FACE MASK DETECTION

Puttaraja - 191IT139
Information Technology
National Institute of Technology Karnataka
Surathkal, India 575025
Email: puttaraja.191it139@nitk.edu.in

Rakesh Kallimani - 191IT244
Information Technology
National Institute of Technology Karnataka
Surathkal, India 575025
Email: rakeshk.191it244@nitk.edu.in

***Abstract*—In the current situation of COVID-19 - an airborne infectious disease - face mask has become an integral part of our lives. Many health organizations suggests that wearing a face mask is one of the best preventive measures against COVID-19 since it eliminates cross-contamination. In this paper we have proposed a face mask detection model built using Tensorflow along with real time detection using OpenCV. This model can classify images into two which are 'with mask' and 'without mask'. The proposed model has its initial layers taken from MobileNet-v2 which is a light weight image classification model upon which a few layers are added on to produce the final model. The data-set for the model has two categories which are 'with mask' and 'without mask' which totals to 3706 images . The real time detection was done by feature extraction from the region of interest (RIO) of an image or a frame from the video and classified it into one of the above mentioned categories. This was done using functions and libraries under Tensorflow and OpenCV This model has achieved a training accuracy of 98.79and validation accuracy of 99.10**

I. INTRODUCTION

COVID-19 pandemic has become a rise of concern for the human community these days. To prevent the spread of this disease, an effective and protective measure is to use a face mask in public areas. As a result ,there is a need to build the face mask detector which will ensure whether a person is wearing a mask or not. During the COVID-19 pandemic, face masks have been used as a public as well as personal health control measure against the transmission of corona virus. Their use is intended as personal protection to prevent infection and as source control to limit transmission of the virus in a community or health care setting. But most people are neglecting and not following this strictly. So, it becomes important for us to ensure people are wearing face masks.
The model make sures a person is wearing a mask or not and uses transfer learning and python scripts to train the model.

This project aims to build a face mask detector using concepts of Convolutional Neural Networks (CNN) and OpenCV so that we can ensure public safety guidelines such as wearing face masks are followed. Keeping this project as simple and versatile as possible so that, later it can be integrated with embedded systems for applications in airports, railway stations and other public places to keep a check on people.

II. LITERATURE SURVEY

Generally, most of the publications focuses on whether people are wearing a face mask or not.

Authors	Methodology	Merits	Limitations	Dataset Used
M. Khalifa et al.	Transfer learning Using ResNet and SVM	Works well on most datasets	Try most of classical machine learning methods to get lowest consume time and highest accuracy.	SMFD, LFW
C. Li et al.	YOLOv3 algorithm and Darknet-53	Have balanced the speed accuracy trade off very well	Can introduce more complex models for better and complex feature extraction.	CelebA, WIDER FACE
B. Qin et al.	SRCNet	Have included the third category of 'incorrect facemask wearing	Does not perform well on low quality images.	FDDB
Shaik et al.	Deep learning and VGG-16	Classifies seven facial expressions and quite accurate	KDEF cannot be used in cross-cultural studies and is gender specific	KDEF
Kavitha et al.	Viola-Jones algorithm GMM	Works well on low dimensional face datasets	False detection due to face orientation.	AR face, MAFA

Fig. 1: Literature Review

In [1] the authors proposed a transfer learning model using ResNet and used standard machine learning algorithms for classification. They achieved an accuracy 99.49using SVM classification model for classification.

In [2], the authors used the YOLOv3 algorithm for face detection. YOLOv3 uses Darknet-53 as the backbone. The proposed method achieved 93.9on CelebA and WIDER FACE dataset . The testing was the FDDB dataset.

B Qin et al [3] established SRCNet and classified the mask-wearing situation into three categories: no facemaskwearing, incorrect facemask-wearing, and correct facemaskwearing and the proposed SRCNet achieved 98.70

Shaik et al [4] used deep learning real-time face emotion classification and recognition. They used VGG-16 to classify seven facial expressions. The proposed model was trained on the KDEF dataset and achieved 88

In [5] Kavitha et al proposed a Gaussian Mixture Model (GMM) which is applied for foreground detection to extract the region of interest (ROI).The accuracy obtained from this model was 97.50

III. PROBLEM STATEMENT

Wearing a face mask has become mandatory but unfortunately most of the people are not following it. For safety purposes there should be a system which ensures that everyone is wearing a mask or not. The above problem was solved by using Tensorflow and OpenCV to build a face mask detector.

