Covid-19 Face Mask Detection Using TensorFlow, Keras and OpenCV

2020

49873.2020.9342585

*Abstract*—COVID-19 pandemic has rapidly affected our day-to-day life disrupting the world trade and movements. Wearing a protective face mask has become a new normal. In the near future, many public service providers will ask the customers to wear masks correctly to avail of their services. Therefore, face mask detection has become a crucial task to help global society. This paper presents a simplified approach to achieve this purpose using some basic Machine Learning packages like TensorFlow, Keras, OpenCV and Scikit-Learn. The proposed method detects the face from the image correctly and then identifies if it has a mask on it or not. As a surveillance task performer, it can also detect a face along with a mask in motion. The method attains accuracy up to 95.77% and 94.58% respectively on two different datasets. We explore optimized values of parameters using the Sequential Convolutional Neural Network model to detect the presence of masks correctly without causing over-fitting.

1. INTRODUCTION

According to the World Health Organization (WHO)’s official Situation Report – 205, coronavirus disease 2019 (COVID-19) has globally infected over 20 million people causing over 0.7million deaths. Individuals with COVID-19 have had a wide scope of symptoms reported – going from mellow manifestations to serious illness. Respiratory problems like shortness of breath or difficulty in breathing is one of them. Elder people having lung disease can possess serious complications from COVID-19 illness as they appear to be at higher risk. Some common human coronaviruses that infect public around the world. Persons having respiratory problems can expose anyone (who is in close contact with them) to infective beads. Surroundings of a tainted individual can cause contact transmission as droplets carrying virus may withal arrive on his adjacent surfaces.

To curb certain respiratory viral ailments, including COVID-19, wearing a clinical mask is very necessary. The public should be aware of whether to put on the mask for source control or aversion of COVID-19. Potential points of interest of the utilization of masks lie in reducing vulnerability of risk from a noxious individual during the “pre-symptomatic” period and stigmatization of discrete persons putting on masks to restraint the spread of virus. WHO stresses on prioritizing medical masks and respirators for health care assistants? Therefore, face mask detection has become a crucial task in present global society.

Face mask detection involves in detecting the location of the face and then determining whether it has a mask on it or not. The issue is proximately cognate to general object detection to detect the classes of objects. Face identification categorically deals with distinguishing a specific group of entities i.e., Face. It has numerous applications, such as autonomous driving, education, surveillance, and so on. This paper presents a simplified approach to serve the above purpose using the basic Machine Learning (ML) packages such as TensorFlow, Keras, OpenCV and Scikit-Learn.

1. RELATED WORK

In face detection method, a face is detected from an image that has several attributes in it. According to research, face detection requires expression recognition, face tracking, and pose estimation. Given a solitary image, the challenge is to identify the face from the picture. Face detection is a difficult errand because the faces change in size, shape, color, etc. and they are not immutable. It becomes a laborious job for opaque image impeded by some other thing not confronting camera, and so forth. I think occlusive face detection comes with two major challenges:

1) unavailability of sizably voluminous datasets containing both masked and unmasked faces.

2) exclusion of facial expression in the covered area.

Utilizing the Locally Linear Embedding (LLE) algorithm and the dictionaries trained on an immensely colossal pool of masked faces, synthesized mundane faces, several mislaid expressions can be recuperated and the ascendancy of facial cues can be mitigated to great extent. According to the work report, ***Convolutional Neural Network*** (CNNs) in computer vision comes with a strict constraint regarding the size of the input image. The prevalent practice reconfigures the images before fitting them into the network to surmount the inhibition.

Here the main challenge of the task is to detect the face from the image correctly and then identify if it has a mask on it or not. In order to perform surveillance tasks, the proposed method should also detect a face along with a mask in motion.

1. DATASET

This dataset consists of 4095 images belonging to two classes:

* With Mask: 2,165 images
* Without Mask: 1,930 images

4. INCORPORATED PACKAGES

# A. TensorFlow

TensorFlow, an interface for expressing machine learning algorithms, is utilized for implementing ML systems into fabrication over a bunch of areas of computer science, including sentiment analysis, voice recognition, geographic information extraction, computer vision, text summarization, information retrieval, computational drug discovery and flaw detection to pursue research. In the proposed model, the whole Sequential CNN architecture (consists of several layers) uses TensorFlow at backend. It is also used to reshape the data (image) in the data processing.

