REPORT FOR FACE MASK DETECTION

As a project work for course

ADVANCED MACHINE LEARNING (INT248)

Name	: Rohit Kumar Singh
Registration Number	: 11815212
Program	: CSE B.tech
Semester	: Fourth
School	: School of Computer Science
Name of the University	: Lovely Professional University
Date of Submission	: 20th November 2021

LOVELY PROFESSIONAL UNIVERSITY

Transforming Education Transforming India

FACE MASK DETECTION

ABSTRACT:-

After the breakout of the worldwide pandemic COVID-19, there arises a severe need of protection mechanisms, face mask being the primary one. The basic aim of the project is to detect the presence of a face mask on human faces on live streaming video as well as on images. We have used deep learning to develop our face detector model. The architecture used for the object detection purpose is Single Shot Detector (SSD) because of its good performance accuracy and high speed. Alongside this, we have used basic concepts of transfer learning in neural networks to finally output presence or absence of a face mask in an image or a video stream. Experimental results show that our model performs well on the test data with 100% and 99% precision and recall, respectively.

TABLE OF CONTENT

1 Project Introduction

The year 2020 has shown mankind some mind-boggling series of events amongst which the COVID19 pandemic is the most life-changing event which has startled the world since the year began. Affecting the health and lives of masses, COVID-19 has called for strict measures to be followed in order to prevent the spread of disease. From the very basic hygiene standards to the treatments in the hospitals, people are doing all they can for their own and the society's safety; face masks are one of the personal protective equipment. People wear face masks once they step out of their homes and authorities strictly ensure that people are wearing face masks while they are in groups and public places. To monitor that people are following this basic safety principle, a strategy should be developed. A face mask detector system can be implemented to check this. Face mask detection means to identify whether a person is wearing a mask or not. The first step to recognize the presence of a mask on the face is to detect the face, which makes the strategy divided into two parts: to detect faces and to detect masks on those faces. Face detection is one of the applications of object detection and can be used in many areas like security, biometrics, law enforcement and more. There are many detector systems developed around the world and being implemented. However, all this science needs optimization; a better, more precise detector, because the world cannot afford any more increase in corona cases. 1 In this project, we will be developing a face mask detector that is able to distinguish between faces with masks and faces with no masks. In this report, we have proposed a detector which employs SSD for face detection and a neural network to detect presence of a face mask. The implementation of the algorithm is on images, videos and live video streams.

2 Literature Review

Object detection is one of the trending topics in the field of image processing and computer vision. Ranging from small scale personal applications to large scale industrial applications, object detection and recognition is employed in a wide range of industries. Some examples include image retrieval, security and intelligence, OCR, medical imaging and agricultural monitoring. In object detection, an image is read and one or more objects in that image are categorized. The location of those objects is also specified by a boundary called the bounding box. Traditionally, researchers used pattern recognition to predict faces based on prior face models. A breakthrough face detection technology then was developed named as Viola Jones detector that was an optimized technique of using Haar [1], digital image features used in object recognition. However, it failed because it did not perform well on faces in dark areas and non-frontal faces. Since then, researchers are eager to develop new algorithms based on deep learning to improve the models. Deep learning allows us to learn features with end to end manner and removing the need to use prior knowledge for forming feature extractors. There are various methods of object detection based.

