Real-Time Corona Mask Detection Including Facial Keypoints

IIT-K Internship Project

July 18, 2020

Jeet Banik
Jain University, Bangalore
jeetbanik111@gmail.com

Pranjal Dubey
Galgotias University, UP
pranjaldub1999@gmail.com

ABSTRACT

The COVID-19 pandemic i.e., the Coronavirus disease of 2019 has affected the world tremendously. One major protection/prevention method for people is to wear face masks in public areas to prevent the spread of this disease. Furthermore, many public service providers require customers to use the service only if they wear masks, correctly covering both their nose and mouth, as per the guidelines from the World Health Organization (WHO). Manual screening and identification of people violating this policy is a herculean task in public areas. So, the ideal approach is to employ technology solutions like Artificial Intelligence and Deep Learning to be used at real-time to make this task simple, which is easy to use and automated. However, there are only a few research studies about face mask detection based on image analysis. In this paper, we propose "DeepFaceMask", which is a high-accuracy and efficient face mask detector. The proposed DeepFaceMask is a one-stage detector, which consists of a Deep Convolutional Neural Network (DCNN) to fuse high-level semantic information with multiple feature maps. In addition, we also propose a novel cross-class object removal algorithm to reject predictions with low confidences and the high intersection of union with the help of a pre-trained Deep Convolutional Neural Network (DCNN) specifically designed for joint face detection and alignment prediction known as Multi-task Cascaded Convolutional Neural Networks (MTCNN). Besides this, we also explore the possibility of implementing DeepFaceMask with a light-weighted neural network MobileNet for embedded or mobile devices. Multi-task Cascaded Convolutional Neural Networks, abbreviated as MTCNN, exploits the inherent correlation between detection and alignment to boost up their performance. In particular, our framework leverages a cascaded architecture with three stages of carefully designed DCNN to predict the face and its key points or landmarks location in a coarse-to-fine manner.

INTRODUCTION

In order to effectively prevent the spread of COVID-19 virus, almost everyone wears a mask during coronavirus epidemic. This almost makes conventional facial recognition technology ineffective in many cases, such as community access control, face access control, facial attendance, facial security checks at train stations, etc. The science around the use of masks by the general public to impede COVID-19 transmission is advancing rapidly. Policymakers need guidance on how masks should be used by the general population to combat the COVID-19 pandemic. And masks need to be worn correctly on the face such that it covers the nose and mouth completely, which is often not being followed. Therefore, it is very urgent to improve the recognition performance of the existing face recognition technology on the masked faces. Face mask detection has become a crucial computer vision task to help the global society, but research related to face mask detection is limited.

Face mask detection refers to detect whether a person is wearing a mask or not and what is the location of the face, which we detect including the facial keypoints also. The problem is closely related to general object detection to detect the classes of objects (here we deal with mainly three classes namely: wearing mask correctly, wearing mask incorrectly, and not wearing mask) and face detection is to detect a particular class of objects, i.e. face. Applications of object and face detection can be found in many areas, such as autonomous driving, education, surveillance and so on. Traditional object detectors are usually based on handcrafted feature extractors. We implement deep learning based object detectors as it demonstrates excellent performance and dominate the development of modern object detectors. Without using prior knowledge for forming feature extractors, deep learning allows neural networks to learn features with an end-to-end manner.

PROBLEM STATEMENT

The goal of this project is to train 'Object Detection Models' capable of identifying facial keypoints for 'Face Recognition' and 'Emotion Detection' and the location of masked faces in an image and real-time scenario as well as the location of unmasked faces in the static image. These detectors should be robust to noise and provide as little room as possible to accommodate False Positives for masks due to the potentially dire consequences that they would lead to. Ideally, they should be fast enough to work well for real-world applications, something we hope to focus on in our future implementations.