# B. Keras

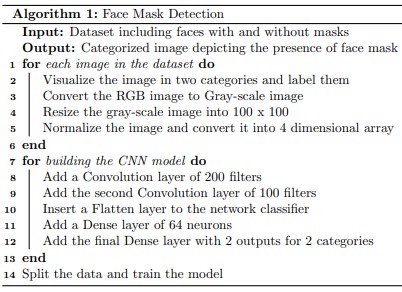
Keras gives fundamental reflections and building units for creation and transportation of ML arrangements with high iteration velocity. It takes full advantage of the scalability and cross-platform capabilities of TensorFlow. The core data structures of Keras are layers and models. All the layers used in the CNN model are implemented using Keras. Along with the conversion of the class vector to the binary class matrix in data processing, it helps to compile the overall model.

# OpenCV

OpenCV (Open-Source Computer Vision Library), an opensource computer vision and ML software library, is utilized to differentiate and recognize faces, recognize objects, group movements in recordings, trace progressive modules, follow eye gesture, track camera actions, expel red eyes from pictures taken utilizing flash, find comparative pictures from an image database, perceive landscape and set up markers to overlay it with increased reality and so forth. The proposed method makes use of these features of OpenCV in resizing and color conversion of data images.

5. THE PROPOSED METHOD

The proposed method consists of a cascade classifier and a pre-trained CNN which contains two 2D convolution layers connected to layers of dense neurons. The algorithm for face mask detection is as follows:

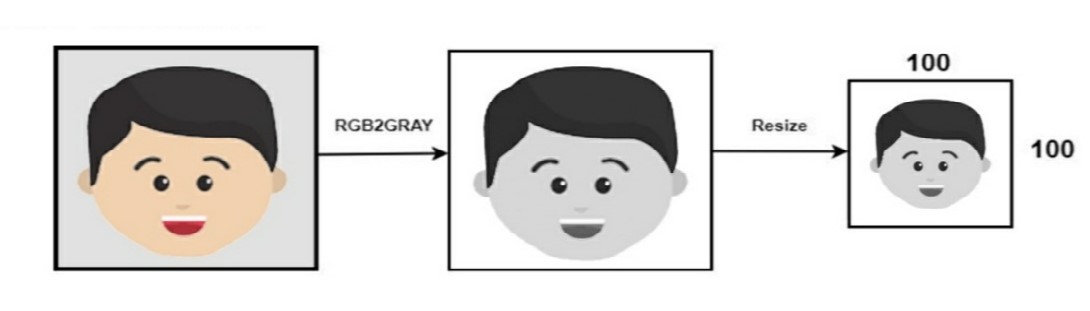


# A. Data Processing

Data preprocessing involves conversion of data from a given format to much more user friendly, desired and meaningful format. It can be in any form like tables, images, videos, graphs, etc. This organized information fit in with an information model or composition and captures relationship between different entities. The proposed method deals with image and video data using NumPy and OpenCV.

*a) Data Visualization:* Data visualization is the process of transforming abstract data to meaningful representations using knowledge communication and insight discovery through encodings. It is helpful to study a particular pattern in the dataset.

1. *Conversion of RGB image to gray image:* Modern descriptor-based image recognition systems regularly work on grayscale images, without elaborating the method used to convert from color-to-grayscale. This is because the color-to-grayscale method is of little consequence when using robust descriptors. Introducing nonessential information could increase the size of training data required to achieve good performance. As grayscale rationalizes the algorithm and diminishes the computational requisites, it is utilized for extracting descriptors instead of working on color images instantaneously.



We use the function *cv2.cvtColor(input image, flag)* for changing the color space. Here flag determines the type of conversion [9]. In this case, the flag *cv2.COLOR BGR2GRAY* is used for gray conversion.

Deep CNNs require a fixed-size input image. Therefore, we need a fixed common size for all the images in the dataset. Using *cv2.resize()* the gray scale image is resized into 224 x 224.

# B. FaceNet or Mask Net

a) FaceNet is a face recognition pipeline that learns mapping from faces to a position in a multidimensional space where the distance between points directly correspond to a measure of face similarity.

