# Real-Time Face Mask Detection, Tracking and Counting using Haar Cascade Classifier and Motion Analysis

Pietro Basci
Politecnico di Torino
s266004@studenti.polito.it

## Abstract

*The objective of this work is to build a system that is able to monitor in real-time the use of face mask in places were it is considered mandatory. To do that, motion analysis technique will be used to find people inside the frame and then those regions will be analyzed using the Haar Cascade Classifier to detect masks.*

*Motion analysis allows to detect moving objects inside a frame by learning background according to history of frames and then performing a background subtraction on the new frame. This method will be used to identify the region of interest that will be analyzed by the classifier. To each ROI will also be assigned a centroid with an unique id to perform the count.*

*Once identified the region, its content will be analyzed by two Haar Cascade Classifiers: the first will identify frontal faces, while the second will be used on faces extracted by the first to detect masks. People will be tracked during all the time they appear inside the recorded area and only when they disappear for a given number of frames, counters will be updated.*

## 1. Introduction

During the COVID-19 pandemic period, the use of face masks has been adopted by almost all countries as a prevention measure to reduce the number of contagions.

A lot of studies shows that wearing a mask in crowded spaces such as airports, railway stations and malls, or when interpersonal distances are not guaranteed, helps to reduce risks of infections.

Therefore, a real-time system able to detect face masks on videos gathered by cameras can provide important informations that can be exploited for public safety. These can be used to increase controls where a large number of people without masks are detected, or can be simply used to trigger a signal that remembers to wear a mask.

The problem of face masks detection has been addressed in different works mainly based on deep learning approaches. Despite these methods are very powerful tools on these tasks, the main drawback is that they are quite computationally expensive: without the use of a GPU they are not able to provide a real-time detection. The idea here is to try to build a system that does not exploit the most advanced neural networks architectures for the object detection so that it can run in real-time also on CPU. This aspect could make the system more suitable for this specific application since it can help to reduce the hardware cost.

The system will perform the people counting acting in different steps: firstly it will identify people inside the frame and will start to track them, and then it will increase the counters (*with_mask*/*without_mask*) according to the results of classifications performed on people faces.

To make classifications, two different Haar Classifier will be used: the first will be the default frontal faces Haar Classifier provided by OpenCv, then a new classifier, trained for this specific task, will be used to define if the face wears or not a mask.

The new classifier will be evaluated using different metrics that give informations about the quality of predictions as well as the inference time. Since the research has monitoring applications, it should be appropriate to reduce as much as possible false negatives in order to obtain an accurate count of people without masks and reduce the number of false warnings.

## 2. Dataset

### 2.1. Overview

The dataset used to train the new Haar Classifier, has been collected from different sources available on Kaggle [5][6][7][8]. Specifically, a couple of scripts has been realized to extract only close-up faces (wearing or not wearing mask) and divide them into two folders: positive and negative. The resulting dataset was composed by 4509 positive and 5432 negative images.

To properly evaluate the model, some samples have been kept for validation and test phases. Hence, the dataset has been split in three sets in the proportion 76:12:12. The resulting size of each split are: *Training Set* (7561), *Validation Set* (1190) and *Test Set* (1190). Figure 1 shows some samples of the considered dataset.

Figure 1: Examples of dataset's images

## 3. Methods and algorithms

The main role of the system is to make a count of people with mask and people without mask that enter inside a place where a mask must be worn. To make an accurate count, it is not possible to simply track faces identified by Haar Classifier. Indeed, due to false negatives that it can produce while a person crosses the area, it can introduce errors in the tracking phase and this can lead to a wrong final count.

Therefore, to avoid this problem, *motion analysis* technique has been considered to detect the full body and track it: each of the new bodies identified is associated to a new *centroid* with an unique *id*, while in the case of bodies whose centroid is near to a centroid found in the previous frame it is likely that it represents the same person that moved, so the id remains the same.

Once identified a person, a crop of the original frame is extracted and analyzed by two *Haar Cascade Classifiers*. In particular, the first classifier will be used to identify the face, while the second will be used to detect the mask on the face extracted by the first. This method can help to reduce the computation needed at inference time since a frame with no people inside does not requires to be analyzed by the classifiers, and also to further reduce possible false positives as the first classifier acts only on a reduced area and the second only on faces.

