LAB ASSIGNMENT-4

COVID FACE MASK DETECTION USING CNN with Regularization Techniques

BY,

MADASU DEEPIKA
19MIA1066

Abstract:
In recent trend, In world-wide lockdowns has been imposed due to COVID19 outbreak and Face Mask became mandatory for everyone while roaming outside.
This paper approaches Deep Learning for detecting Faces with and without mask.

While going through the pandemic and the post pandemic situations wearing a mask are compulsory for everyone in order to prevent the transmission of corona virus. This resulted in ineffectiveness of the existing conventional face recognition systems.

Objective:

The main aim is to identify that whether a person's face is covered with mask or not as per the CCTV camera surveillance or a webcam recording. It keeps on checking if a person is wearing mask or not. For classification, feature extraction and detection of the masked faces, Convolutional Neural Network (CNN) are used.

These help in easy detection of masked faces with higher accuracy in a very less time and with high security.

Keywords: Covid-19, Deep Learning, Face mask, Convolution Neural Network.

Introduction:

Effective methods to restrain COVID-19 pandemic want high attention to mitigate negatively wedged communal health and world economy, with the brim-full horizon however to unfold within the absence of effective antiviral and restricted medical resources, several measures area unit suggested by World Health Organization to regulate the infection rate.

To contribute towards communal health, this paper aims to plot a extremely correct and period technique that may expeditiously discover non-mask faces publicly and therefore, imposing to wear mask. In this paper simplified approach to serve the above purpose using the basic Deep Learning(DL) packages such as TensorFlow, Keras, and Scikit-Learn. The proposed method detects the face from the image correctly and then identifies if it has a mask on it or not.

Dataset:

This dataset contains about 1006 equally distributed images of 2 distinct types.
This dataset consists of **2012 images** belonging to two classes:

- **with_mask: 1006 images**
- **without_mask: 1006 images**

All the images are actual images extracted from Bing SearchAPI, Kaggle datasets, and RMFD datasets. From all three sources, the proportion of the images is equal. The images cover diverse races i.e Asian, Caucasian, etc.

Dataset link: COVID Face Mask Detection Dataset | Kaggle

Fig-1 Some images of with mask dataset

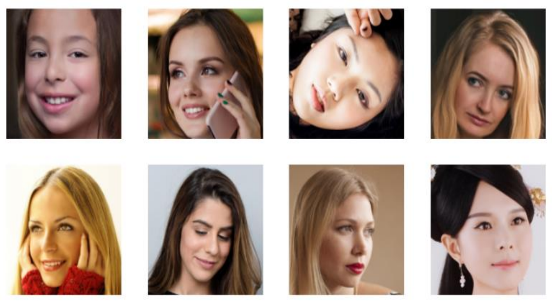


Fig-2 Some images of without mask dataset

Methodology:

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. 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 test set is there a remaining subset of data used to provide an unbiased evaluation of a final model fit on the training dataset.

In our approach, we have dedicated to use data augmentation and rescale the images present in our test-set, validation-set and test set.

A Sequential model is appropriate for a plain stack of layers where each layer has exactly one input tensor and one output tensor.

By using the sequential method, the architecture of the model is built layer by layer, convolution layer, maxpool layer, flatten layer, drop out, dense layer.

Architecture:

The model is created with the following layers.

 The input image is grayscale. since there is only one channel.

 The size of the image is 48 *48

 Convolution layer with the images as the input, with the 'relu' activation Function, padding is valid.

 Next, the Maxpooling layer is created with the size of 2*2 and the stride of 2*2, with the valid padding.

 Then the output of the max-pooling layer is normalized with the Batch normalization layer.

 In the middle of the layers, a Dropout is used to overcome the overfitting of the model.

 In this way, another 2 layers of convolution, max pooling followed by batch normalization is done.

 Then the flattened layer is implemented. This layer flattens all the outputs of the Batch Normalization layer.

 Then the flattened output is given to the next created dense layer.

 The Dense Layer is created after the flatten this will create a fully connected layer that is used for the classification.

 Then the drop-out layer is created to reduce the number of outputs from layer to layer.

 Then a Batch Normalization is created.

 In this way another 2 layers of dense, dropout followed by Batch normalization is created.

 Then at last the last layer a dense layer is created with the 2 neurons, with the activation function 'Softmax'.

 The last layer gives the output which specifies one of the 2 classes i.e., whether a person is with a mask or without a mask.

