Face mask detector: a Computer Vision application

Federico Agostini
federico.agostini.5@studenti.unipd.it

Federico Bottaro
federico.bottaro.1@studenti.unipd.it

Abstract

The very recent Covid-19 outbrake forced the population to adopt new habits in order to prevent and control the spreading of the disease. In particular, face masks represent one of the most important means to reduce its advance. Computer Vision and Neural Networks can be exploited to ensue the respect of these new social rules. The aim of this work is to build a pipeline to detect and recognize whether people are wearing face masks, both in images, in videos and in real time. The results of this work can be applied to automate the surveillance in public places, where human control can be a difficult task.

1. Introduction

Covid-19 epidemic highlighted the usage of face masks as the primary mean to contain and reduce the spreading of the disease. Many Countries decided to make it mandatory in public and crowded places. However, acting a strong control could be a difficult task for the Governments and Police forces without the use of Artificial Intelligence (AI). AI along with Computer Vision (CV) can be exploited to enhance and automate the process of controlling the respect of these basic social rules.

This work aims to build an autonomous system that can detect whether people wear face masks. The task can be split in two different ones(as suggested in [6]):

- Build and train a Neural Network (NN) to understand whether a person is wearing or not the face mask from an image of the face;
- Apply a face detector to extract faces from images and video, and then process them using the NN developed in the previous step.

The learning paradigm adopted to train the NN is supervised learning; hence we used a labelled dataset to build the classificator, which is discussed in Section 2. The final performances will be influenced by the complexity of the data, and the actual scenarios where the NN should be applied also will depend on it. The architecture and the implementation of both the NN and the face detector, that is implemented throght the CV package `OpenCV`, are developed in Section 3.

2. Dataset

The quality of the dataset used to train the NN heavily influences its performance in real life scenarios. Since the novelty of the task, there are no big and well established labelled datasets of people wearing face masks. One possible solution (suggested in the work of Prajna Bhandary [1]) is to artificially build our data by placing images of face masks on top of existing face pictures, using facial landmarks to put it in the right position. The drawback of this approach is that it could lack of generalization capabilities, and not perform well when applied in different situations. In addition, it may only learn to identify masks with the same color as the one used to build the dataset, whether they can have different shapes and colors.

Figure 1: Examples of images used to train the network.

The dataset we decided to use is hosted on *Kaggle* [1]. This one combines both real and stylized images, as shown in Fiure 1. The original dataset is split into training, validation and test subsets; however we decided to join train and validation, in order to increase the size of the training set. Even in this way, the number of images of people wearing face masks is quite small (about 800); to overcome this problem,

[1] https://www.kaggle.com/ashishjangra27/face-mask-12k-images-dataset.

data augmentation is performed stretching, rotating and performing random crop (see Figure 2). One weakness of the data resides in the fact that all the images are more or less frontal picture, so we can loose accuracy in prediction when dealing with different perspectives.

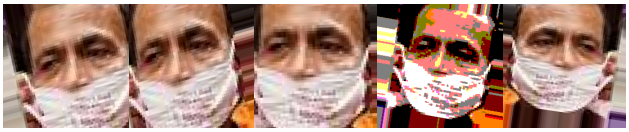


Figure 2: Examples of augmentation in train data with mask.

At the end our dataset is composed by:

- **Training set**:
  - 4011 with mask (815 originals + 3196 augmented)
  - 5401 without mask
- **Test set**:
  - 483 with mask
  - 509 without mask

The dataset described above is used to train the NN; in addition, the complete algorithm (face detector + face mask discriminator) is tested over some images and videos acquired from the web.

3. Method

As introduced in Section 1, our approach implies to split the problem in two tasks:

- Train and use a binary NN classificator, which can identify whether a face mask is used given a picture of a person's face;
- Extract Regions of Interest (ROIs) from generic images and then process them through the classifier. In our scenario, ROIs are represented by faces, so a face detector is implemented.

To tackle the first point, two solutions are proposed. First of all, a simple model with a convolutional architecture is built from scratch and trained over our dataset; in addition, also a transfer learning approach is tested. In this second case, a pretrained Network is adopted as base model, which was trained over a large dataset, and on top of that a fully connected part is constructed: in this way the abstract feature learned by the base model are exploited to solve our specific classification.