People will be tracked during all the time they appear inside the recorded area and only when they disappear for a given number of frames, counters will be updated. Counters will be updated according to results produced by the classifiers during all the tracking phase. Since they can produce some misclassifications, adopting a sort of *majority vote* to derive the final classification allows to reduce errors in the final count.

### 3.1. People Detection

In order to identify people inside a frame, the function *BackgroundSubtractorMOG2* available in OpenCV has been used. It implements the Gaussian mixture model background subtraction proposed by Zivkovic et al. and described in [2] and [3].

This function allows to compute the foreground mask, i.e. a binary image containing the pixels belonging to moving objects in the scene, performing a subtraction between the current frame and a background model, containing the static part of the scene. The background model is initialized at the beginning and then it is updated in order to adapt to possible changes in the scene.

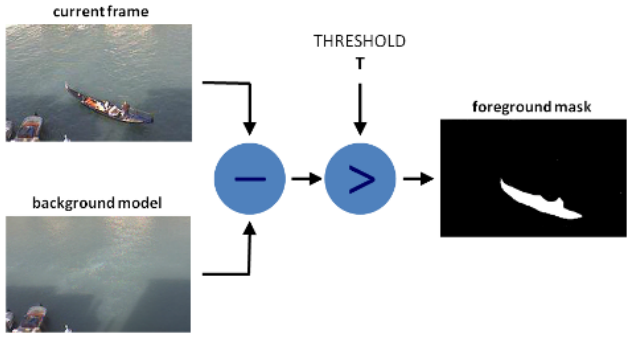


Figure 2: Background subtraction method [4]

*BackgroundSubtractorMOG2* allows to detect also shadows, in this case it returns a grayscale image. In this specific application, detecting also shadows and then performing a *threshold* operation to obtain the final binary mask produced better results. Moreover, on the foreground mask has been performed an *opening* operation to remove white spots due to noise, followed by a *closing* operation to fill black spots inside the object.

### 3.2. People Tracking and Counting

Once identified people inside the frame, a rectangle around each of them was found. The rectangle has been used to compute the *centroid* to track the movement of people. It is a point whose coordinates are 1/2 of the width and 1/3 of the height of the rectangle. The final result is shown in the following image.

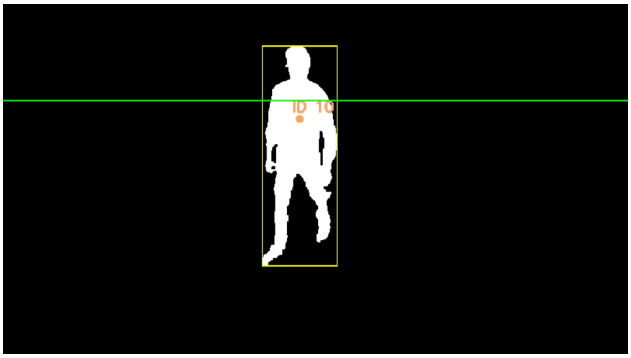


Figure 3: Example of person detected in a frame

Each of the new people identified in the current frame is associated to a new centroid with an unique *id*. In the case that into the previous frame there was a centroid with an euclidean distance with respect to the new one below a given threshold, it is likely that it describes the same person that was walking and so the id remains the same.

To count people that enter inside the place, the system keeps track of the centroid vertical position and increase the number of *people inside* counter when it crosses the limit represented by the green line. This count represents the total number of people that got inside the place.

The *people with mask* and the *people without mask* counters represent the number of people that got inside the place and has been recognized by the system as people with or without mask. These counters are updated only when a person, that crossed the green line, disappear from the recorded area for a given number of frames. This is used to implement a *majority vote* method that allows to reduce the effects of misclassifications and obtain a more accurate final count.

Another counter *undefined* has been consider to count people that the system was not able to recognize as people with or without mask. This could happen in the extreme case when the first classifier was not able to identify the face during all the tracking phase or when the second classifier recognized him as people with mask and people without mask for the same number of frame (no majority).