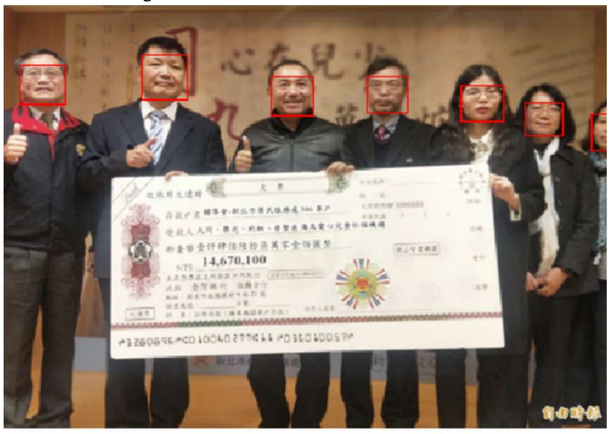
VISION

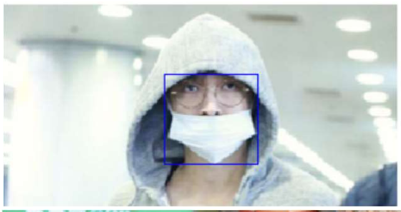
This project was created with the vision of developing a "Real-Time Mask Detection System" available for public use, to help public health officials and small to large establishments all over the world to effectively combat this COVID19 pandemic. We hope that the models developed here by the small research AI/ML community enable developers around the globe to be able to use and deploy the same to build systems that would be capable to withstand the demands of a real-time, real-world use case. In particular, it would help factories ensure mask compliance is followed, help ensure safety for visitors in containment zones or public places where it is crucial for such measures to be taken, and so on. The applications are boundless and are of urgent need in this critical time.

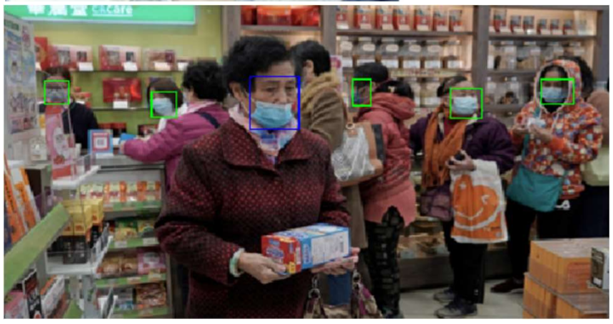
DATASET

The dataset we will be using primarily is the MaskPascalVOC zip file taken from the website: https://makeml.app/datasets/mask
The dataset contains 853 images of the following classes: With mask, Without mask, and Mask weared incorrect. It is labeled with bounding box annotations for object detection. But the number of images belonging to the class of mask worn incorrectly are too less in quantity compared to the other two classes in the dataset, which was creating class imbalance so, we collected data from additional sources having the class name as None for people not wearing masks correctly, which we have combined separately and uploaded on the web, here's the url:
https://mega.nz/file/l5thGJbY#wQHynfAuq9TVK0OuCVs28itFQYprI8UY6z0APnXOqMg

So, finally our combined dataset overall has the following four labels namely: with_mask, without_mask, mask_weared_incorrect and none. This is divided finally into 3 classes, first one having the label "with_mask" which we signify later by a green colour bounding box on the face with a text label over it as "Correctly Masked", second having the label "without_mask" which we signify later by a red colour bounding box on the face with a text label over it as "Unmasked", and the third one having either of the two labels, "mask_weared_incorrect" or "none" which we signify later by a blue colour bounding box on the face with a text label over it as "Incorrectly Maksed". We finally split the dataset into a training, validation and test set with an 8:1:1 split, respectively. We made sure that for each class, a random 10% was in the validation set, another random 10% was in the test set and a random 80% was in the training set.







