REAL TIME FACE MASK DETECTION

Agam Madan
Dept. of Computer science and Engineering
Bharati Vidyapeeth's College of Engineering, IP University
New Delhi, India
madanagam@gmail.com

Rohan Arora
Dept. of Computer science and Engineering
Bharati Vidyapeeth's College of Engineering, IP University
New Delhi, India
rohanaroraaashi@gmail.com

Piyush Katariya
Dept. of Computer science and Engineering
Bharati Vidyapeeth's College of Engineering, IP University
New Delhi, India
piyushkatariya95@gmail.com

Abstract

Face mask detection had seen a major progress in the fields of Computer vision and Image processing and since the rise of Covid-19 pandemic. Many face detection models have been created using several algorithms and techniques. The proposed approach in this paper uses deep learning, TensorFlow, Keras and OpenCV to detect face masks. This model has the potential to be used for safety purposes since it is very resource efficient to deploy. The proposed approach uses MobilenetV2 architecture as a framework for classifier which is very lightweight and can even be used in embedded devices (like NVIDIA Jetson Nano, Raspberry pi) to perform real time mask detection. The technique deployed in this paper gives us an accuracy score of 0.9253 and F1 score of 0.93. The dataset provided in this paper which was collected from various sources can be used by other researchers for further advanced models such as those of face recognition, facial landmarks, and facial part detection process.

Keywords: MobileNetV2, Data Augmentation, Bottleneck, Fine Tuning.

1. Introduction

Face Mask detection has turned up to be an astonishing problem in the field of image processing and computer vision. Face detection has various use cases ranging from face recognition to capturing facial motions where the latter needs the face to be detected with a very high precision. Due to the rapid advancement of machine learning algorithms, the jeopardies of face detection technology seem to be well addressed yet. This technology is more relevant in today's era because this application is used to detect faces not only in static images and videos but also in real time inspection and supervision. With the advancements of convolution neural networks [1] and deep learning very high accuracy in image classification and object detection can be achieved.

Face Mask detection has become a very trending application due to Covid-19 pandemic, which demands for a person to wear face masks, keep social distancing, and use hand sanitizers to wash your hands. While other problems of social distancing and sanitization have been addressed up till now, the problem of face mask detection has not been properly addressed yet. This paper proposes a model for face mask detection using OpenCV DNN [2], TensorFlow [26], Keras and MobileNetV2 architecture [3] which is used as an image classifier. This model works very well in differentiating images having frontal faces with masks from images having frontal faces without masks. This paper also keeps complete attention towards removal of various inaccurate predictions which occurred in various other proposed models.

Detection of face masks is an extremely challenging task for the present proposed models of face detectors[4 - 8]. This is because faces with masks have varied accommodations, various degrees of obstructions and diversified mask types. They are used to facilitate auto-focusing [9], human computer interaction [10] and image database management [11]. Even after having such extraordinary and exceptional results in the existing face detectors, still there is high rising scrutiny in the development of more advanced face detectors as for existing models event analysis and video surveillance is still a challenging job. Several reasons were found for the poor achievement of existing face mask detection model as compared to the normal ones, two of them were First due to lack of a good datasets with proper masked faces and facial recognition, Secondly, the presence of masks on the face brings an undeniable kind of noise which further deteriorates the detection process. These issues have been studied in some existing research papers such as [12 - 14], still there is a great demand for a large dataset so that an efficient face mask detection model can be easily developed.

The main contributions of this paper are three folds.
i.) A GitHub repository is made available which contains self-made dataset of masked faces including datasets taken from online resources. This dataset could be used for developing new face mask detectors and performing several applications.
ii.) OpenCV DNNs have been used for face mask detection, which allows for real time detection without much resource usage. It is also able to detect faces in different orientations and can also detect occuluded faces with good accuracy.
iii.) Several provocations which were faced during development of this model have been considered in this paper, this may help the reader for developing more improved face mask detectors.
The upcoming section 2 discusses the similar work proposed and completed in the domain of face mask detection. In section 3 the description of the face mask detection dataset used to train this model is given. In section 4 we discussed the methodology adopted and technology used to build this Face mask detection model. In section 5 experimental results have been shown along with a comparison table. Section 6 shows the conclusion of the paper.

2. Related Work

One of the main contributions in this research paper is the dataset created and the technology used. Some of the works done by researchers and analysts on face mask detection models have been mentioned in this section.
In the recent past various researchers and analysts mainly focused on grayscale face image [15]. While some were completely based on pattern recognition models, having initial information of the face model while others were using AdaBoost [16], which was a good classifier for training purposes. Then came the Viola Jones Detector which provided a major breakthrough in face detection technology and real time face detection got possible. It faced various problems like the orientation and brightness of face, making it hard to intercept. So basically it failed to work in dull and dim light. So, researchers started searching for a new alternative model which could easily detect faces as well as masks on the face.
In the past many datasets for face detection have been developed to assess face detection models. Earlier datasets mainly consisted of images collected in supervised surroundings, while recent datasets are constructed by taking online images like WiderFace [20], IJB-A [18], MALF [17], and CelebA [19]. Annotations are provided for present faces in these datasets as compared to earlier ones. Large datasets are much more needed for making a better training and testing data and perform real world applications in a much simpler way. This calls for the use of various deep learning algorithms which can read faces and mask straight from the data provided by the user.
Face Mask detection models are grouped into different variations:
i. Boosting-based categories:
In this category, boosted cascades with easy haar features were embraced using the Viola Jones face detector [21], which we discussed above in this section. Then a Multiview face mask detector was made motivated by the Viola Jones detector model. In addition to this, a face mask detector model was made using decision trees algorithms. Face mask detectors in the boosting-based category are often very effective in detecting face masks.
ii. DPM-based category:
In this category, the structure and orientations of several different faces are modelled using DPM. In 2006 Ramanan proposed a Random forest tree model for face mask detection, which accurately guesses face structures and facial poses. Mathias et al. [22], one of the renowned researchers made a DPM-based face mask detector using 26, 000 faces divided into masks and without masks category. His work achieved an exceptional accuracy of 97.14%. Further models of face mask detectors were made by Chen et al. [23]. Typically, DPM-based face mask detection models can achieve majestic precisions, but it may suffer from very towering computational cost due to the use of DPM.
iii. CNN-based category:
These types of face detector models learn directly from the dataset provided by the user and then apply several deep learning algorithms on it [24]. In the year 2007 Li et al [25] came up with Cascade CNN. In [17] Yang et al. came up with the idea of features aggregation of faces in the face detection model. In further research works Farfade et al. [15] upgraded the AlexNet architecture for fine tuning the image dataset. For uninhibited circumstances Zhu et al. [6] proposes Contextual Multi-Scale Region-based Convolutional Neural Network (CMS-RCNN) which brought a great impact on the face detection models. To minimize the error on the substitute layers of CNN layers and dealing with the biased obstructions generated in the mask detection models Opitz et al. [13] prepared a grid loss layer. As the technology advanced further CNN-based 3D models started coming up, among them one was proposed by Li et al. [25]. It was an end to end learning structure for face mask detection models. Several other works were done in the fields of pose recognition, gender estimation, localization of landmarks etc.
We have developed our Face mask detection model using deep neural network modules from OpenCV and TensorFlow which contains a Single Shot Multibox Detector object detection model. Typical classification architectures like ResNet-10 which is used as a backbone architecture for this model and for image classification and