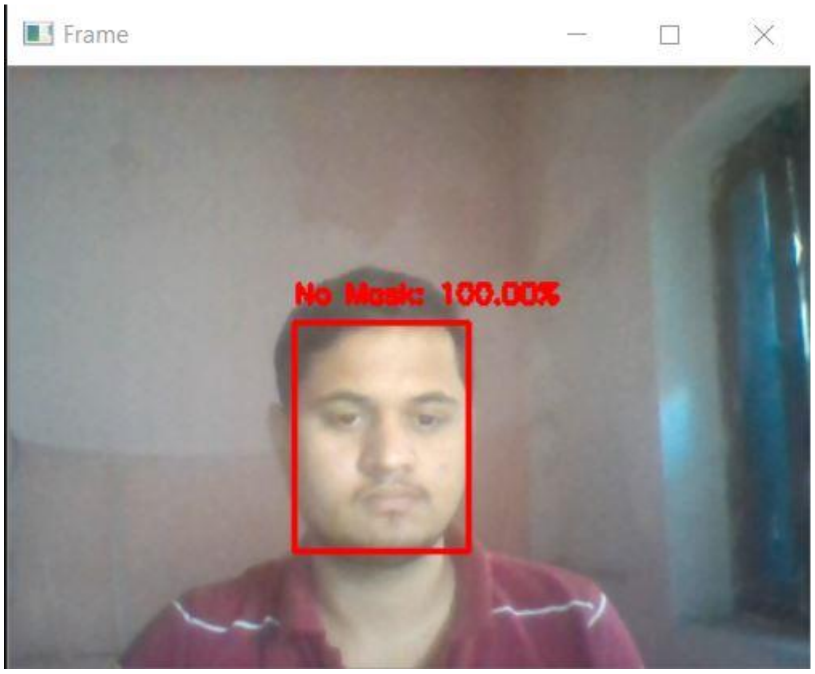
Two stage detectors use two neural networks to detect objects, for instance region-based convolutional neural networks (R-CNN) and faster R-CNN. The first neural network is used to generate region proposals and the second one refines these region proposals; performing a coarse-to-fine detection. This strategy results in high detection performance compromising on speed. The seminal work R-CNN is proposed by R. Girshick et al. [2]. R-CNN uses selective search to propose some candidate regions which may contain objects. After that, the proposals are fed into a CNN model to extract features, and a support vector machine (SVM) is used to recognize classes of objects. However, the second stage of R-CNN is computationally expensive since the network has to detect proposals on a one-by-one manner and uses a separate SVM for final classification. Fast R-CNN [3] solves this problem by introducing a region of interest (ROI) pooling layer to input all proposal regions at once. Faster RCNN [4] is the evolution of R-CNN and Fast R-CNN, and as the name implies its training and testing speed is greater than those of its predecessors. While R-CNN and Fast R-CNN use selective search algorithms limiting the detection speed, Faster R-CNN learns the proposed object regions itself using a region proposal network (RPN). 2 On the other hand, a one stage detector utilizes only a single neural network for region proposals and for detection; some primary ones being SSD (Single Shot Detection) [5] and YOLO (You Only Look Once) [6]. To achieve this, the bounding boxes should be predefined. YOLO divides the image into several cells and then matches the bounding boxes to objects for each cell. This, however, is not good for small sized objects. Thus, multi scale detection is introduced in SSD which can detect objects of varying sizes in an image. Later, in order to improve detection accuracy, Lin et. al [7] proposes Retina Network (RetinaNet) by combining an SSD and FPN (feature pyramid network) to increase detection accuracy and reduce class imbalance. One-stage detectors have higher speed but trades off the detection performance but then only are preferred over two-stage detectors. Like object detection, face detection adopts the same architectures as one-stage and two-stage de�tectors, but in order to improve face detection accuracy, more face-like features are being added. However, there is occasional research focusing on face mask detection. Some already existing face mask detectors have been modeled using OpenCV, Pytorch Lightning, MobileNet, RetinaNet and Support Vector Machines. Therefore, the ratio of the samples in train/validation while splitting the dataset was kept equal using the train test split function of sklearn. Moreover, to deal with unbalanced data, they passed this information to the loss function to avoid unproportioned step sizes of the optimizer. They did this by assigning a weight to each class, according to its representability in the dataset. They assigned more weight to classes with a small number of samples so that the network will be penalized more if it makes mistakes predicting the label of these classes. While classes with large numbers of samples, they assigned to them a smaller weight. This makes their network training agnostic to the proportion of classes. The weights for each class were chosen using the formula below: Class Weight = 1 – (Class Cardinality)/(Cardinalities of all classes) To load the data efficiently this project used the data loader. For instance, in this project, they used the PyTorch lighting, and to load them for training and validation they divided data into 32 batches and assigned the works of loading to the 4 number of workers, and this procedure allowed them to perform multi-process data loading. Like most of the projects, this project also used Adam optimizer. If any Model has a high rate of learning, it learns faster, but it bounces a lot to reach the global minima and may diverge from the global minima. However, a small learning rate may take considerably lower time to train, but it reaches to the global minima. If the loss of the model declines quickly for any learning rate, then that learning rate would be the best learning rate. However, it seems that this project considered the 0.00001 learning rate would be the best for their model so that it could work efficiently. To train the model they defined a model checkpointing callback where they wanted to save the best accuracy and the lowest loss. They tried to train the model for 10 epochs and after finding optimal epoch, they saved the model for 8 epochs to test on the real data. To get rid of the problem of occlusions of the face which causes trouble face detectors to detect masks in the images, they used a built-in OpenCV deep learning face detection model. For instance, the Haar-Cascade model could be used but the problem of the Haar-Cascade model is that the detection frame is a rectangle, not a square. That is why, without capturing the portion of the background, the face frame can fit the entirety of the face, which can interfere with the face mask model predictions. In the second project [9], a dataset was created by Prajna Bhandary using a PyImageSearch reader. This dataset consists of 1,376 images belonging to all races and is balanced. There are 690 images with masks and 686 without masks. Firstly, it took normal images of faces and then created a customized computer vision Python script to add face masks to them. Thereby, it created a real�world applicable artificial dataset. This method used the facial landmarks which allow them to detect the different parts of the faces such as eyes, eyebrows, nose, mouth, jawline etc. To use the facial landmarks, it takes a picture of a person who is not wearing a mask, and, then, it detects the portion of that person's face. After knowing the location of the face in the image, it extracted the face Region of Interest (ROI). After localizing facial landmarks, a picture of a mask is placed into 3 the face. In this project, embedded devices are used for deployment that could reduce the cost of manufacturing. MobileNetV2 architecture is used as it is a highly efficient architecture to apply on embedded devices with limited computational capacity such as Google Coral, NVIDIA Jetson Nano. This project performed well, however, if a large portion of the face is occluded by the mask, this model could not detect whether a person is wearing a mask or not. The dataset used to train the face detector did not have images of people wearing face masks as a result, if the large portion of faces is occluded, the face detector would probably fail to detect properly. To get rid of this problem, they should gather actual images of people wearing masks rather than artificially generated images.