*Additionally, we are also implementing the main facial keypoints inside the bounding box while detecting the face and the dataset used to train this model is taken from the website: https://ibug.doc.ic.ac.uk/download/annotations/xm2vts.zip/ The data is in the format of a CSV (Comma Separated Values) file where there are sixty-eight keypoints of images representing x, y coordinates. This data is being fed into a deep CNN or ConvNet model with the final layer having 68*2=136 dimensions output predicting the X and Y coordinates of those sixty eight keypoints. Smooth L1 loss and MSE (Mean Squared Error) loss metrics resulted in the best accuracy outputs, we choose Smooth L1 loss metric for our final model as it performed better in real-time comparatively.*

RELATED WORK

1. Object Detection

The face detection technique used here is MTCNN (Multi-task Cascaded Convolutional Networks). Face detection and alignment in unconstrained environment are challenging due to various poses, illuminations and occlusions. Recent studies show that deep learning approaches can achieve impressive performance on these two tasks. In this paper, we have used a Deep Cascaded multi-task framework which exploits the inherent correlation between detection and alignment to boost up their performance. In particular, this frame-work leverages a cascaded architecture with three stages of carefully designed Deep Convolutional Neural Networks to predict face and landmark location in a coarse-to-fine manner. In addition, it proposes a new online hard sample mining strategy that further improves the performance in practice. This method achieves superior accuracy over the state-of-the-art techniques on the challenging FDBB (Face Detection Data Set and Benchmark) and Wider Face benchmarks for face detection, and AFLW (Annotated Facial Landmarks in the Wild) benchmark for face alignment, while keeps real time performance.

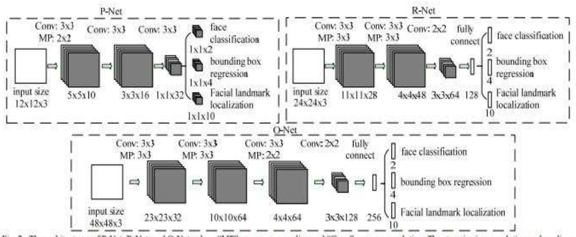
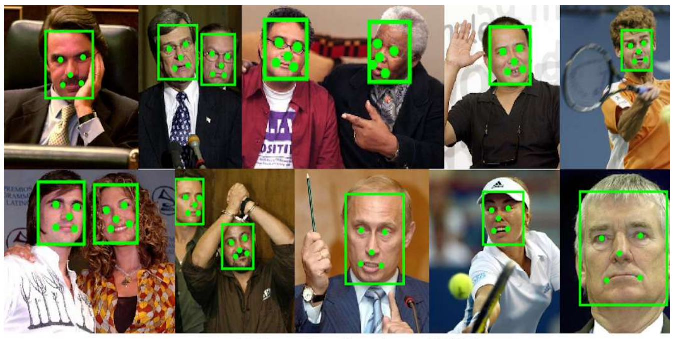
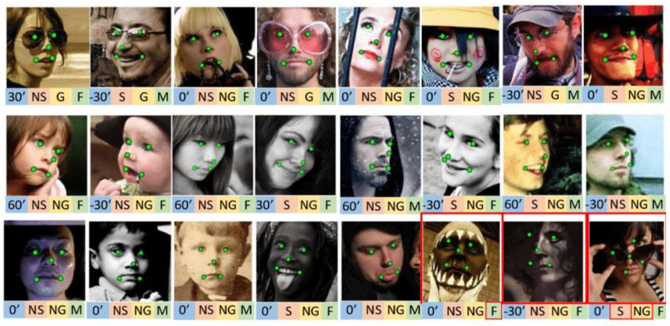


Fig. 2. The architectures of P-Net, R-Net, and O-Net, where "MP" means max pooling and "Conv" means convolution. The step size in convolution and pooling is 1 and 2, respectively.

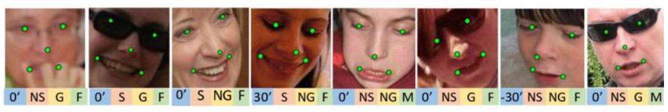
Realtime analysis of MTCNN face and keypoints detection:


(a) Examples of results on FDDB

(b) Examples of results on WIDER FACE



(a) Results on AFLW: Faces with occlusion (row 1), pose variation (row 2), different lighting conditions (column 1-2 in row 3), low image quality (column 3 in row 3), different expressions (column 4-5 in row 3), three inaccurate cases are shown in column 6-8 in row 3.