fine tuned MobileNetV2 classifier has been used, MobileNetV2 classifier has been an improvement over MobileNetV1 architecture classifier as it consists of a 3×3 convolution layer as the initial layer, which is followed by 13 times the previous building blocks while MobileNetV2 architecture consists of 17 of these building blocks in a row followed by a 1×1 convolution, an average max pooling layer, and a classification layer. **Residual connection** is a new addition in MobileNetV2 classifier.

3. Dataset used

There are only a few datasets available for the detection of face masks. Most of them are either artificially created which doesn't represent real world accurately or the dataset is full of noise and wrong labels. So, to choose the right dataset which would work best for the proposed approach required a little effort. The dataset used in for training the model in given approach was a combination of various open source dataset and pictures which included data from Kaggle's the Medical Mask Dataset by Mikolaj Witkowski and Prajna Bhandandary dataset available at PyImageSearch. Also data was collected using the dataset provided by Masked face recognition dataset and application [27]. The Kaggle dataset contains pictures of people wearing medical masks along with XML files containing their description and the location of masks. This dataset had a total of 678 images. The other Artificial mask dataset was taken from 'Prajna Bhandary' from PyImageSearch. The dataset includes 1,376 images separated into two classes with mask, 690 images and without mask, 686 images. The dataset from Masked face recognition and application contained a lot of noise and a lot repetitions were present in the images of this dataset. Since a good dataset dictates the accuracy of the model trained on it so the data from the above specified datasets were taken. They were then processed and also all the repetitions were removed manually. The data cleaning was done manually to remove the corrupt images which we found in our dataset. Finding these corrupt images was a tricky task but due to valid efforts we divided the work and cleaned the data set with mask images and without mask images. Making a clean dataset was a vital part. As it is well known, identifying and correcting errors in a dataset removes negative effects from any predictive model.

The artificial dataset created by Prajna Bhandary took normal images of faces and applied facial landmarks. Facial landmarks allowed to locate facial features of a person like eyes, eyebrows, nose, mouth, and jawline. This used an artificial way to create a dataset by including a mask on a non-masked person image but those images were not again used in the artificial generation process. The use of non-face mask samples, involved the risk of the model becoming heavily biased. It was a risk to use such a dataset but we included more images from various other sources which would in fact consist of person images with mask and unmasked which compensated for the error correction.

In the end we got an dataset which included 5521 images having label "*with_mask*" and "*without_mask*" also contained 5512 images to make a balanced dataset. A databatch created using this dataset is shown in Figure 1 and the distribution between the two classes is visualized in Figure 2. The created dataset can also be used for detecting assailants who cover their faces while performing unlawful deeds. The Dataset has been made made available at: https://github.com/TheSSJ2612/Real-Time-Medical-Mask-Detection/releases/download/v0.1/Dataset.zip