The face detection instead is carried out by taking advantage of a pretrained model available in the OpenCV library.

3.1. Simple Convolutional NN

The classification model is built using `Keras` [2] that is a frontend for Tensorflow, a machine learning environment developed by Google. Figure 9 describes the architecture of the Network. It is composed by three convolutional layers alternated by MaxPooling ones, while the final part flattens the result and propagates it through three fully connected layers; moreover, dropout is implemented both in the convolutional and in the fully connected section, in order to add some regularization during the learning phase. The last neuron features a Sigmoid activation function to perform the binary classification: in this way it is also possible to extract a probability estimate of the prediction. From now we refer to this model as *Convolutional Network*.

In addition to the preprocessing explained in Section 2, each image is resized to 224×224 pixels, and each RGB channel is normalized to 1. The appropriate loss for the task is the binary crossentropy, and the learning process is guided by the Adam optimizer; in addition, also a scheduler is implemented to control the learning rate in an exponential decreasing way. Training is carried out for 20 epoches, using batches of 128 images.

Even with this simple structure, results show good performances; a more detailed and accurate description, along with the comparison with the transfer learning approach, is presented in Section 4.

3.2. Transfer learning

Transfer learning is a technique where a Network pretrained over a large dataset is used as a base model in order to take advantage of its knowledge of low level features; on top of that another Network is built to achieve a specific task, in our case to recognize the presence or absence of a face mask.

MobileNetV2 [7] represents the state of the art architecture for mobile and resource constrained environments. In particular, it is trained over the ImageNet [3] dataset. Due to the complexity of the architecture, and the huge amount of images in the dataset, training is infeasible without proper hardware: hence the weights are downloaded from `Keras` and the architecture freezed, so they are not updated. The head model adapts MobileNetV2 to perform the binary classification task: an average pooling layer precedes two fully connected ones, with dropout. The final neuron activates with a Sigmoid function, as the previous convolutional model.

Learning is performed by the Adam optimizer, with an exponential decay scheduler. The duration is set to 20 epoches.

This solution is referred as *MobileNetV2* in the next Sections.

3.3. Face detector

In a real application it is very difficult to have only one person that stands in front of a camera waiting for a response of the program. It is more likely to have a situation where different people are moving into a room or in a public square and so the frame catched by the camera is more complex with respect to the ones used to train the prediction network. In such scenario there are several strategies that may work; we decide to implement a CV method to recognize and extract faces in a given frame. The Network used for this task is a Residual Network, in particular ResNet-10. This category of networks features "identity shortcut connections" that skips one or more layers. Network architecture and weights can be downloaded from `OpenCV` Github repository[2]. It is built with Caffe [4] and trained with the SSD framework [5] over the WIDER face dataset [8]. The usage of a pre-trained Network is due to the fact that great hardware resources and time are required to train a good detector. In addition, it should be more accurate with respect to `OpenCV` Haar Cascade when analyzing faces with different viewing angles and not only on straight on perspective.

This face detector requires that input images are preprocessed by resizing them to 300×300 pixels and acting mean subtraction in each RGB channel. Given a frame, the output is composed by the list of objects detected (represented by left, top, right, bottom margin with coordinates represented as fraction of the whole image) plus a confidence value, that can be used to tune the sensibility of the prediction.

Summing up, the face detector extracts faces from the images, which become the input for our Network built with `Keras`, that will predict whether a mask is worn.

4. Experiments and results

4.1. Neural Network classifier

We start analyzing the results of the training for our convolutional architecture and the one based on MobleNetV2. Figure 3 shows the accuracy (top) and the loss (bottom) for the two Networks, both in the training and in the test set. The two models display very good results, with an accuracy close to 1 (see Table 1) in the training and also in test set. This means that even a simpler model can accomplish our task with great performances. The reason can be that the dataset is simple to learn: in fact, as stated in Section 2, most images represent frontal pictures and hence the difference between having a mask or not is pretty detectable. In addition, it can be noticed that the results on the test set are better than on the training one; this is unusual, and it may be that there is no great difference between images in training and test.