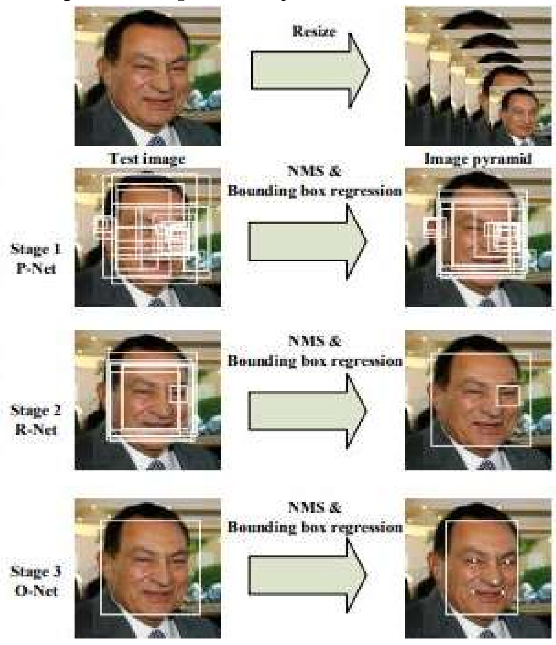
(b) Results on AFW

MTCNN (Multi-task Cascaded Convolutional Neural Networks) is an algorithm consisting of 3 stages, which detects the bounding boxes of faces in an image along with their 5 Point Face Landmarks. Each stage gradually improves the detection results by passing it's inputs through a CNN, which returns candidate bounding boxes with their scores, followed by non max suppression.

The MTCNN framework / Architecture uses three separate networks:

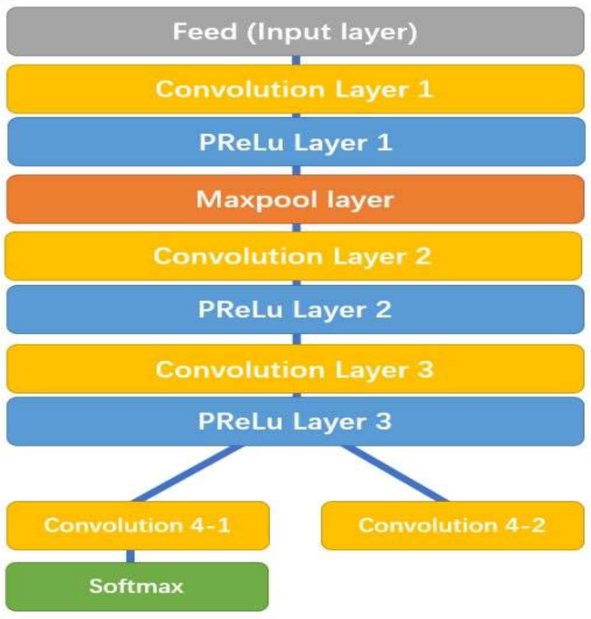
- *P – Net*
- *R – Net*
- *O – Net*

Sliding window process of MTCNN:



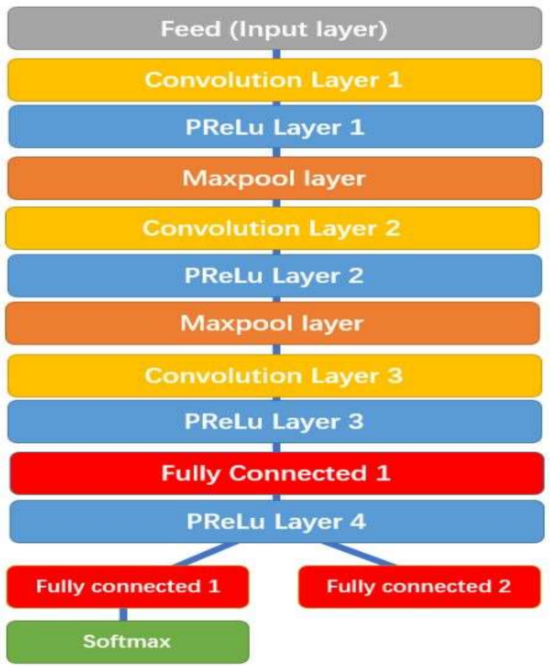
- *<u>Structure of P-Net:</u>*

*P-Net predicts bounding box using sliding a 12*12 size kernel/filter across the image.*



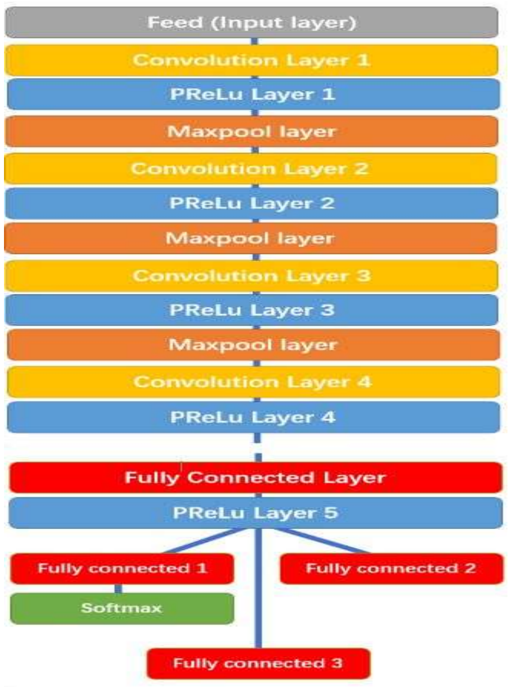
- *<u>Structure of R-Net:</u>*

R-Net has similar structure, but uses more layer, thus predicts more accurate bounding box coordinates.



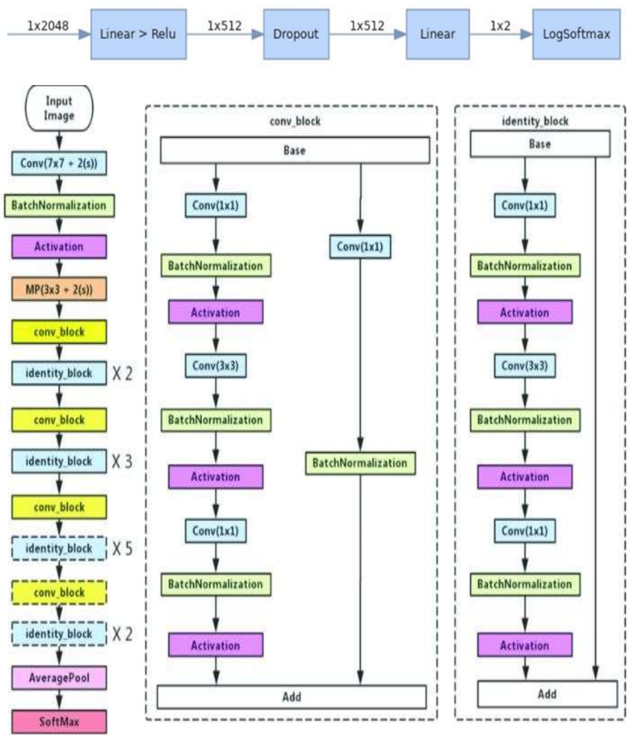
- *<u>Structure of O-Net:</u>*

O-Net takes the output of R-Net and predicts three sets of data namely - the probability of face being in the box, bounding box, and facial keypoints.