## 3.3. People Classification

The rectangle area identified in the previous step represents also the *region of interest* were the first classifier will look for the face. This reduce the area analyzed, allowing a reduction of the risks of false positive detections and computational costs. To further reduce the effort, it is possible to consider only the half rectangle at the top of the area since in this region are located heads.

To find faces, the *haarcascade_frontalface_default.xml* classifier provided by OpenCv has been used. Also in this case when a fece is detected, a crop of the original image is extracted and analyzed by a second classifier to detect the presence of a mask. To do that another Haar Cascade Classifier has been trained on the dataset described above, and tested to evaluate its performance.

### 3.3.1. Haar Cascade Classifier

Haar feature-based cascade classifiers is an effective object detection method proposed by Paul Viola et al. [1]. It is a machine learning based approach where a cascade function is trained from a huge set of images. Initially, the algorithm needs a lot of positive images (images of faces) and negative images (images without faces) to train the classifier. To extract features, Haar features shown in Figure 4 are used. Each feature is a single value obtained by subtracting sum of pixels under the white rectangle from sum of pixels under the black rectangle.

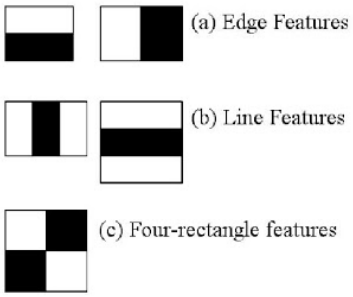


Figure 4: Haar features

To find the sum of the pixels under white and black rectangles in an efficient way, authors introduced the integral image: however large the image, it reduces the calculations for a given pixel to an operation involving just four pixels.

Haar Classifiers use Boosting that is a classification scheme that combines weak learners into a more accurate ensemble classifier. Boosting is used to create binary (face/not face) classification nodes characterized by high detection and weak rejection. The final classifier is a weighted sum of these weak classifiers. It is called weak because it alone can't classify the image, but together with others forms a strong classifier.

Since in an image, most of the image is non-face region, it is a better idea to have a simple method to check if a window is not a face region. If it is not, discard it in a single shot, and don't process it again. Instead, focus on regions where there can be a face. To do that, they introduced the concept of Cascade of Classifiers. Instead of applying all features on a window, the features are grouped into different stages of classifiers and applied one-by-one. If a window fails the first stage, it is discarded and the remaining features are not considered on it. If it passes, the second stage of features is applied and the process continues. The window which passes all stages is a face region.


Figure 5: Structure of Cascade Classifier

## 3.4.Evaluation Metrics

The first metric used to evaluate the model was *Accuracy* that represents the ratio of the number of correct predictions over the total number of predictions. Since it is not a good metric if the considered test set is unbalanced, also the *Confusion Matrix* has been analyzed. Confusion Matrix is a performance measurement for machine learning classification problem. It consists in a table with 4 different combinations of predicted and actual values

known as *True Positive*, *True Negative*, *False Positive* and *False Negative*.

From these values it is possible to compute some other metrics such as:

$$Precision = \frac{TP}{TP + FP}$$

which is the proportion of positive identifications that was actually correct;

$$Recall = \frac{TP}{TP + FN}$$

which is the proportion of actual positives that was identified correctly;

$$f1 - score = \frac{2 \cdot Precision \cdot Recall}{Precosoion + Recall}$$

which combines the previous two.

For this specific application, it should be appropriate to choose a model characterized by as less as possible false negatives (i.e. Recall near to 1) in order to obtain an accurate count of people without masks and reduce the number of false warnings.

Finally, since the system has to work in real-time, also the inference time needed by the model expressed in s/frame and so the relative fps (frame per second) has been analyzed.

## 3.5. Results

As introduced above, the dataset has been split into three sets: Training Set, Validation Set and Test Set.

The objective is to train the model on the training set, perform hyper-parameters tuning and model choice using the validation set. In this way, all those choice are not affected by test set's samples. Finally the selected model is tested on the test set to see how it works on new data, i.e. how good it is to generalize.