A. *Objectives*

- Build face mask detector : to build a face mask detector which is as accurate as possible.
- To check if people are wearing mask : to check whether people are wearing face masks as it stops the micro droplets expelled from cough or sneeze thus eliminating cross contamination.
- To check if people are wearing mask according to WHO standards : Many people although wearing a mask may not be wearing it properly which may lead to the further spread of the disease.

IV. METHODOLOGY

The methodology includes three phases.

First creating the dataset for training our model.
Second building the face mask detection model with the help of Tensorflow/Keras.
Third real time detection using OpenCV.

A. *Dataset creation / characteristics :*

This project was conducted on a dataset consisting of 3706 images which includes two categories of 'with mask'(1788) and 'without mask'(1918) and it is still expanding. A large part of the dataset for the 'with mask' category was created manually by pasting a face mask on the 'without mask' category images. This was done by python scripts using OpenCV. For this Haar Cascade algorithm was adopted which is basically a machine learning object detection algorithm which is used to identify objects in an image or video.

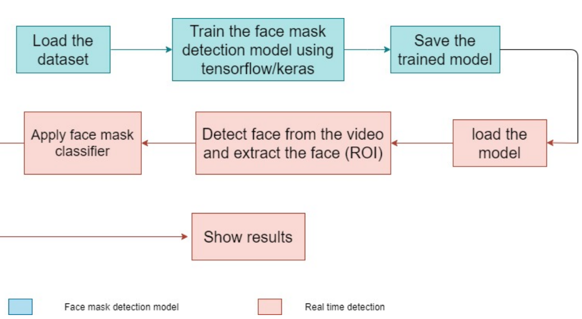

Fig. 2: Flow chart

Using the above algorithm we detect the face in the given image and compute the bounding box location of the face in the image. Once we know where in the image the face is, we extract the Face region of Interest(ROI) using Numpy slicing and from there, we apply facial landmarks which help us to localize the eyes , nose , mouth ,chin etc. Next an image of a mask with a transparent background is loaded. The script automatically applies the mask to the face using the facial landmarks. Hence creating a images with face mask. This process is repeated for all the input images, thereby creating the artificial data set.In the training section of our model we have data augmentation were we flip , resize the image in all possible ways so that our model gets trained well. Fig. 2 shows images of the data set.

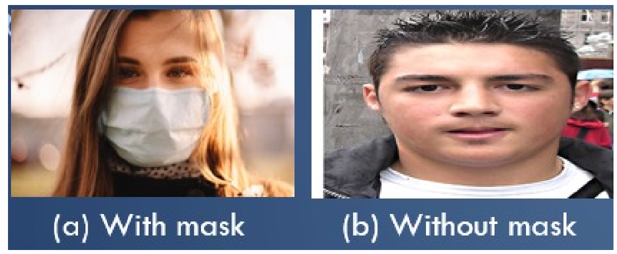

Fig. 3: Sample images from the dataset

B. *Model building :*

The model includes two parts, the base model and the head model. The base model is deep transfer learning neural network (MobileNet-v2) used as feature extractor and head model a customized model which is placed on top of the base model for better classification.

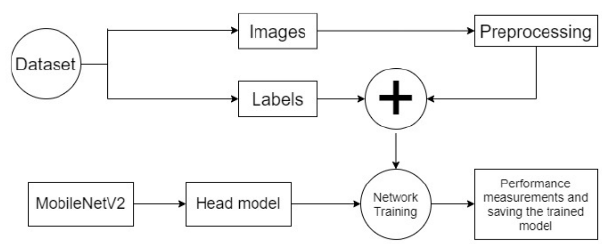

Fig. 4: Proposed model structure

MobileNet-v2 is a light weight convolutional neural network that has 53 layers. It contains initial fully convolution layer with 32 filters, followed by 19 residual bottleneck layers. MobileNet-v2 is based on inverted residual structure where the connections are between the bottleneck layers. The intermediate expansion layer uses lightweight depthwise convolutions to filter features as a source of non-linearity. The final layers are removed from MobileNet-v2 and the head model is added on top of this for classification.
The head model consists of 5 layers which are, AveragePooling2D, Flatten, Dense, Dropout, Dense (output layer with softmax as activation function). This head model is added on top of the base model to produce the final model that we are going to train and classify images accordingly.

C. *Real time detection and testing :*

1) Testing of static images: Consists of three main parts
• Load an input image from the disk

• Detect faces in the images
• Apply our face mask detector to classify the face as 'with mask' or 'without mask'.