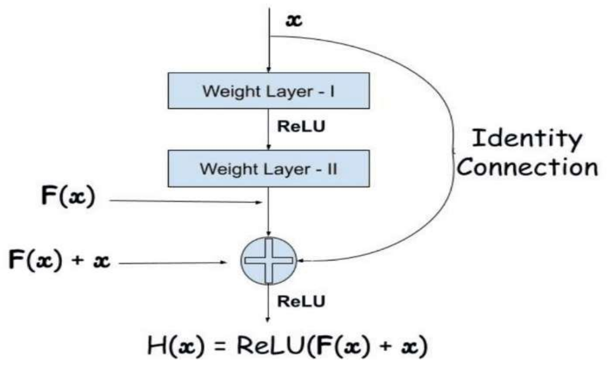
2. Image Classification

Image classification refers to extracting specific desired features from a static or a real time image and classifying it to solve a specific problem at hand. This objective was accomplished by using transfer learning approach. ResNet-50 pre-trained model was used as a feature extractor connected with a custom fully connected layer for robust and efficient image classification. The model was trained on a dataset consisting three classes, masked, not masked, not properly masked respectively. The problem with the dataset was that it didn't represent the same amount of each class i.e. it was an imbalance of data, so the model was trained on two datasets combined. To achieve more robust results, custom image augmentation techniques were implemented during the training process. The convolutional layers of ResNet-50 were used as feature extractor (last convolutional layers), rest all were frozen during training. Thus, fine tuning the model gave much better results from traditional state-of-the-art architectures. It also helped in tackling vanishing gradients problem by leveraging the use of skip connections and strong robust feature extractor proved to be efficient enough to extract features from a relatively small dataset. ResNet-50 layers were connected to linear layers before end-to-end result prediction.



Since 2013, the Deep Learning community started to build deeper networks because they were able to achieve high accuracy values. Furthermore, deeper networks can represent more complex features, thus the robustness of the model and its performance can be increased. However, stacking up more layers didn't work for the researchers. While training deeper neural networks, the problem of accuracy degradation was observed. In other words, adding more layers to the network either made the accuracy value to saturate or it abruptly started to decrease. The culprit for accuracy degradation was the vanishing gradient effect which can only be observed in deeper networks. During the backpropagation stage, the error is calculated and gradient values are determined. The gradients are sent back to hidden layers and the weights are updated accordingly. The process of gradient determination and sending it back to the next hidden layer is continued until the input layer is reached. The gradient becomes smaller and smaller as it reaches the bottom of the network. Therefore, the weights of the initial layers will either update very slowly or remain the same. In other words, the initial layers of the network won't learn effectively. Hence, deep network training will not converge and accuracy will either start to degrade or saturate at a particular value. Although the vanishing gradient problem was addressed using the normalized initialization of weights, deeper network accuracy was still not increasing. Deep Residual Network is almost similar to the networks which have convolution, pooling, activation and fully-connected layers stacked one over the other. The only construction to the simple network to make it a residual network is the identity connection between the layers. The picture below shows the residual block used in the network. It can be seen the identity connection as the curved arrow originating from the input and sinking to the end of the residual block.

Skip connections used by ResNet-50.



<u>*Key Features of ResNet:*</u>

- *ResNet uses Batch Normalization at its core. The Batch Normalization adjusts the input layer to increase the performance of the network. The problem of covariate shift is mitigated.*
- *ResNet makes use of the Identity Connection, which helps to protect the network from vanishing gradient problems.*
- *Deep Residual Network uses bottleneck residual block design to increase the performance of the network.*