[2]https://github.com/opencv/opencv/tree/master/samples/dnn/face_detector

At the light of these considerations, the better results achieved by MobileNetV2 need to be taken with a grain of salt. In fact, it is possible that this NN is overfitting our data, while the simpler ones may have better generalization properties, even if the test accuracy is slightly lower. In Sections 4.3 and 4.4 some experiments over real images and video are presented, and more detailed observations are proposed.

Table 1: Accuracy reached by the two NN models.

Model	Dataset	Accuracy
MobileNetV2	Train	0.9946
	Test	0.9922
Convolutional Network	Train	0.9621
	Test	0.9788

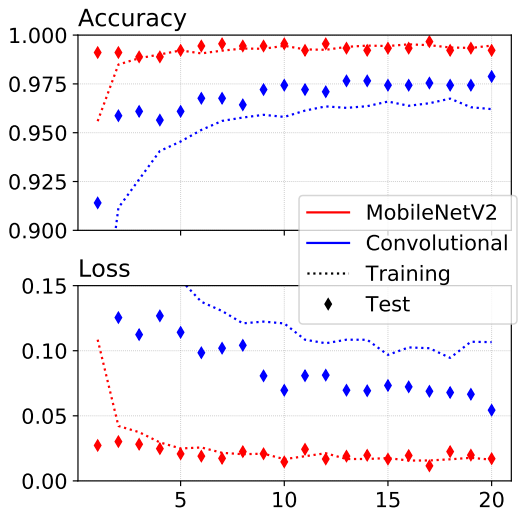


Figure 3: Accuracy (top) and loss (bottom) of the Networks as function of the training epoch.

4.2. Face detector

`OpenCV` does not provide metrics to evaluate the performances of ResNet-10 face detector; however we can still make some critical analysis on this model. We have that each object identified by the detector is associated to the *confidence* parameter. We can set a minimum threshold for this parameter between 0 and 1, in order to filter weak face detections. A value near to 0 results in extracting ROIs even if no real faces are present, while if too high we can miss information. An example is shown in Figure 5, where we have a comparison with a confidence of 0.15 and 0.75. The higher value of 0.75 (bottom one) does not allow to catch

all faces in the picture, while with 0.15 (top 2 images) we are able to extract more information.

4.3. Predictions over real images

To better understand our models, some tests over images and videos are performed. This allows to better compare the two NN models trained for our classification task. Figure 4 shows a quite simple scenario, where three people are walking in a public place with a straight on perspective. As we can see the face recognition is able to catch all the complete faces present in the pictures and we do not have a great distinction between the models.

(a) Convolutional Network and confidence at 0.75

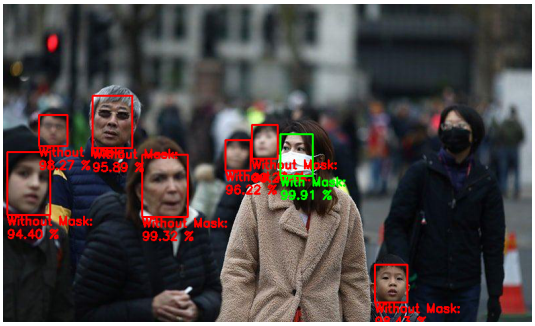
(b) MobileNetV2 Network and confidence at 0.75

Figure 4

Figure 5 is a more complex situation, which allows to notice some differences between the Networks and the importance of the confidence parameter. First of all we have to tune the confidence of the face detector to be able to catch all ROIs in the frame. In particular, we see some difficulties to detect people in the background or when faces are small and blurred and not looking straight on camera; in addition, also the fact of wearing a face mask makes the task more difficult: this is not surprising, since the face detector was trained over faces without masks. This is clearly noticeable in 5c. Moreover, some misclassifications emerge (see 5a). This uncertainty with high probability is attributable to the train set, combined with the complexity of MobileNetworkV2. Furthermore, the simpler Convolutional architecture seems to act better when faces are blurred and in the background.

An additional evidence appears also in Figure 6, where the more complex model fails to correctly classify the woman on the left side.

