection model. This shows the ability of the classifier to predict at different thresholds. Two parameters are plotted in the ROC curve, one is the true positive rate (TPR) and other is the false positive rate (FPR). TPR and FPR for different thresholds are plotted as the ROC curve. The area under the ROC curve (AUC) measures the performance of the binary classifier. The value of AUC ranges from 0 to 1.AUC is 1 when the model predicts 100% correct and is 0 when it predicts 100% wrong. The AUC achieved from our classifier model is 0.985 that points towards a good classifier.

5. Conclusion

A computer vision based approach thus provides a smart and effective way for properly controlling crowd during pandemics in banks, hospitals, govt. buildings etc. This system helps in properly allocating and managing crowds in every pandemic situation. The features in our project i.e. people counter and mask detector using the object detection and tracking algorithm ensures that people are following the basic guidelines during this pandemic. We thus reduce the chance of being affected by the pandemic with the help of this crowd controllingsystem.

In overly dense crowds, the camera might not be able to point on every person in the frame, or due to occlusion making it difficult to get a proper count as well as track them which needs to be improved. Due to decrease in accuracy in people counting during these conditions, mask detection also becomes a challenge. The developed system faces difficulties in classifying faces covered by hands since it almost looks like the person is wearing a mask.

This paper presents a people counting system as a way to manage crowds by controlling the count of people. Keeping in mind the Pandemic situation Mask- Detection feature was also added. If the count exceeds the occupancy limit or if the model recognizes whether people are not wearing masks then the alarm is produced. The proposed system will reduce the time taken by humans for managing the crowd count and for checking masks and ensures them that this work is done by the system itself in no time. This model requires comparatively less time and provides great accuracy.

References

1. Pazzaglia, G., Mameli, M., Rossi, L., Paolanti, M., Mancini, A., Zingaretti, P. and Frontoni, E., 2021. People Counting on Low Cost Embedded Hardware During the SARS-CoV-2 Pandemic. In Pattern Recognition. ICPR International Workshops and Challenges: Virtual Event, January 10–15, 2021, Proceedings, Part II (pp. 521-533). Springer International Publishing.
2. Gowari, Y., Gaikwad, R., Gurav, A. and Raut, V., 2021. People Monitoring and Mask Detection Using Real-Time Video Analyzing (No. 4997). EasyChair.
3. Barandiaran, J., Murguia, B. and Boto, F., 2008, May. Real-time people counting using multiple lines. In 2008 Ninth International Workshop on Image Analysis for Multimedia Interactive Services (pp. 159-162). IEEE.
4. Ryan, D., Denman, S., Fookes, C. and Sridharan, S., 2009, December. Crowd counting using multiple local features. In 2009 Digital Image Computing: Tech- niques and Applications (pp. 81-88). IEEE.
5. Chen, T.H., Chen, T.Y. and Chen, Z.X., 2006, June. An intelligent people-flow counting method for passing through a gate. In 2006 IEEE Conference on Robotics, Automation and Mechatronics (pp. 1-6). IEEE.
6. Del Pizzo, L., Foggia, P., Greco, A., Percannella, G. and Vento, M., 2016. Counting people by RGB or depth overhead cameras. Pattern Recognition Letters, 81, pp.41-50.
7. Liciotti, D., Paolanti, M., Pietrini, R., Frontoni, E. and Zingaretti, P., 2018, August. Convolutional networks for semantic heads segmentation using top-view depth data in crowded environment. In 2018 24th international conference on pattern recognition (ICPR) (pp. 1384-1389). IEEE.
8. D. Meena and R. Sharan, "An approach to face detection and recognition," 2016 International Conference on Recent Advances and Innovations in Engineering (ICRAIE), Jaipur, 2016, pp. 1-6, doi:10.1109/ICRAIE.2016.7939462.
9. Bosheng Qin,Dongxiao Li. Identifying Facemask-wearing Condition Us- ing Image SuperResolution with Classification Network to Prevent COVID19, 13 May 2020, PREPRINT (Version 1) available at Research Square[+https://doi.org/10.21203/rs.3.rs- 28668/v1+]
10. Ren, J., Hussain, A., Zhao, H., Huang, K., Zheng, J., Cai, J., Chen, R. and Xiao,
11. Y. eds., 2020. Advances in Brain Inspired Cognitive Systems: 10th International Conference, BICS 2019, Guangzhou, China, July 13–14, 2019, Proceedings (Vol. 11691). Springer Nature.