The python script requires three TensorFlow/Keras imports to (1) load our MaskNet model and (2) pre-process the input image. (3) OpenCV is required for display and image manipulations.

First the image is loaded from the disk with the trained deep learning model in memory. Image pre processing is done using 'preprocess input' function which is imported from 'tensorflow.keras.applications.mobilenet v2' library. Upon loading the image from disk a copy of the image is made and frame dimensions are grabbed for future scaling and display purposes. Pre-processing is handled by OpenCV's blobFromImage function. As shown in the parameters, the image is resized and mean subtraction is performed. Face detection is performed to localize where in the image all faces are. Once all the faces are detected in the image, we loop over the detections to find the probability of face detection. If the probability is greater than 0.5 the face is considered else it is rejected. Once the faces are identified in the image, the bounding box value for a particular face is computed and then extract face ROI via Numpy slicing, pre-processing the ROI and perform the mask detection to predict 'with mask' or 'without mask'.

```
face = frame[startY:endY, startX:endX]
face = cv2.cvtColor(face, cv2.COLOR_BGR2RGB)
face = cv2.resize(face, (224, 224))
face = img_to_array(face)
face = preprocess_input(face)
```

Fig. 5: code for RIO and pre-processing

2) Real time video detection: Videos are basically made up of frames, which are still images and the face detection is applied for each frame in a video. So when it comes to detecting a face in still image and detecting a face in a realtime video stream, there is not much difference between them. The trained face-mask detector model is loaded and the frames are captured from the webcam stream using OpenCV for which the read() function is used. We run a infinite loop to get all the frames until the user presses q to quit the program. The read() function returns, the actual video frame read (one frame on each loop).

In python script we have declared a function called as 'detect and predict mask' function.

This function detects faces and then applies our face mask classifier to each face ROI. Inside, we construct a blob, detect faces, and initialize lists, two of which the function is set to return. These lists include our faces (i.e ROIs) and locs(the face locations). Then we loop over the detections to find the probability of face detection. If the probability is greater than 0.5 we consider the face else we reject it. This is basically to filter out the weak detections. Inside this loop we also extract bounding boxes while ensuring bounding box coordinates do not fall outside the bounds of the image. Once the face ROIs is extracted and pre-processing is done, it is passed to the face

```
detect_and_predict_mask (frame, faceNet, MaskNet)
        blob = cv2.dnn.blogFromImage(frame, shape)
        detections = faceNet.forward()
        for i in detection.shape[2] {
                find confidence
                if confidence > 0.5 {
                        find RIO
                        find location
                        find predictions
                }
        }
        return (predictions, location)
```

Fig. 6: Pseudo code for detect and predict mask function

mask detector. It performs the face mask detection to predict whether the person is wearing mask or not.

V. RESULTS AND ANALYSIS

A. Results and analysis for the proposed model :

All the computation required for model building was done on an i7 processor (4.4GHz) and the results obtained are from the above device. The different metrics used to evaluate the performance of the model were :

$$Precision = \frac{TP}{TP + FP} \tag{1}$$

$$Recall = \frac{TP}{TP + FN} \tag{2}$$

$$F1Score = 2 * \frac{Precision * Recall}{Precision + Reccall} \tag{3}$$

(where TN = True Negatives, FP = False Positives, TP = True Positives and FN = False Negatives)

	precision	recall	f1-score	support
with_mask	0.99	0.99	0.99	383
without_mask	0.99	0.99	0.99	384
accuracy			0.99	767
macro avg	0.99	0.99	0.99	767
weighted avg	0.99	0.99	0.99	767

Fig. 7: Classification report

Fig. 7 shows a very good performance of the proposed model. It shows the 'with mask' category has precision of 0.99, recall of 0.99 and f1-score of 0.99, 'without mask' category has precision of 0.99, recall of 0.99 and f1-score of 0.99. We used more number of images in the dataset (3706) to train the model and used 35 epochs to get good result.

B. *Results and analysis for real time detection :*

1) Testing on static images: For checking the whether our model works well or not.We have randomly collected around 5 images and have used them for testing and these images are put under a new folder called 'examples'. The class label is determined based on probabilities returned by the mask detector model and assign an associated color for the annotation.

Fig. 8: Analysis of static images

Fig. 8 tells that the model can accurately predict and classify static images, while Fig. 8(c) points out that the model is quite robust and trained well as it can correctly predict images containing multiple faces also.

2) Real time video detection: For displaying the results, first the class label is determined based on probabilities returned by the mask detector model and assign an associated color for the annotation. The color will be green for with mask and red for without mask. The output is a label text along with the probability of detection, as well as a bounding rectangular box for the face using OpenCV.