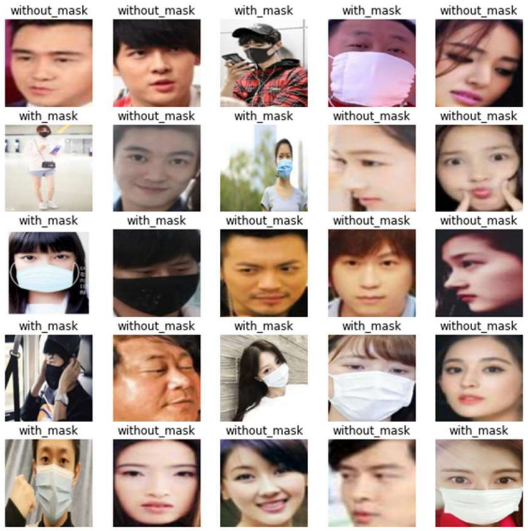

Figure 1: Dataset with mask and without mask

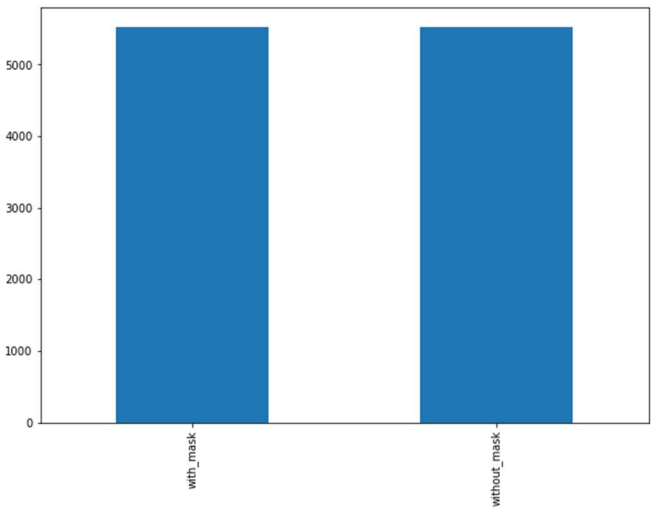

Figure 2: Bar Graph of with mask and without mask

4. Methodology

To correctly identify whether a person has worn a mask, the first step would be to train the model using a proper dataset. Details about the dataset have been discussed above in section 3. After training the classifier, an accurate face detection model to detect faces is required so that the proposed model can classify whether the person is wearing a mask or not. The task in this paper is to maximize the accuracy for mask detection without being too resource heavy. For doing this task DNN module was used from OpenCV which contains a SSD (Single Shot Multibox Detector) object detection model with ResNet-10 as its backbone architecture. This approach helps in detecting faces in real time even on embedded devices like Raspberry Pi. The following classifier uses a pretrained model MobileNetV2 to predict whether the person is wearing a mask or not. The approach used in this paper is depicted in the flow diagram in Figure 3.

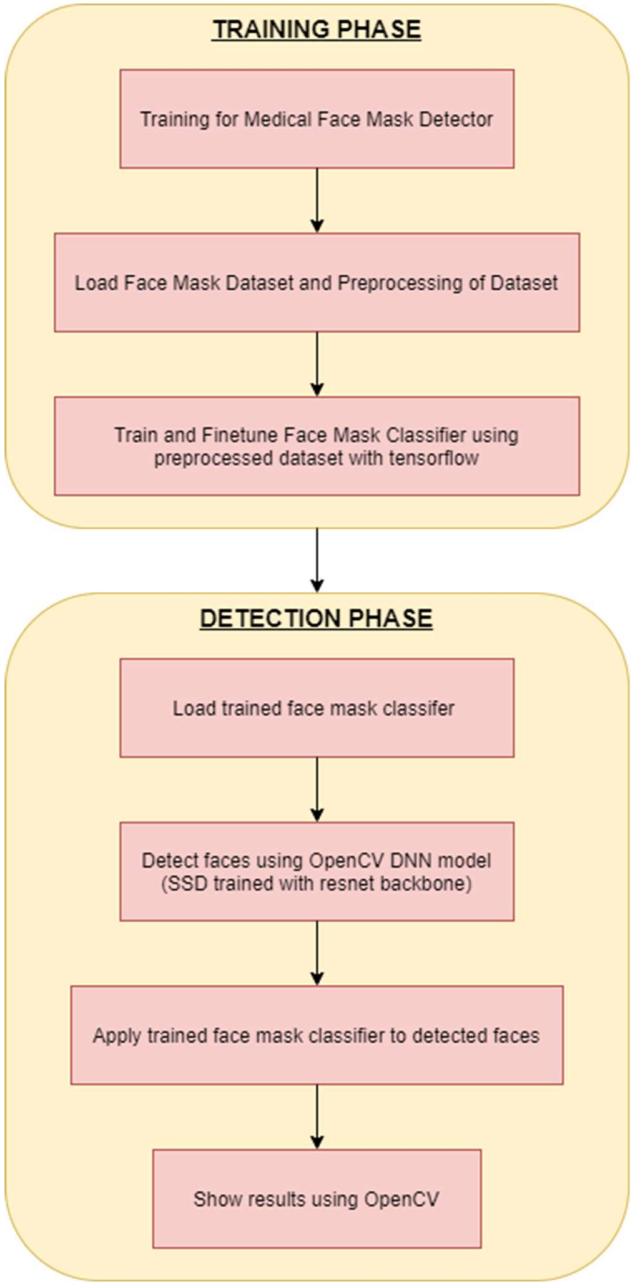

Figure 3: Architecture of our proposed work

4.1 Face Detection using OPEN-CV DNN

This model is included in the GitHub repository of OpenCV starting from version 3.3. It is based on Single Shot multibox Detector (SSD) and uses ResNet-10 architecture as the backbone. The images from which the model is trained have not been disclosed. Two versions of the model are made available by OpenCV:
i.) Original Caffe Implementation (Floating point 16 version)
ii.) TensorFlow Implementation (8-bit quantized version)
caffemodel for our implementation for the proposed approach to detect faces for the detection of facial masks. For this the caffemodel and prototxt files were loaded using *cv2.dnn.readNet("path/to/prototxtfile", "path/to/caffemodelweights").* After applying the face detection model, we get the output of the number of faces detected, the location of their bounding boxes, and the confidence score in those predictions. These outputs are then used as input for the face mask classifier. Using this approach to detect faces allows for real time detection with much resource usage. It is also able to detect faces in different orientations, i.e., left, right, top, and bottom with good accuracy. It is also able to detect faces of different scales, i.e., big, or small.

4.2 Classification of images using MobileNetV2

MobileNetV2 is a Deep Neural Network which we have deployed for the classification problem. Pretrained weights of ImageNet were loaded from TensorFlow. Then the base layers are frozen to avoid impairment of already learned features. Then new trainable layers are added and these layers are trained on the collected dataset so that it can learn the features to classify a face wearing a mask from a face not wearing a mask. Then the model is finetuned and then the weights are saved. Using pretrained models helps in avoiding unnecessary computational costs and also helps in taking advantage of already biased weights with out losing already learned features. This procedure is shown in Figure 4.

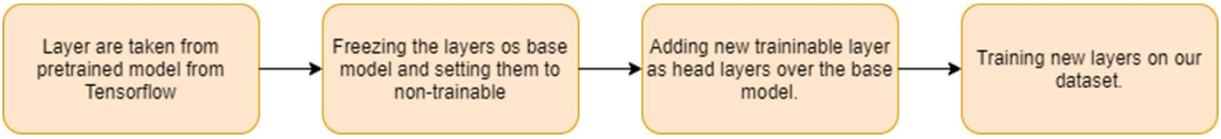

Figure 4: Pipeline of using Pretrained Model

4.2.1 Architecture of MobileNetV2

MobileNetV2 is a Convolutional Neural Network (CNN) based deep learning model which uses the following layers as shown in Table 1.

