Face Mask Detector: a Computer Vision Application

Federico Agostini
Federico Bottaro

Introduction

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.

Artificial Intelligence (AI) along with Computer Vision (CV) can be exploited to enhance and automate the process of controlling the respect of the basic social rules imposed by the government all around the world.

Introduction

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:

- 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 face from images and video, and then process them using the NN developed in the previous step.

Supervised learning paradigm to train Neural Network

Large amount of labelled data with faces with and without mask

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.

- A first solution suggested by Prajna Bhandary consist in artificially build our data by placing images of face masks on top of existing face pictures.
- The lack of generalization capabilities can drive to a bad performance when applied in different situation
- The use of only two type of mask is not sufficient to identify such an object present in the market with various colors and shape.

Dataset

The actual dataset we decide to use is hosted by *Kaggle*.

Train
Validation
Test

} Train

→

Due to the reduced amount of data with people wearing a mask, we join train and validation but even in this way we have obout 800 images with the face mask

The solution we implement is to make standard data augmentation (stretching, rotating and performing random crop).

Training:
- 4011 with mask
- 5401 without mask.

Test:
- 483 with mask
- 509 without mask.

Method

Task #1: classification

Model 1: Convolutional Network

- Convolution Network build with Keras.
- The convolutional layers are alternated by MaxPooling ones
- Final part is composed of fully connected layers
- Dropout to regularize the learning.
- The last neurons give the prediction as output through sigmoid function.

Training parameters:

- Image resized to 224x224 pixels with RGB channel normalized to 1.
- Loss: binary crossentropy.
- Optimizer: Adam.
- Learning rate: driven by a scheduler to act an exponential decrease of its value.
- Training time: 20 epochs using batches of 128 images.

Method

Task #1: classification

Model 2: MobileNetV2

- Transfer learning
- MobileNetV2 represents the state of the art architecture for mobile and resource constrained environments. In particular, it is trained over the ImageNet dataset an it is able to classify object from different classes.
- Head model added at the top of MobileNet: Average Pooling layer precedes two fully connected ones, with dropout. Again, the prediction was extracted from a Sigmoid.
- Training parameters follow the previous ones