3 Dataset

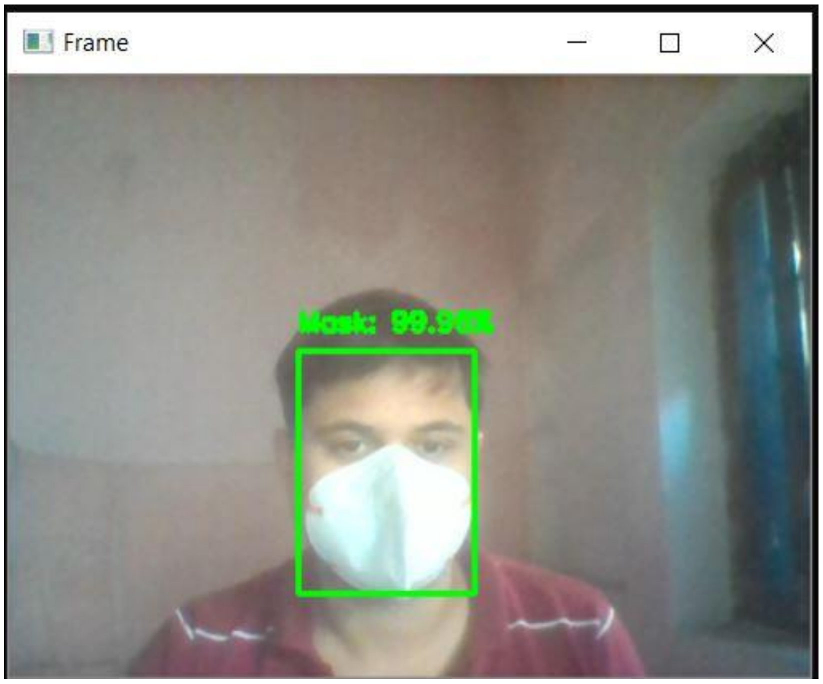
The dataset which we have used consists of 3833 total images out of which 1915 are of masked faces and 1918 are of unmasked faces. All the images are actual images extracted from Bing Search API, Kaggle datasets and open source image library. From all the three sources, the proportion of the images is equal. The images cover diverse races i.e Asian, Caucasian etc. The proportion of masked to unmasked faces determine that the dataset is balanced. We need to split our dataset into three parts: training dataset, test dataset and validation dataset. The purpose of splitting data is to avoid overfitting which is paying attention to minor details/noise which is not necessary and only optimizes the training dataset accuracy. We need a model that performs well on a dataset that it has never seen (test data), which is called generalization. The training set is the actual subset of the dataset that we use to train the model. The model observes and learns from this data and then optimizes its parameters. The validation dataset is used to select hyperparameters (learning rate, regularization parameters). When the model is performing well enough on our validation dataset, we can stop learning using a training dataset. The test set is the remaining subset of data used to provide an unbiased evaluation of a final model fit on the training dataset. Data is split as per a split ratio which is highly dependent on the type of model we are building and the dataset itself. If our dataset and model are such that a lot of training is required, then we use a larger chunk of the data just for training which is our case. If the model has a lot of hyperparameters that can be tuned, then we need to take a higher amount of validation dataset. Models with a smaller number of hyperparameters are easy to tune and update, and so we can take a smaller validation dataset. In our approach, we have dedicated 80% of the dataset as the training data and the remaining 20% as the testing data, which makes the split ratio as 0.8:0.2 of train to test set. Out of the training data, we have used 20% as a validation data set. Overall, 64% of the dataset is used for training, 16% for validation and 20% for testing.