Table 1: MobileNet V2 Architecture

Input	**Operator**	**Expansion rate of the channels**	**Number of input channels**	**Repetitions**	**Stride**
224^2 x 3	Conv2d	-	32	1	2
112^2 x 32	bottleneck	1	16	1	1
112^2 x 16	bottleneck	6	24	2	2
56^2 x 64	bottleneck	6	32	3	2
28^2 x 32	bottleneck	6	64	4	2
14^2 x 64	bottleneck	6	96	3	1
14^2 x 96	bottleneck	6	160	3	2
7^2 x 160	bottleneck	6	320	1	1
7^2 x 320	Conv2d 1x1	-	1280	1	1

7^2 x 1280	Argpool 7x7	-	-	1	-
1 x 1 x 1280	Conv2d 1x1	-	K	-	

4.2.1.1 Convolutional Layer

This layer is the building block of CNN. It works on a sliding window mechanism, which helps in extracting features from an image. This helps in generation feature maps. The convolution of two matrices, one being the input image matrix of size X*Y*Z and convolutional kernel of size k*k*Z with stride size s and padding e gives us the output of size $(\frac{X-k+2e}{s}+1) \times (\frac{Y-k+2e}{s}+1) \times f$ where Z is the depth of image, and f is the number of filters. The Output of convolution C between matrices P of size (X, Y) and Q of size (A, B) can be expressed as:

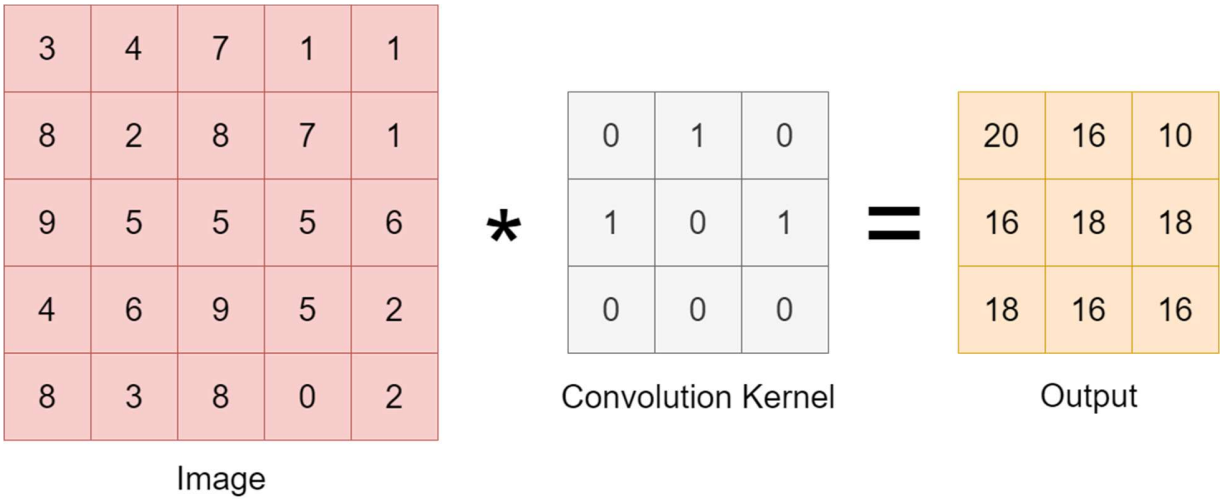

Figure 5: Convolutional Operation

$$C(i,j) = \sum_{x=0}^{X-1} \sum_{y=0}^{Y-1} P(x,y) * Q(i-x, j-q) \quad (1)$$

Where $0 \leq i \leq X + A - 1$ and $0 \leq j \leq Y + B - 1$. Figure 5 shows the convolutional operation.

4.2.1.2 Pooling Layer

Applying the pooling operations helps in speeding up the calculations by allowing reduction in the size of the input matrix without losing many features. Different kind of pooling operations can be applied out of which some are explained below:
i. Max Pooling: It takes the maximum value present in the selected region where the kernel is currently at as the value for output of matrix value for that cell. Figure 6 shows the max-pooling operation.
ii. Average Pooling: It takes the average of all the values that are currently in the region where the kernel is at and takes this value as the output for the matrix value of that cell.