Method

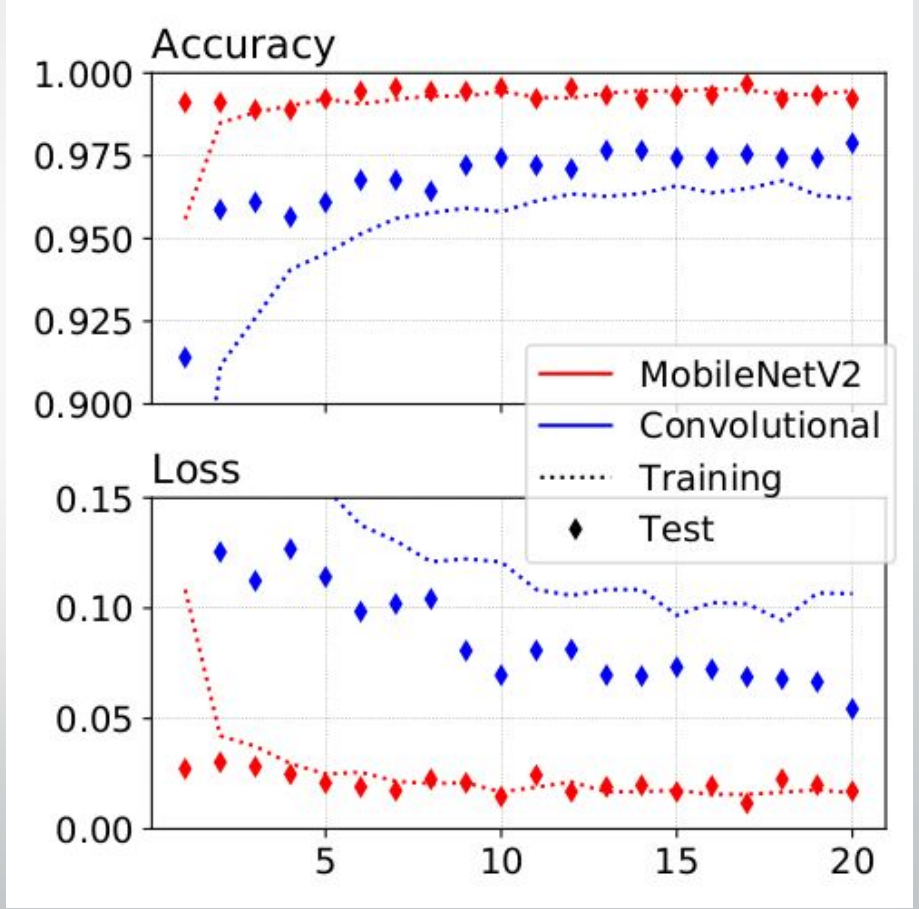
Task #2: Face Detection

- Why use a face detector? Need to extract faces from images to make predictions.
- The face detection is carried out using a pretrained model available in the OpenCV library.
- The network used for this task is a Residual Network, in particular ResNet-10. It is built with Caffe and trained with the SSD framework over the WIDER face dataset. This network is to be prefered to OpenCV Haar Cascade when analyzing faces with different viewing angles and not only on straight on perspective.
- As preprocessing here we need to pass to this Network 300x300 pixels images and acting mean subtraction in each RGB channel according to the train procedure.

Experiment

Neural Network: classifier

- Accuracy and loss
- Results on the test set are better than on the training one; this is unusual.
- No great difference between images in training and test.
- The better results achieved by MobileNetV2 need to be taken with a grain of salt.



Experiment

Neural Network: photo analysis

- To analyze the performance of the Networks we test on some complex images.

- We can better understand the behavior of the predictors.

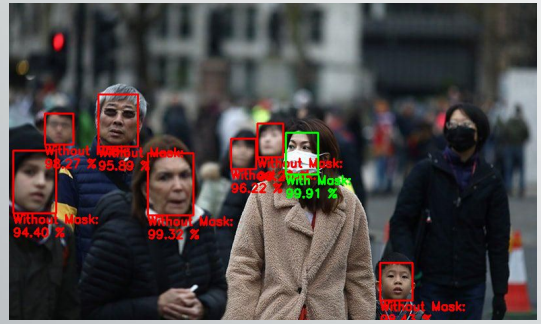
MobileNetV2

Convolutional Network

Experiment

Neural Network: photo analysis

- In crowded images emerge the importance of *confidence* parameter
- MobileNetV2 fails when faces are not straight on or autofocused



Convolutional Network with a confidence of 0.15

MobileNetV2 with a confidence of 0.15

Convolutional Network with a confidence of 0.75

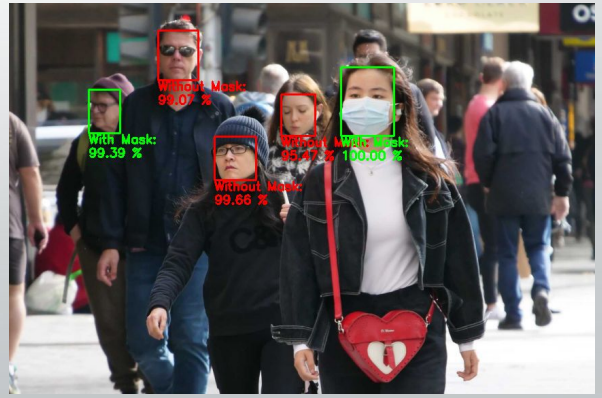
Experiment

Neural Network: photo analysis

- Also here MobileNetV2 performs worse as far it concerns the woman on the left
- This misunderstanding may be researched in the simplicity of the training set.