4 Result and Experiment analysis

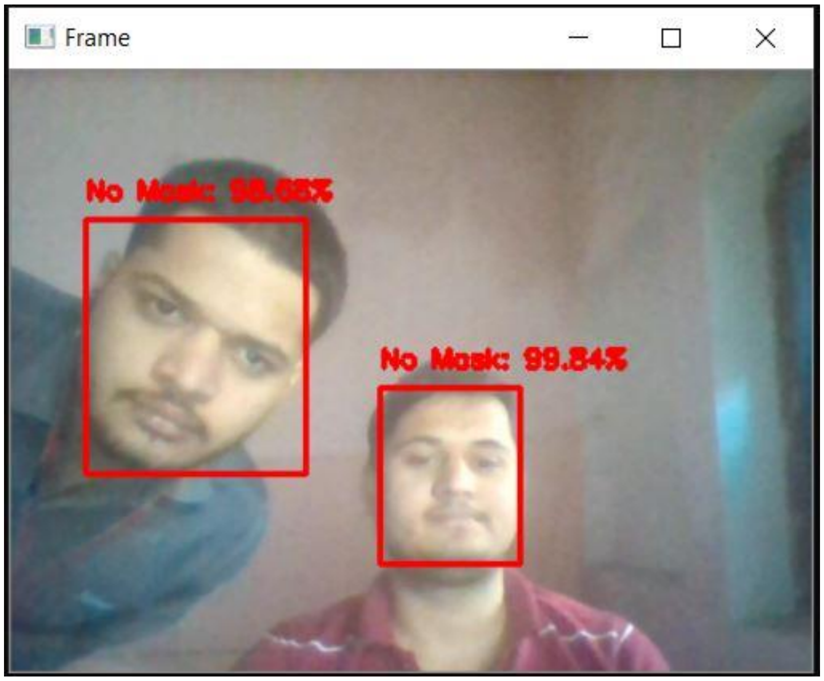
We implemented our model on images containing one and more faces. We also implemented it on videos and live video streams by removing and wearing masks one by one. Some screenshots of the results are shown below:



My Face without mask



My Face with mask



Me and my brother Face without mask

GitHub Link:- https://github.com/rohitsingh39/Face-Mask-Detection.git

5 Future Work

More than fifty countries around the world have recently initiated wearing face masks compulsory. People have to cover their faces in public, supermarkets, public transports, offices, and stores. Retail companies often use software to count the number of people entering their stores. They may also like to measure impressions on digital displays and promotional screens. We are planning to improve our Face Mask Detection tool and release it as an open-source project. Our software can be equated to any existing USB, IP cameras, and CCTV cameras to detect people without a mask. This detection live video feed can be implemented in web and desktop applications so that the operator can see notice messages. Software operators can also get an image in case someone is not wearing a mask. Furthermore, an alarm system can also be implemented to sound a beep when someone without a mask enters the area. This software can also be connected to the entrance gates and only people wearing face masks can come in.

6 Conclusion

To mitigate the spread of COVID-19 pandemic, measures must be taken. We have modeled a face mask detector using SSD architecture and transfer learning methods in neural networks. To train, validate and test the model, we used the dataset that consisted of 1916 masked faces images and 1919 unmasked faces images. These images were taken from various resources like Kaggle and open source images. The model was inferred on images and live video streams. This face mask detector can be deployed in many areas like shopping malls, airports and other heavy traffic places to monitor the public and to avoid the spread of the disease by checking who is following basic rules and who is not.

7 Github Link:

https://github.com/rohitsingh39/Face-Mask-Detection.git

8 References

https://www.researchgate.net/publication/344173985_Face_Mask_Detector

Rosebrock, A., 2020. COVID-19: Face Mask Detector With Opencv, Keras/Tensorflow, And Deep Learning - Pyimagesearch. [online] PyImageSearch. Available at:

https://www.pyimagesearch.com/2020/05/04/covid�19-face-mask-detector-with-opencv-keras-tensorflow-and-deep-learning/ .