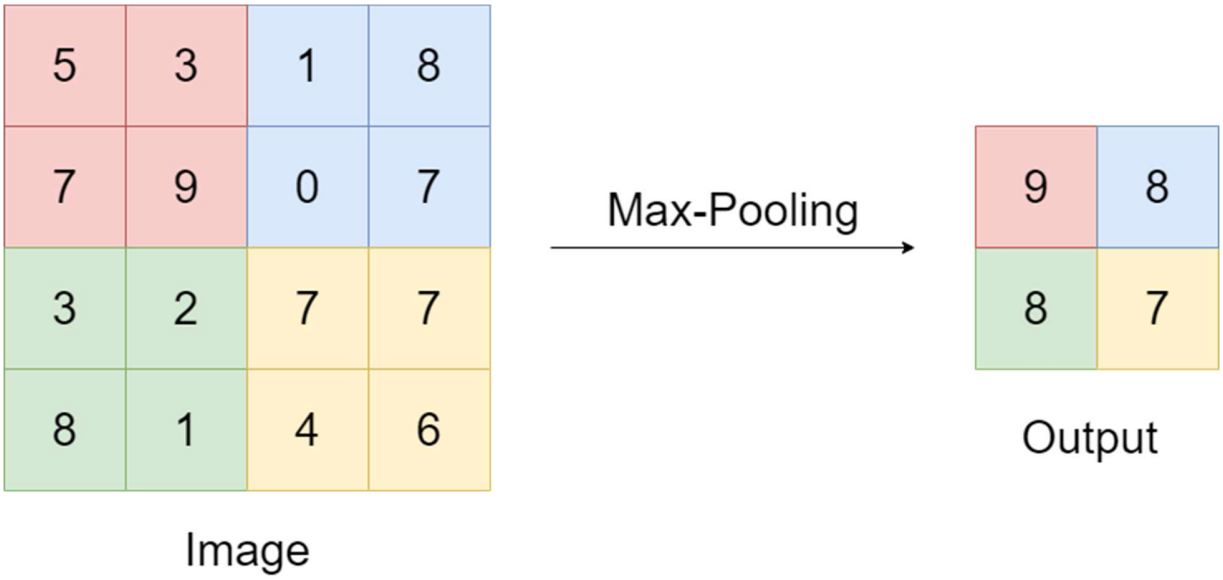

Figure 6: Max-Pooling Operation

4.2.1.3 Dropout Regularization

This helps in reducing the overfitting which may occur while training by dropping random biased neurons from the model. These neurons can be a part of hidden layers as well as visible layers. The likelihood for a neuron to be dropped can be changed by changing the dropout ratio.

4.2.1.4 Non-Linear Layer

These layers usually follow the convolutional layers and are also called activation layers. Most commonly used non-linear functions include different kinds of Rectified Linear Unit (ReLU), i.e., Leaky ReLU, Noisy ReLU, Exponential ReLU, etc., sigmoid function as well as tanh functions. Figure 7 shows different kinds of activation functions.

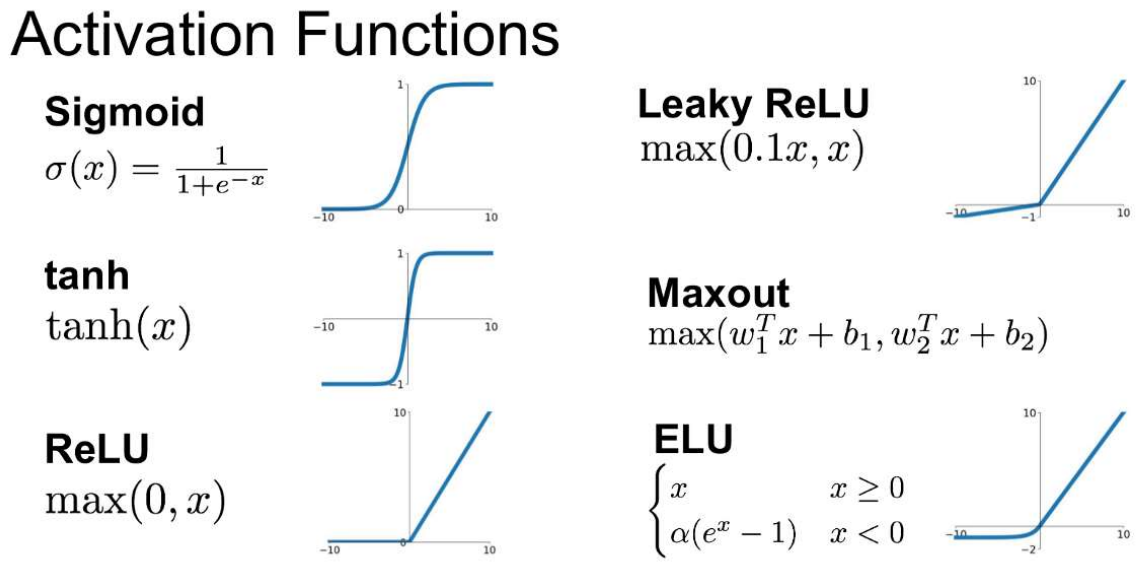

Figure 7: Different Activation Functions
*(source: https://miro.medium.com/max/1400/1*ZafDv3VUm60Eh10OeJu1vw.png)*

4.2.1.5 Fully-Connected Layer

These layers are added at the end of the model and have full connections to activations layers. These layers help in classifying the given images in multi-class or binary classification. SoftMax is an example of activation functions used in these layers and it gives the result of predicted output classes in terms of probability.

4.2.1.6 Linear Bottlenecks

As multiple matrix multiplications cannot be diminished to a single numerical operation so non-linear activation functions like ReLU6 are applied in the neural networks so that several discrepancies can be removed easily. Through this a multilayer neural network can be built. Since ReLU activation function abandons the values which are less than 0. To challenge the loss of information, dimensions of the network can be increased by increasing the number of channels.
For a reversed residual block layers of the blocks are compressed, and the contrary procedure is done as done above. This is done at the point where the skip connections are linked, this could affect the execution of the network. To deal with this, the concept of linear bottleneck was introduced in which before adding the block to initial activation the last convolution of the residual block is given a linear output.

4.3 Algorithms explaining complete pipeline

ALGORITHM 1: Preprocessing and Training the dataset

```
Sorted_Alphanumerically(List)
{
Sort the given list in lexicographical order
}

Preprocessing(Image)
{
Convert image to array tensor
Resize image to 224 x 224 x 3 for our model
Normalize the created tensor to values between -1 and
1 for faster calculations
Return image
}

Load_files(path to folder)
{
Load all files and their paths and labels in a list
Preprocessing(input images)
Sort the list using Sorted_Alphanumerically()
Convert labels to numerical values
Convert list to Numpy array for faster calculations
}

Train_Test_Split(data_array)
{
Split the data into the ratio of 0.2, i.e., 80% data for
training and 20% data for testing
Create train and test batches
Return data_batches
}

Data_Augmentator(data_batches)
{
Augment training data using techniques like rotation,
shearing, flipping, etc.
}

 Create_Model()
{
Load pretrained MobileNetV2 as base model
Train its head while not training the base layers to
avoid losing learned features
Compile the model with Adam optimizer using binary
cross entropy loss function
Return the model
}

 Train_and_Save_Model(createdmodel)
{
Set the values for epochs
Give callbacks like tensor board to log metrics
Train model
Save the model on disk
}

Visualize_Model(modelpath)
{
Plot accuracy and loss curves
Print classification report results
Make predictions on test batches
Plot confusion matrix and roc curves
}
```