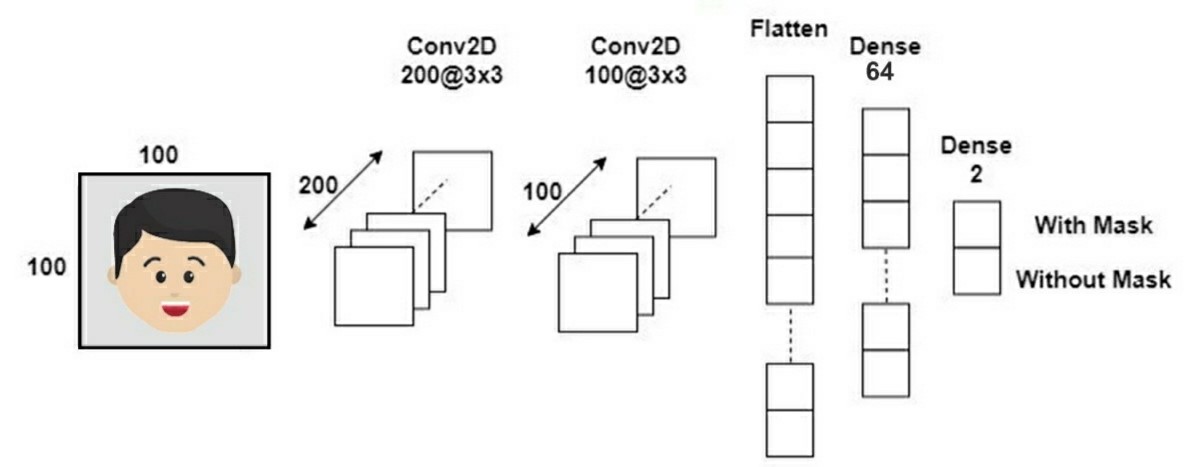
b) In Conclusion, **MaskNet** proves to be an efficient method to deal with partial point cloud registration when used with any existing registration algorithm. An interesting property of this pipeline is that both the **MaskNet** and the registration algorithm are independent of each other and can be trained separately.

# C. Training of Model

*a) Building the model using CNN architecture:* CNN has become ascendant in miscellaneous computer vision tasks. The current method makes use of Sequential CNN.

The First Convolution layer is followed by Rectified Linear Unit (ReLU) and MaxPooling layers. The Convolution layer learns from 200 filters. Kernel size is set to 3 x 3 which specifies the height and width of the 2D convolution window. As the model should be aware of the shape of the input expected, the first layer in the model needs to be provided with information about input shape. Following layers can perform instinctive shape reckoning [13]. In this case, *input shape* is specified as *data.shape[1:]* which returns the dimensions of the data array from index 1. Default padding is “valid” where the spatial dimensions are sanctioned to truncate and the input volume is non-zero padded. The activation parameter to the Conv2D class is set as “relu”. It represents an approximately linear function that possesses all the assets of linear models that can easily be optimized with gradient-descent methods. Considering the performance and generalization in deep learning, it is better compared to other activation functions. Max Pooling is used to reduce the spatial dimensions of the output volume. Pool size is set to 3 x 3 and the resulting output has a shape (number of rows or columns) of: shape of output = (input shape - pool size + 1) / strides), where strides have default value (1,1).

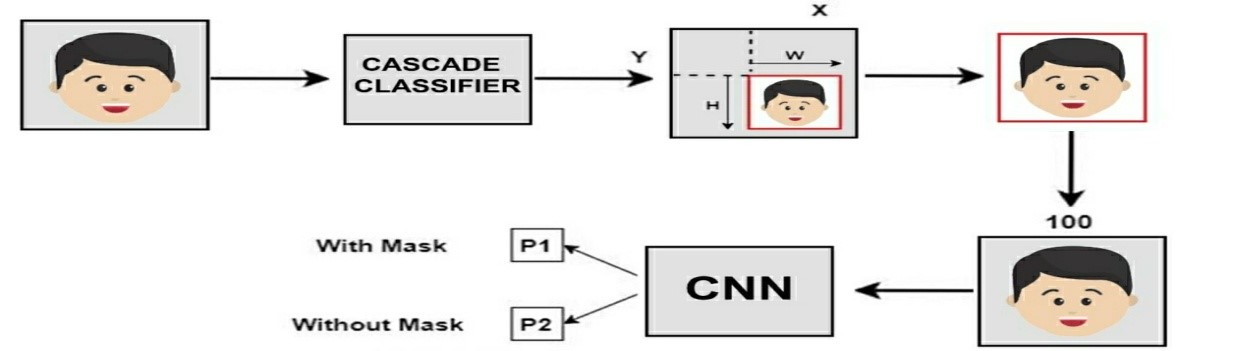
The second Convolution layer has 100 filters and Kernel size is set to 3 x 3. It is followed by ReLu and MaxPooling layers. To insert the data into CNN, the long vector of input is passed through a Flatten layer which transforms matrix of features into a vector that can be fed into a fully connected neural network classifier. To reduce overfitting a Dropout layer with a 50% chance of setting inputs to zero is added to the model. Then a Dense layer of 64 neurons with a ReLu activation function is added. The final layer (Dense) with two outputs for two categories uses the Softmax activation function.