(a) MobileNetV2 Network and confidence at 0.15



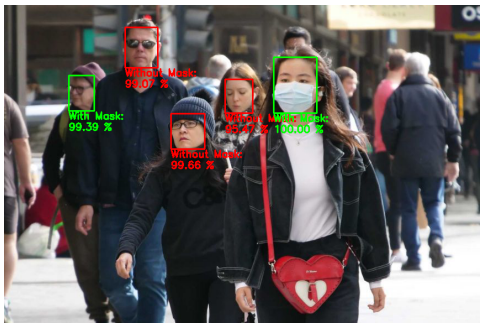
(b) Convolutional Network and confidence at 0.15

(c) Convolutional Network and confidence at 0.75

Figure 5: Comparison between the two Networks and different levels of confidence.

In conclusion, the analyses over real images suggests that MobileNetV2 may have overfitted the data during the training phase, while the simpler model holds better generalization capabilities. More in detailed, greater differences happen when faces are not clear and in foreground, or when people are not looking straight into the camera. As far as it concerns the face detector, the confidence threshold needs to be tune for each specific case, keeping in mind that we are trying to extract faces even if they are partially covered by the mask, so we need to allow low levels of the parame-

(a) Convolutional Network and confidence at 0.25



(b) MobileNetV2 Network and confidence at 0.25

Figure 6

ter.

4.4. Video analysis

Predictions can also be made during video or real time streamings. This is a more challenging task, but it helps to gain a better knowledge of the models.

One first study involves a video taken from *TG3*, an Italian newscast, and a parallel analysis of MobileNetV2 and the Convolutional Network is proposed. Figure 7 show some snapshots taken from the video analysis. It sums up the three different major situations that in general happen:

- If there are people looking in a frontal position with respect the the camera, both the models are able to get the right prediction (Picture 7a);
- In 7b the face detector cannot extract ROIs if the face if largely covered, so no prediction is performed at all, sice nothing is passed to the classificator;
- The last case, the most probable in real life scenarios, applies to people looking away from the camera or standing quite far away from it. In this case, as shown in Figure 7c, the Convolutional Network seems to be more accurate and a lot more stable than the one based on MobileNetV2.

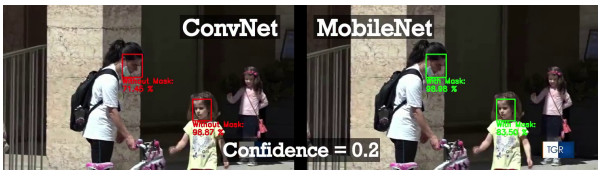
The video analysis confirms the hypothesis raised in Section 4.3: the reliability of Convolutional Network is higher than the one of MobileNetV2 when applying the predictions to real life situations.

Also our streaming test endorse this assumption. In Figure 8 two snapshots are taken from a real time record from the PC webcam. MobileNetV2 is fooled by a subject having a beard: it is said to wear a mask even if it is not the case.

Another aspect that emerges from the videos is that MobileNetV2 changes rapidly the prediction and seems to be very unstable, while the Convolutional Network keeps the same response for a longer duration.

(a)

(b)



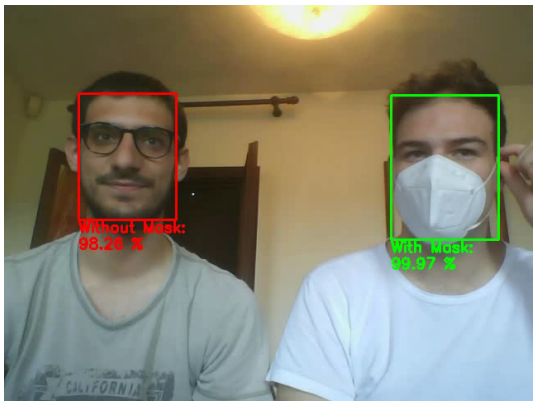
(c)

Figure 7: Snapshot taken from *TG3* newscast video.

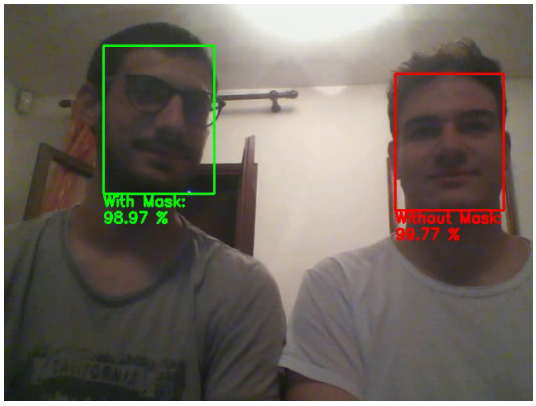
5. Conclusions and future developments

In this work a pipeline to detect whether people are wearing face masks is presented. First of all a face detector extract ROIs from images or videos, and then a NN makes the prediction. Two different predictors are tested, which feature architecture with dissimilar complexity. Experiments over real images and videos are performed to better understand the capabilities of the models.