ALGORITHM 2: Deployment of Face Mask Detector

```
Detect_mask_image(Input_file)
{
Load face detector model and mask classifier model
Detect faces in image using face detector model
If face are detected
{
Pass cropped faces to mask classification model
Get predictions from model
Show the predictions on image and save the resultant
image
}
Else
{Give output no faces detected}
}
Detect_mask_realtime()
{
Load face detector model and mask classifier model
Start Real time feed
Pass the feed through face detection model
If faces are detected
{
Pass cropped faces to mask  classification model
Get predictions from model
}
If predictions are returned from model
{
Show output in playback output of real time stream
}
End stream when q is pressed
}
Load_Image_or_RealTime(choice)
{
Check whether selected choice is real time detection
or mask detection on image
If Choice = Image
{
Load Image
Detect_mask_image(Input_File)
}
If Choice = RealTime
{
Detect_mask_realtime()
}
}
```

Figure 8 shows the explains the algorithms used in our approach with graphical representation.

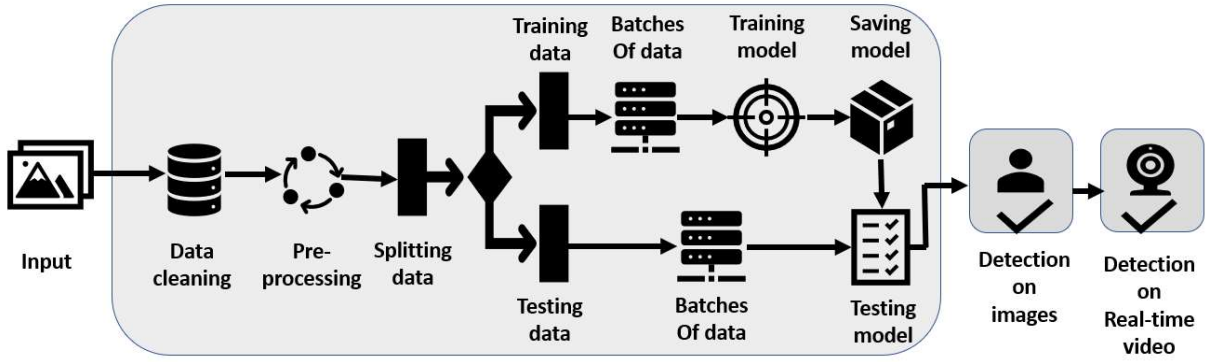

Figure 8: Explanation of Algorithms

5. Experimental Results

To solve the binary classification, deep learning model problem int this paper, Keras is used to make classification model in this paper, which is an advanced-level artificial neural networks API. The evaluation metrics used in this paper are accuracy, area under the ROC (Receiver Operating Characteristics) curve, confusion matrix, classification report and comparison of models. The accuracy gives us the level of correct prediction of masked person identified by the machine by the proposed model as shown in Figure 9. The Roc curve compares a model's true positive rate (tpr) from a model's with false positive rate (fpr) as shown in Figure 12. The model's roc accuracy score is close to the ideal roc curve but not the same. Confusion matrix shown in Figure 11 depicts a matrix to compare the labels, model prediction and actual labels it was supposed to predict. It is showing where the model is getting confused. Classification report shown in Table 2 explains the level of accuracy, precision, recall and f1 score of our model. The average accuracy of our model is '93%' for predicting if a person is wearing a mask or not on a validation dataset as shown in Figure 9. The training loss curve corresponding to training and validation is shown in Figure 10.