The parameters used to train the classifier analyzed are shown in Figure 6.

```
PARAMETERS:
cascadeDirName: output_model
vecFileName: mask.vec
bgFileName: bg.txt
numPos: 2900
numNeg: 4129
numStages: 25
precalcValBufSize[Mb] : 2048
precalcIdxBufSize[Mb] : 2048
acceptanceRatioBreakValue : -1
stageType: BOOST
featureType: HAAR
sampleWidth: 20
sampleHeight: 20
boostType: DAB
minHitRate: 0.995
maxFalseAlarmRate: 0.5
weightTrimRate: 0.95
maxDepth: 1
maxWeakCount: 100
mode: ALL
Number of unique features given windowSize [20,20] : 125199
```

Figure 6: Haar Cascade's parameters

### 3.5.1. Statistics

| ***Haar Cascade*** | Training Set | Validation Set | Test Set |
|---|---|---|---|
| True Positive | 2549 | 403 | 446 |
| False Positive | 456 | 88 | 38 |
| True Negative | 3673 | 564 | 613 |
| False Negative | 883 | 135 | 93 |
| Accuracy | 82.3% | 81.3% | 89.0% |
| Precision | 84.8% | 82.1% | 92.1% |
| Recall | 74.3% | 74.9% | 82.7% |
| F1-Score | 79.2% | 78.3% | 87.2% |

Table 1: Classification metrics

The previous table shows that results on the training set are not far from results obtained on the validation sets with an accuracy around 82%. It means that, considering a more complex model, it is possible to further improve accuracy. Results on test set instead are the best, but these are related to this particular split of the original dataset: considering a different random split, results can change but they should remain around 80% of accuracy. Moreover, it is possible to see that the model achieves very high Precision score (above 92% on Test set). This means that among all positive predictions, less than 8% are wrong, so when a mask is detected it is very likely that it is a good prediction.

The system achieves an inference time of approximately 0.2 s/frame which corresponds to 5 fps running on CPU. However, inference time can slightly change depending on the number of people in the frame. Indeed, when no person is detected, classifiers are not used at all and fps can slightly increase. Instead, when more people are detected, classifiers are used to analyze each ROI identified, and consequently fps tends to decrease a bit (<0.015 s/frame). Previous results are an average obtained considering a video stream of 147s.

Tests has been conducted using a 2.3GHz 4-core Intel Core i7-based MacBook Pro with 8GB of RAM and using as input the video stream of the built-in 720p webcam.

### 3.5.2. Visualization

Figure 7 and Figure 8 show two outputs of the model on two different frames. At the bottom, there is the person segmentation and localization. At the top, there is the localization of the face and classification as a person with mask or without mask. In the left corner there are the counters used to track the usage of masks, while the green lines represents the entrance boundary.