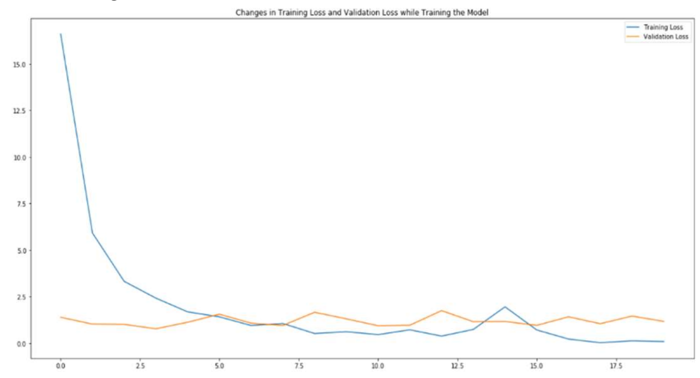
TRAINING

After pre-processing the data, our combined dataset consists a total number of 4198 images. Number of images labelled 1 i.e. wearing mask correctly are 3232, number of images labelled 2 i.e. not wearing mask are 717, number of images labelled 3 i.e. wearing mask incorrectly are 249.
The dataset was then divided into training data, validation data and test data. It was split into 8:1:1 ratio i.e. train set size is 3358, validation set size is 419 and test set size is 421. The difference in the validation and test size, despite the same ratio, is because test set size was calculated after calculating the train set size and validation set size and their summation was subtracted from the total number of images. Also, images were randomly shuffled for no imbalance of class and robust performance of model, in batch size of 64 for faster computation. We trained the model using cross-entropy loss and Adam optimizer (an upgrade to stochastic gradient descent with momentum capabilities). In addition, the learning rate was set as 10^{-3} i.e. 0.001 and the number of epochs as 20, post this the model stopped learning based on earlier observations during training.
Challenges faced during the training process was that single data source wasn't enough to provide sufficient number of images belonging to each class. So, many data sources were considered and a robust, balanced, sufficiently large dataset was created that would provide enough data for the model to adapt to variances in data. GPU was used in training the model due to the large data. Training on GPU proved to be about 3x times faster than training on the CPU.
GPU model used while training: NVIDIA GeForce GTX 1050 2GB GDDR5. Lighting and camera settings play a major role in model performance. Thus, we used MTCNN, which easily tackles such problems.
Total params: 24,558,146
Trainable params: 16,014,850
Non-trainable params: 8,543,296

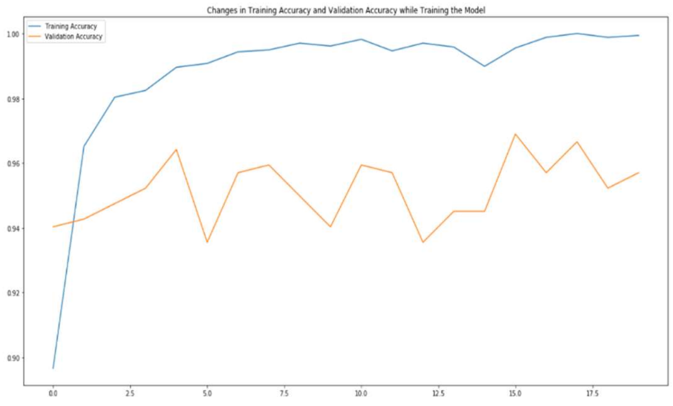
RESULTS

The best model saved during training resulted in a validation loss of 0.9591 and validation accuracy of 0.9689 which was used in testing in real-time. Also, the model resulted in 98% accuracy on the test data.

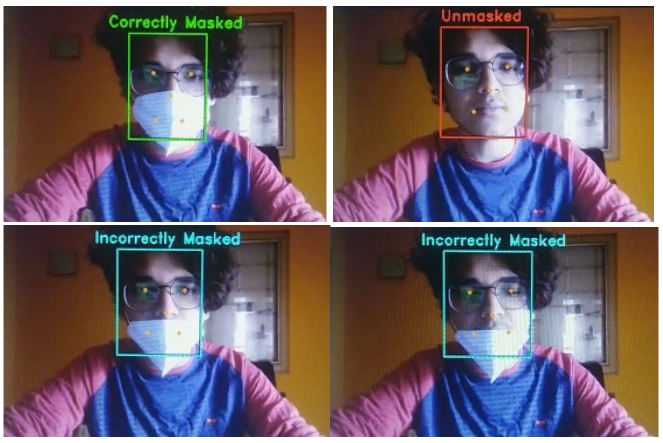
Training Loss and Validation Loss Visualization