Figure 9: Training accuracy curve on train and validation dataset

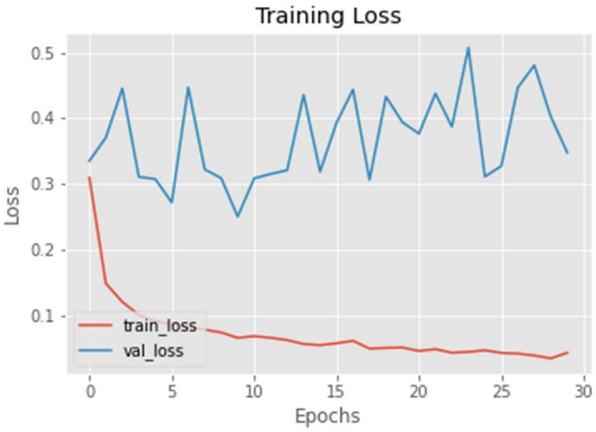

Figure 10: Training loss curve on train and validation dataset

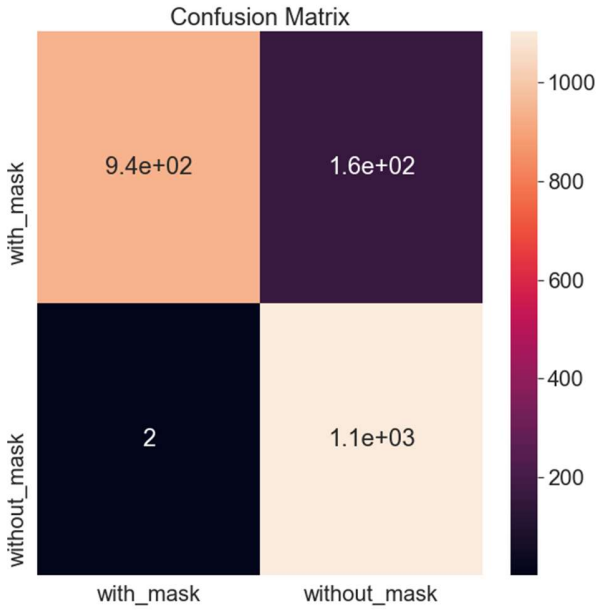

Figure 11: Confusion matrix

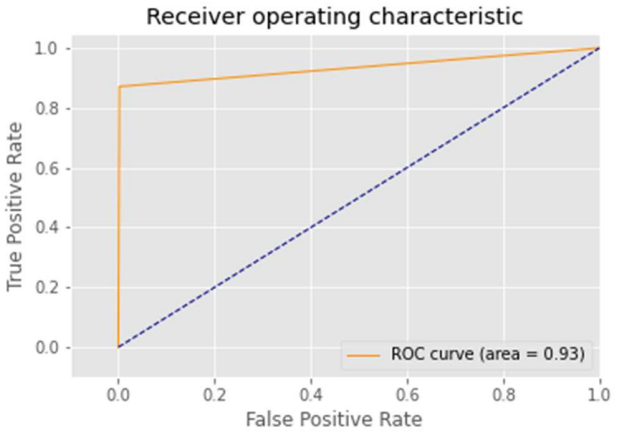

Figure 12: Roc curve

The plots are based on model accuracy, the pyplot command style function that makes matplotlib work like MATLAB. In Figure 9 the red curve shows the training accuracy which is nearly equal to 98%, whereas the blue line represents training accuracy on the validation dataset. Like the previous plot the Figure 10 represents training loss where the red curve shows loss in training dataset less than 0.1, whereas the blue curve shows training loss on the validation dataset. The confusion matrix is plotted with help of heatmap showing a 2D matrix data in graphical format. It has successfully identified 941 true positives, 163 false negatives, 2 false positive and 1103 true negative out of 2209 images used for validation. The Roc curve in Figure 12 shows graphical representation of true positive rate and false positive rate. The roc accuracy score showing 93% predicts the correctness of predicting the model values.
Figure 13 shows the predictions on some images. These are the predictions made on 12 test images by our model using MobileNetv2. The rectangular green box depicts the correct way of wearing a mask with accuracy score on the top left while the red rectangular box represents the incorrect way of wearing a mask. The model learns from the pattern of train dataset and labels and then makes predictions. The classification report of our model is shown in Table 2.

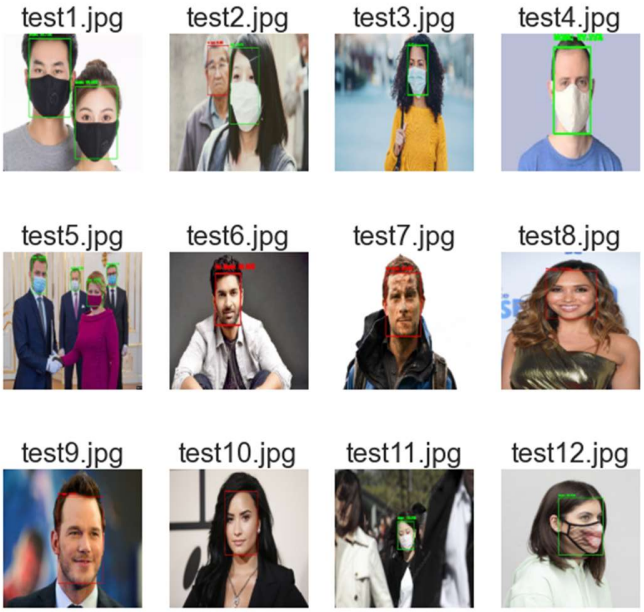

Figure 13: Predictions on test images

Table 2: Classification Report

	Precision	**recall**	**F1 Score**	**support**
with mask	1.00	0.85	0.93	1104
without mask	0.87	1.00	0.93	1105
accuracy			0.93	2209
Macro average	0.94	0.93	0.93	2209
Weighted average	0.94	0.93	0.93	2209

The proposed approach was also compared with different preexisting models by training them on the same dataset and the results from them have been shown in Table 3. In the end MobileNetV2 was chosen to be our model for the proposed approach even though it's accuracy is slightly less than VGG16 and ResNet - 50 since it is easy to deploy in real time even on embedded devices which is not possible with heavy models and to do real time detection using these models requires good computational power which might make it difficult to play in real life

Table 3: Comparison between different models

Method	**Year**	**Accuracy (%)**	**F1 Score**
LeNet -5	1998	85.6	0.86
AlexNet	2012	89.2	0.88
VGG -16	2014	93.21	0.92
ResNet - 50	2016	92.9	0.91
MobileNetV2	**2020**	**92.53**	**0.93**

6. Conclusion

In the proposed face mask detection model, both the training and development of the image dataset which were divided into categories of people having masks and people not having masks have been done successfully. The technique of OpenCV deep neural networks used in this model generated fruitful results. Classification of images was done accurately using Mobilenetv2 image classifier which is one of the uniqueness of the proposed approach.
Many existing researches faced problematic results while some were able to generate better accuracy with their dataset. The problem of various wrong predictions have been successfully removed from the model as the dataset used was collected from various other sources and images used in the dataset was cleaned manually to increase the accuracy of the results. Real world applications are a much more challenging issue for the upcoming future. The proposed face mask detection model should hopefully help the concerned authorities in this great pandemic situation which had largely gained roots in most part of the world, the dataset provided in this paper can be used by other researchers for further advanced models such as those of face recognition, facial landmarks and facial part detection process.