Convolutional Neural Network architecture

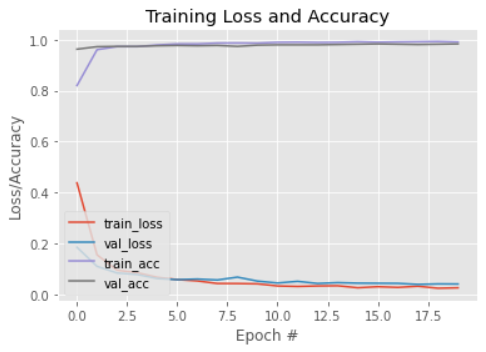
The learning process needs to be configured first with the compile method. Here, optimizer is used. *categorical crossentropy* which is also known as multiclass log loss is used as a loss function (the objective that the model tries to minimize). As the problem is a classification problem, metrics is set to “accuracy”.

*b) Splitting the data and training the CNN model:* After setting the blueprint to analyze the data, the model needs to be trained using a specific dataset and then to be tested against a different dataset. A proper model and optimized *train test split* help to produce accurate results while making a prediction. The test size is set to 0.1 i.e., 90% data of the dataset undergoes training and the rest 10% goes for testing purposes. The validation loss is monitored using *ModelCheckpoints*. Next, the images in the training set and the test set are fitted to the Sequential model. Here, 20% of the training data is used as validation data. The model is trained for 20 epochs (iterations) which maintains a trade-off between accuracy and chances of overfitting.

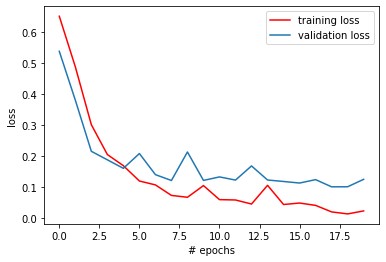


6. RESULT AND ANALYSIS

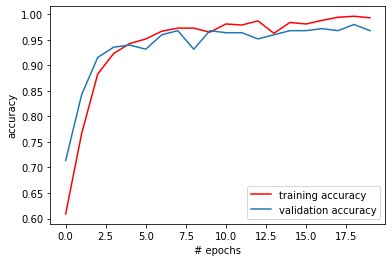
The model is trained, validated and tested upon two datasets. Corresponding to dataset 1, the method attains accuracy up to 95.77% depicts how this optimized accuracy mitigates the cost of error. Dataset 2 is more versatile than dataset 1 as it has multiple faces in the frame and different types of masks having different colors as well. Therefore, the model attains an accuracy of 94.58% on dataset 2 as depicts the contrast between training and validation loss corresponding to dataset 2. One of the main reasons behind achieving this accuracy lies in *MaxPooling*. It provides rudimentary translation invariance to the internal representation along with the reduction in the number of parameters the model has to learn. This sample-based discretization process down-samples the input representation consisting of image, by reducing its dimensionality. Number of neurons has the optimized value of 64 which is not too high. A much higher number of neurons and filters can lead to worse performance. The optimized filter values and pool size help to filter out the main portion (face) of the image to detect the existence of mask correctly without causing over-fitting.



#Epoch Training Loss and Accuracy

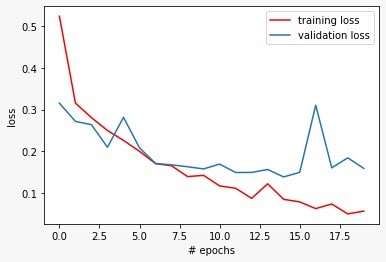


# Epochs vs loss corresponding to dataset 1

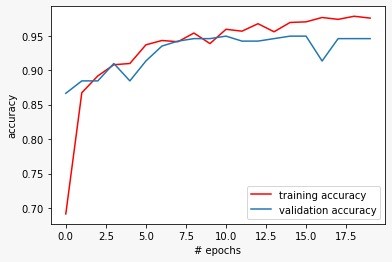


# Epochs vs accuracy corresponding to dataset 1

The system can efficiently detect partially occluded faces either with a mask or hair or hand. It considers the occlusion degree of four regions – nose, mouth, chin and eye to differentiate between annotated mask or face covered by hand. Therefore, a mask covering the face fully including nose and chin will only be treated as “with mask” by the model.



# Epochs vs loss corresponding to dataset 2



The main challenges faced by the method mainly comprise of varying angles and lack of clarity. Indistinct moving faces in the video stream make it more difficult. However, following the trajectories of several frames of the video helps to create a better decision – “with mask” or “without mask”.

7. CONCLUSIONS

In this paper, we briefly explained the motivation of the work at first. Then, we illustrated the learning and performance task of the model. Using basic ML tools and simplified techniques the method has achieved reasonably high accuracy. It can be used for a variety of applications. Wearing a mask may be obligatory in the near future, considering the Covid-19 crisis. Many public service providers will ask the customers to wear masks correctly to avail of their services. The deployed model will contribute immensely to the public health care system. In future it can be extended to detect if a person is wearing the mask properly or not. The model can be further improved to detect if the mask is virus prone or not i.e., the type of the mask is surgical, N95 or not.