Training Accuracy and Validation Accuracy Visualization



Tested in real-time



Real-Time Applications:

- *Mall security checks / Super markets*
- *Offices spaces / Schools*
- *Hospitals*
- *Mobile applications for alerts*

FUTURE CONSIDERATIONS

It is evident that one of our biggest obstacles during the COVID-19 pandemic is to make sure people follow the safety regulations especially in public places for his/ her own safety and the safety of others around. Our DeepFaceMask model will thus detect if people are wearing masks or not, correctly, when deployed to the CCTVs in the public places and can alert the admin as and when people are not wearing masks or wearing masks incorrectly. Additionally, it can be used in head pose estimation, attention detection in classrooms and lectures on masked faces, drowsiness detection on masked faces using facial keypoints tracking the driver's eyes, and so on.

REFERENCES

[1] T.-H. Kim, D.-C. Park, D.-M. Woo, T. Jeong and S.-Y. Min, "Multi-class classifier-based adaboost algorithm", Proceedings of the Second Sino-foreign-interchange Conference on Intelligent Science and Intelligent Data Engineering, pp. 122-127, 2012.

[2] P. Viola and M. J. Jones, "Robust real-time face detection", Int. J. Comput. Vision, vol. 57, no. 2, pp. 137-154, May 2004.

[3] Z. A. Memish, A. I. Zumla, R. F. Al-Hakeem, A. A. Al-Rabeeah, and G. M. Stephens, "Family cluster of middle east respiratory syndrome coronavirus infections," New England Journal of Medicine, vol. 368, no. 26, pp. 2487–2494, 2013.

[4] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," 2017.

[5] A. Shrivastava, A. Gupta, and R. Girshick, "Training region-based object detectors with online hard example mining," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 761–769.

[6] S. Ge, J. Li, Q. Ye, and Z. Luo, "Detecting masked faces in the wild with lle-cnns," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 2682–2690.

[7] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga et al., "Pytorch: An imperative style, high-performance deep learning library," in Advances in Neural Information Processing Systems, 2019, pp. 8024–8035.

[8] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.

[9] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," arXiv preprint arXiv:1704.04861, 2017.

[10] J. Deng, J. Guo, Y. Zhou, J. Yu, I. Kotsia, and S.Zafeiriou, "Retinaface: Single-stage dense face localization in the wild," arXiv preprint arXiv:1905.00641, 2019.

[11] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2014, pp. 580–587.

[12] Krizhevsky A, Sutskever I, Hinton GE (2012) ImageNet classification with deep convolutional neural networks. Adv Neural Inf Process Syst 25.

[13] S. Yang, P. Luo, C.-C. Loy, and X. Tang, "Wider face: A face detection benchmark," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 5525–5533.

[14] R. Girshick, "Fast r-cnn," in Proceedings of the IEEE international conference on computer vision, 2015, pp. 1440–1448.

[15] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in 2009 IEEE conference on computer vision and pattern recognition. Ieee, 2009, pp. 248–255.

[16] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in Proceedings of the IEEE conference on computer vision, 2016, pp. 779–788.

[17] A. Krizhevsky, I. Sutskever and G. E. Hinton, "Imagenet classification with deep convolutional neural networks", Advances in Neural Information Processing Systems 25, pp. 1097-1105, 2012.

[18] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition", CoRR, 2014.

[19] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, et al., "Going deeper with convolutions", 2015.

[20] K. He, X. Zhang, S. Ren and J. Sun, "Deep residual learning for image recognition", 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 770-778, 2016.

[21] P. Viola and M. J. Jones, "Robust real-time face detection", Int. J. Comput. Vision, vol. 57, no. 2, pp. 137-154, May 2004.