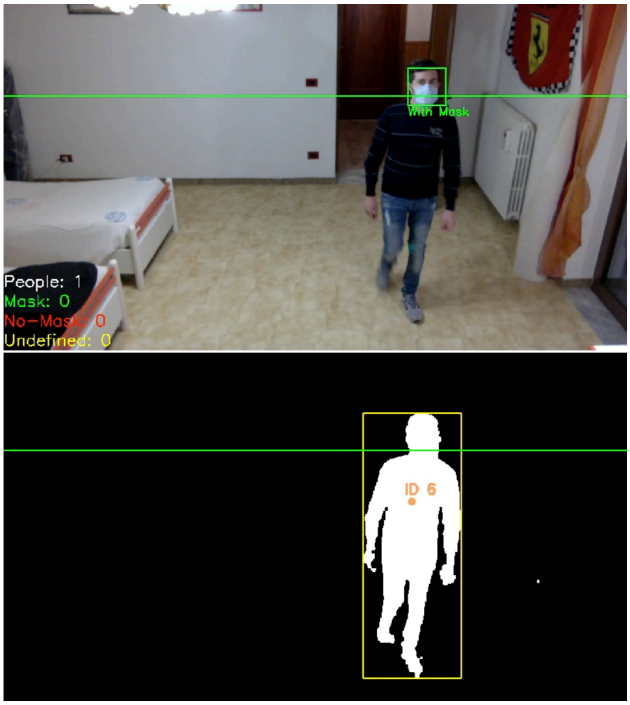


Figure 7: A frame representing a person with mask

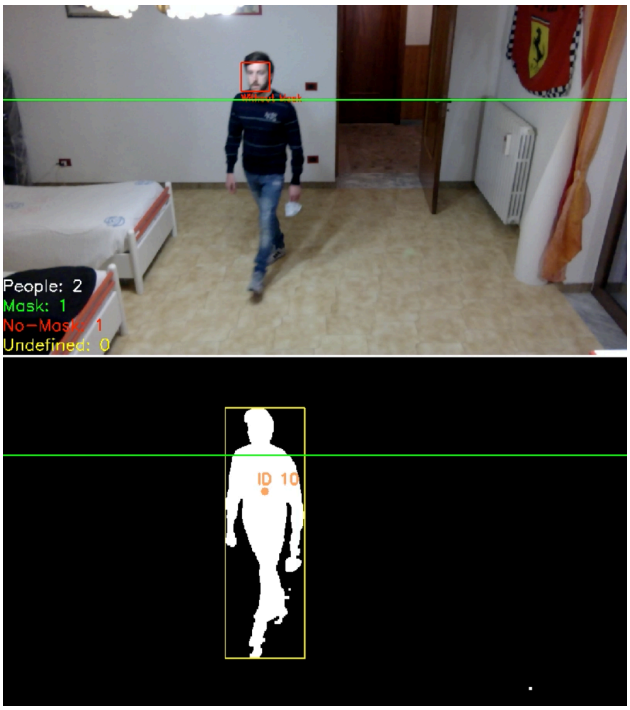


Figure 8: A frame representing a person without mask

## 4.Conclusion

In this work the problem of the face mask detection and counting has been addressed. The method proposed allows to perform an accurate count of people who enter inside a place where a mask must be worn and it is able also to detect if they are wearing a mask and accordingly update counters. Thanks to the majority vote method, the two counters (*Mask*/*No-Mask*) are updated considering all the classifications that the system made on each detected face and consequently it allows to reduce as much as possible the effects of any misclassifications.

The main limit of this system is that people inside the area must be sufficiently spaced each other. Indeed, since the system keeps track of the full body, if a group of people crosses the area, they could be counted as a single person. However the system will work properly in all those scenarios where one person at a time can enters or where more people at a time can enter but not as a group, for instance near subway turnstiles.

## 5. Future Work

As future work, it is possible to further improve the performance of the second model by choosing a more complex classifier that should help to reduce the number of misclassifications. Specifically, a Convolutional Neural Network such as *AlexNet*[9] could be used to perform the classification mask/no_mask on faces extracted by the first Haar Classifier. In this way, the first classifier acts as a sort of Region Proposal algorithm that extracts feces which are used to feed the CNN to perform the final classification in a more reliable way.

## References

[1] Paul Viola and Michael J. Jones, *Rapid Object Detection using a Boosted Cascade of Simple Features.* CVPR, IEEE Computer Society, 2001.

[2] Zoran Zivkovic, *Improved adaptive gaussian mixture model for background subtraction.* In Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on, volume 2, pages 28–31. IEEE, 2004.

[3] Zoran Zivkovic and Ferdinand van der Heijden. *Efficient adaptive density estimation per image pixel for the task of background subtraction.* Pattern recognition letters, 27(7): 773–780, 2006.

[4] https://docs.opencv.org/3.4/d1/dc5/tutorial_background_subtraction.html.

[5] https://www.kaggle.com/andrewmvd/face-mask-detection.

[6] https://www.kaggle.com/sumansid/facemask-dataset.

[7] https://www.kaggle.com/omkargurav/face-mask-dataset.

[8] https://www.kaggle.com/prithwirajmitra/covid-face-mask-detection-dataset.

[9] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "ImageNet Classification with Deep Convolutional Neural Networks." *Advances in neural information processing systems*. 2012.