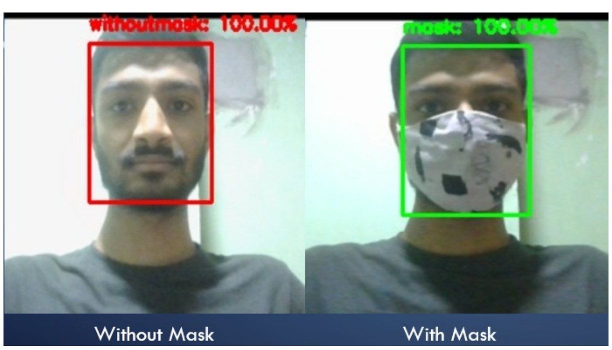

Fig. 9: Frames taken from real time detection using a web camera

Fig. 9 presents the different frames captured during the real time detection using a computer web camera.

VI. CONCLUSION

Most countries are suffering from the COVID-19 pandemic and wearing a face mask has proven to be a very good countermeasure against the spread of corona virus. In this paper a face mask detection system has been proposed which was done using Tensorflow and OpenCV. Transfer learning technique was used which was the initial layers of MobileNetV2 was used as feature extractor upon which further layers where added for classification purposes. Real time detection on static images and videos was done with OpenCV and other tensorflow libraries. The proposed model showed 99.56training accuracy and 99.33

INDIVIDUAL CONTRIBUTION

There are two main parts in our project i.e, Training the model and applying the trained model to detect whether person is wearing mask or not.

- Puttaraja
  - Model Building
  - Data set creation
  - Static image
  - Output analysis and visualization of real time detection.
- Rakesh Kallimani
  - Artificial data set creation.
  - Data pre-processing
  - Real time detection video
  - Analysis and plotting of the results

VII. BASE PAPER

M. Loey, G. Manogaran, M. Hamed N. Taha, N. Eldeen M. Khalifa, A Hybrid Deep Transfer Learning Model with Machine Learning Methods for Face Mask Detection in the Era of the COVID-19 Pandemic, Measurement (2020), doi: https://doi.org/10.1016/j.measurement.2020.108288 was considered as the base paper

ACKNOWLEDGMENT

Principally, We are thankful to the National Institute of Technology Karnataka, Information Technology Department and Prof Mr Dinesh Naik , Professor, Information Technology Department,(Interest - Computer Vision) National Institute of Technology Karnataka for this wonderful opportunity. Thank you for the support and guidance throughout our project work.

REFERENCES

H. Kopka and P. W. Daly, A Guide to LATEX, 3rd ed. Harlow, England: Addison-Wesley, 1999.

Standards to wear Face Mask, www.who.int, (accessed on sept 2020).

X. Liu and S. Zhang, "COVID-19: Face masks and human-to-human transmission," Influenza and Other Respiratory Viruses, vol. n/a, no. n/a, doi: 10.1111/irv.12740..

Charu C Agarwal, "Neural Networks and Deep Learnig:A Textbook", vol 1, pg no. 446-478, Aug 2020.

M. Loey, G. Manogaran, M. Hamed N. Taha, N. Eldeen M. Khalifa,"A Hybrid Deep Transfer Learning Model with Machine Learning Methods for Face Mask Detection in the Era of the COVID-19 Pandemic" Measurement, 2020, doi: https://doi.org/10.1016/j.measurement.2020.108288

B. QIN and D. LI, "Identifying Facemask-wearing Condition Using Image Super-Resolution with Classification Network to Prevent COVID-19,", May 2020, doi: 10.21203/rs.3.rs-28668/v1 , .

C. Li, R. Wang, J. Li, and L. Fei, "Face Detection Based on YOLOv3," in Recent Trends in Intelligent Computing, Communication and Devices,, Singapore, 2020, pp. 277–284, doi: 10.1007/978-981-13-9406-5 34

S. A. Hussain and A. S. A. A. Balushi, "A real time face emotion classification and recognition using deep learning model," J. Phys.: Conf. Ser., vol. 1432, p. 012087, Jan. 2020, doi: 10.1088/1742-6596/1432/1/012087

Kavitha M, Mohamed Monsoor Roomi S, Priya K, Bavithra Devi K, "State model based face mask detection,"

https://www.pyimagesearch.com/2020/05/04/covid-19-face-maskdetector-with-opencv-keras-tensorflow-and-deep-learn