Convolutional Network

MobileNetV2



Experiment

Neural Network: Video analysis

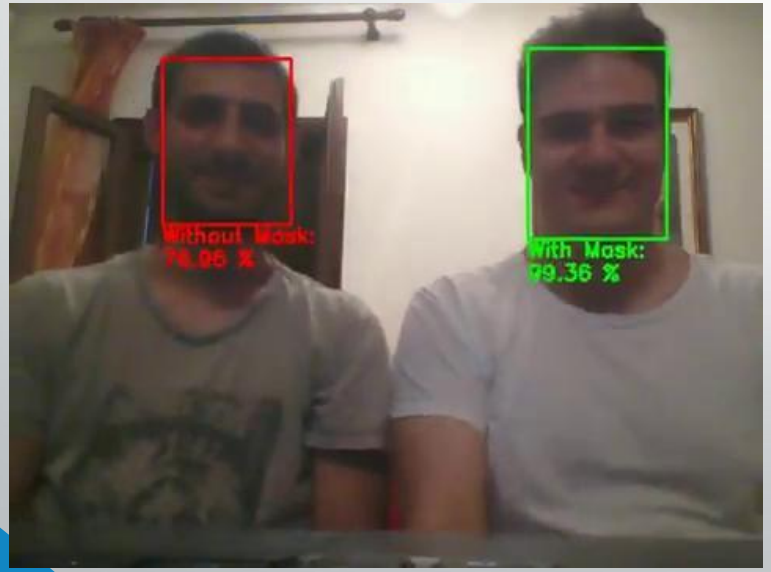
- Less robustness in video analysis.
- MobileNetV2 seems to have a continuous flaring in the output
- Convolutional Network is more stable in this field.
- Faces with occlusion still have a problem in the recognition and we can't make any prediction.

Experiment

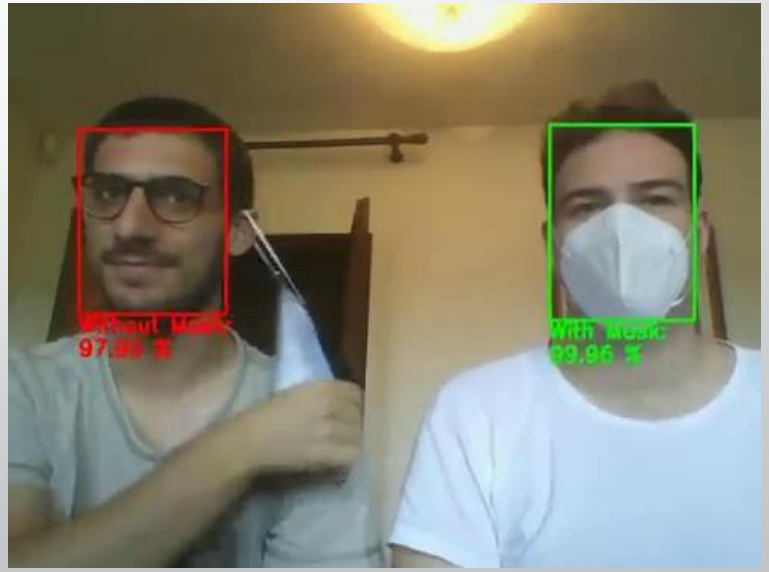
Neural Network: Streaming analysis

- Streaming from the webcam of the PC.
- The procedure are the same used for the video analysis but now it is in real time.

MobileNetV2



Convolutional Network



Conclusion

- Accuracy and the loss of the models in the test and training set figure out a strange behavior that can hide some overfitting.
- The reason may lie in the simplicity of the dataset used. In particular the test set is composed by images quite similar to the ones in the train set.
- The complexity of MobileNetV2 may be inappropriate to a simple task like a binary classification.
- The most suitable analysis are performed on video and moving object.
- MobileNetV2 present an unstable prediction with a lot of flarings in terms of classification while the Convolutional model produce has a more solid output.
- Importance of face detector confidence parameter, as first step of our pipeline

Future development

- Bottleneck: dataset
  - more images in different scenarios
  - include more pixels around the faces
  - taking faces from images, adding annotations by hand

- Bottleneck shifts to face detector
  - Directly train the face detector to recognize also faces with masks
  - Necessity of dataset with annotation
  - Information on where the face are located in the frame
  - Information on the presence of the mask

Thank you for the attention