Metrics extracted during the training of NNs leads to consider MobileNetV2 as the best candidate, with an accuracy of about 99% against the 97% of the Convolutional solution. Delving into the details, we discover that the test set that we use may be quite similar to the training one and so the results is a little bit biased (due to overfitting). Looking for more complex tests such as real pictures, videos and streaming, we discover that the Convolutional Network keeps a better stability in the previsions and produces more reliable forecasts;on the other hand, MobileNetV2 seems to



(a) Convolutional Network



(b) MobileNetV2

Figure 8: Snapshot taken from a real time streaming analysis.

flare during videos and it fails when the inputs are different from the ones given in the training phase (e.g. when faces are not straight to the camera, or are more blurred). The reason behind this behavior can be found in the complexity of the model itself. We have that MobileNetV2 is a very deep network that can be suitable for very complex tasks, but that can leads to overfitting when facing simpler scenarios like the one presented here. At the light of this discussion we can conclude that the Convolutional Network is to be preferred for our scope.

As far as it concerns the face detector, it needs to be pointed out that it is trained over a dataset that features faces without masks on them (even if the WIDER dataset collects situations with high degree of variability in scale, pose and occusion). This could lead to not catching ROIs when face masks are worn, since a great part is covered and hence it may not be detected; if this happens, no prediction is made, hence important information may be missed. To overcome this problem, the confidence parameter needs to be tune for each situation.

The bottleneck of our work is definitely the training set. Collecting a more heterogeneous series of images will surely boost the classification performances, allowing more complex models to be use effectively. For example, taking pictures and manually annotating the positions of the faces may be required to build data that features a vastness of different situations in terms of face positions, light, types of masks, ...

However, in this case the bottleneck may only be shift towards the performances of the face detector. As already stated, it may have difficulties to make good detections if the face is covered. Hence, directly training the face detector to recognize not only faces, but also faces with masks can be the better solution. Also in this case a large dataset annotated by hand is required.

6. Appendix

6.1. Graph

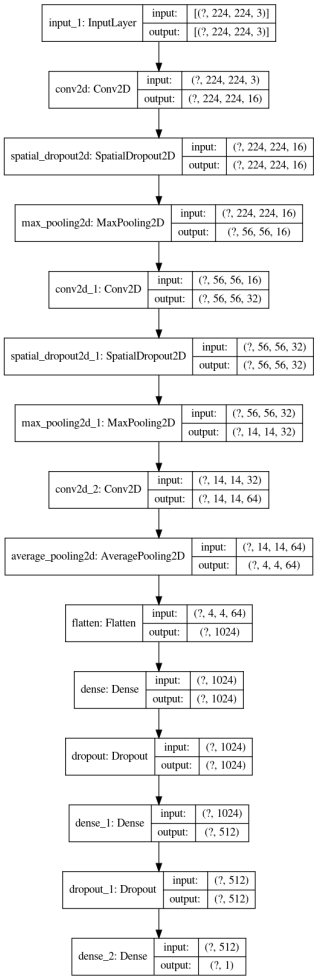


Figure 9: Schematic representation of the Convolutional Neural Network.

References

[1] Prajna Bhandary. Mask classifier. https://www.linkedin.com/feed/update/urn%3Ali%3Aactivity%3A6655711815361761280/.

[2] François Chollet et al. Keras. https://keras.io, 2015.

[3] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

[4] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.

[5] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. SSD: Single shot multibox detector. In *ECCV*, 2016.

[6] Adrian Rosebrock. Covid-19: Face mask detector with opencv, keras/tensorflow, and deep learning. https://www.pyimagesearch.com/2020/05/04/.

[7] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks, 2018.

[8] Shuo Yang, Ping Luo, Chen Change Loy, and Xiaoou Tang. Wider face: A face detection benchmark. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.