References

[1] Lawrence, Steve, C. Lee Giles, Ah Chung Tsoi, and Andrew D. Back. "Face recognition: A convolutional neural-network approach." *IEEE transactions on neural networks* 8, no. 1 (1997): 98-113.
[2] Velasco-Montero, Delia, Jorge Fernández-Berni, Ricardo Carmona-Galán, and Ángel Rodríguez-Vázquez. "Performance analysis of real-time DNN inference on Raspberry Pi." In *Real-Time Image and Video Processing 2018*, vol. 10670, p. 106700F. International Society for Optics and Photonics, 2018.
[3] NGUYEN, HOANH. "FAST OBJECT DETECTION FRAMEWORK BASED ON MOBILENETV2 ARCHITECTURE AND ENHANCED FEATURE PYRAMID." *Journal of Theoretical and Applied Information Technology* 98, no. 05 (2020).
[4] Chen, Dong, Gang Hua, Fang Wen, and Jian Sun. "Supervised transformer network for efficient face detection." In *European Conference on Computer Vision*, pp. 122-138. Springer, Cham, 2016.
[5] Ranjan, Rajeev, Vishal M. Patel, and Rama Chellappa. "Hyperface: A deep multi-task learning framework for face detection, landmark localization, pose estimation, and gender recognition." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 41, no. 1 (2017): 121-135.
[6] Zhu, Chenchen, Yutong Zheng, Khoa Luu, and Marios Savvides. "Cms-rcnn: contextual multi-scale region-based cnn for unconstrained face detection." In *Deep learning for biometrics*, pp. 57-79. Springer, Cham, 2017.
[7] Li, Yunzhu, Benyuan Sun, Tianfu Wu, and Yizhou Wang. "Face detection with end-to-end integration of a convnet and a 3d model." In *European Conference on Computer Vision*, pp. 420-436. Springer, Cham, 2016.
[8] Zhang, Kaipeng, Zhanpeng Zhang, Zhifeng Li, and Yu Qiao. "Joint face detection and alignment using multitask cascaded convolutional networks." *IEEE Signal Processing Letters* 23, no. 10 (2016): 1499-1503.
[9] Huang, Chang, Haizhou Ai, Yuan Li, and Shihong Lao. "High-performance rotation invariant multiview face detection." *IEEE Transactions on pattern analysis and machine intelligence* 29, no. 4 (2007): 671-686.
[10] Jun, Bongjin, Inho Choi, and Daijin Kim. "Local transform features and hybridization for accurate face and human detection." *IEEE transactions on pattern analysis and machine intelligence* 35, no. 6 (2012): 1423-1436.
[11] I. Shlizerman, E. Shechtman, R. Garg, and S. M. Seitz. Exploring photo-bios. ACM TOG, 30:61, 2011.
[12] Ghiasi, Golnaz, and Charless C. Fowlkes. "Occlusion coherence: Localizing occluded faces with a hierarchical deformable part model." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2385-2392. 2014.
[13] Opitz, Michael, Georg Waltner, Georg Poier, Horst Possegger, and Horst Bischof. "Grid loss: Detecting occluded faces." In *European conference on computer vision*, pp. 386-402. Springer, Cham, 2016.
[14] Yang, Shuo, Ping Luo, Chen-Change Loy, and Xiaoou Tang. "From facial parts responses to face detection: A deep learning approach." In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3676-3684. 2015.
[15] Ojala, Timo, Matti Pietikäinen, and Topi Maenpaa. "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns." *IEEE Transactions on pattern analysis and machine intelligence* 24, no. 7 (2002): 971-987.
[16] Kim, Tae-Hyun, Dong-Chul Park, Dong-Min Woo, Tae Kyeong Jeong, and Soo-Young Min. "Multi-class classifier-based adaboost algorithm." In *International Conference on Intelligent Science and Intelligent Data Engineering*, pp. 122-127. Springer, Berlin, Heidelberg, 2011.
[17] Yang, Bin, Junjie Yan, Zhen Lei, and Stan Z. Li. "Fine-grained evaluation on face detection in the wild." In *2015 11th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG)*, vol. 1, pp. 1-7. IEEE, 2015.

[18] Klare, Brendan F., Ben Klein, Emma Taborsky, Austin Blanton, Jordan Cheney, Kristen Allen, Patrick Grother, Alan Mah, and Anil K. Jain. "Pushing the frontiers of unconstrained face detection and recognition: Iarpa janus benchmark a." In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1931-1939. 2015.
[19] Klare, Brendan F., Ben Klein, Emma Taborsky, Austin Blanton, Jordan Cheney, Kristen Allen, Patrick Grother, Alan Mah, and Anil K. Jain. "Pushing the frontiers of unconstrained face detection and recognition: Iarpa janus benchmark a." In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1931-1939. 2015.
[20] Yang, Shuo, Ping Luo, Chen-Change Loy, and Xiaoou Tang. "Wider face: A face detection benchmark." In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5525-5533. 2016.
[21] Viola, Paul, and Michael Jones. "Rapid object detection using a boosted cascade of simple features." In *Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition. CVPR 2001*, vol. 1, pp. I-I. IEEE, 2001.
[22] Zhang, Kaipeng, Zhanpeng Zhang, Zhifeng Li, and Yu Qiao. "Joint face detection and alignment using multitask cascaded convolutional networks." *IEEE Signal Processing Letters* 23, no. 10 (2016): 1499-1503.
[23] Chen, Dong, Shaoqing Ren, Yichen Wei, Xudong Cao, and Jian Sun. "Joint cascade face detection and alignment." In *European conference on computer vision*, pp. 109-122. Springer, Cham, 2014.
[24] Ren, Shaoqing, Kaiming He, Ross Girshick, and Jian Sun. "Faster r-cnn: Towards real-time object detection with region proposal networks." In *Advances in neural information processing systems*, pp. 91-99. 2015.
[25] Li, Haoxiang, Zhe Lin, Xiaohui Shen, Jonathan Brandt, and Gang Hua. "A convolutional neural network cascade for face detection." In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5325-5334. 2015.
[26] Abadi, Martín, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado et al. "Tensorflow: Large-scale machine learning on heterogeneous distributed systems." *arXiv preprint arXiv:1603.04467* (2016).
[27] Wang, Zhongyuan, Guangcheng Wang, Baojin Huang, Zhangyang Xiong, Qi Hong, Hao Wu, Peng Yi et al. "Masked face recognition dataset and application." *arXiv preprint arXiv:2003.09093* (2020).