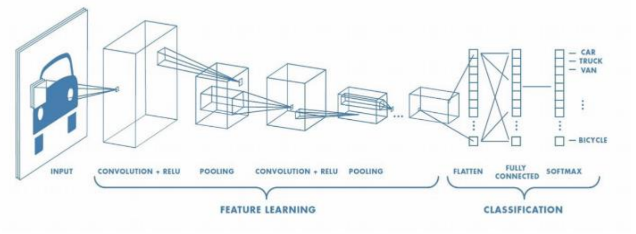
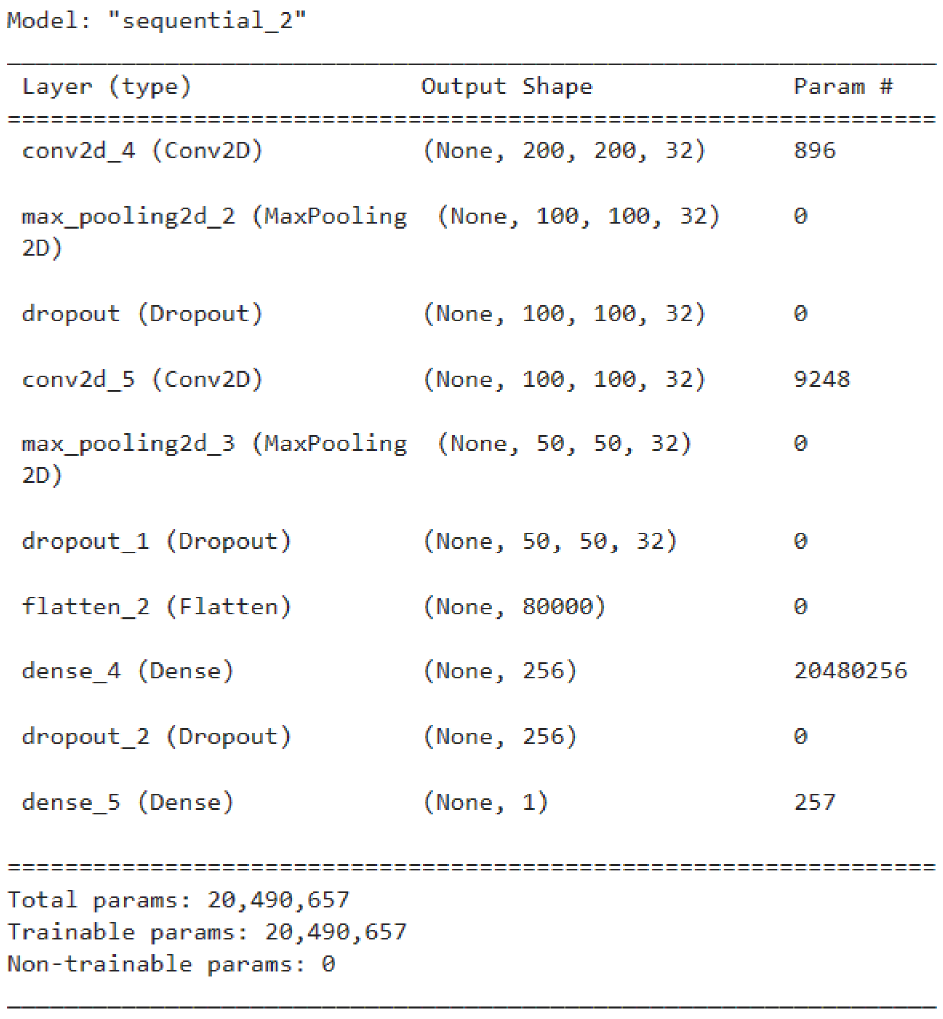


Figure 3

CNN Model:

Pertaining to our CNN model, we trained a CNN with 10 layers and 5, 50 and 100 epochs. A summary of the model and training accuracy across each epochs can be found in Figures 4, 5, 6 & 7.

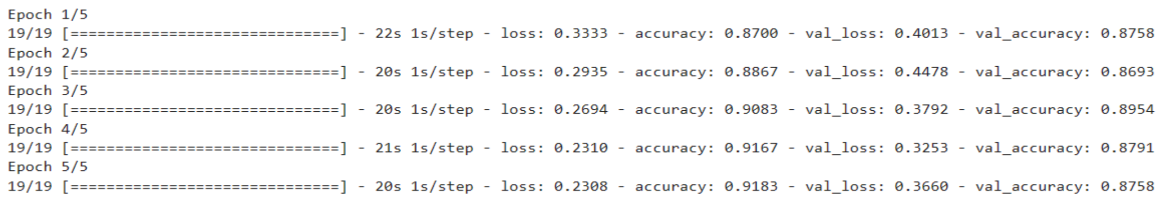


```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_4 (Conv2D)           (None, 200, 200, 32)      896

 max_pooling2d_2 (MaxPooling  (None, 100, 100, 32)     0
 2D)

 dropout (Dropout)           (None, 100, 100, 32)      0

 conv2d_5 (Conv2D)           (None, 100, 100, 32)      9248

 max_pooling2d_3 (MaxPooling  (None, 50, 50, 32)       0
 2D)

 dropout_1 (Dropout)         (None, 50, 50, 32)        0

 flatten_2 (Flatten)         (None, 80000)             0

 dense_4 (Dense)             (None, 256)               20480256

 dropout_2 (Dropout)         (None, 256)               0

 dense_5 (Dense)             (None, 1)                 257

=================================================================
Total params: 20,490,657
Trainable params: 20,490,657
Non-trainable params: 0
_________________________________________________________________
```

Figure 4

DROP OUT + EARLY STOPPING:

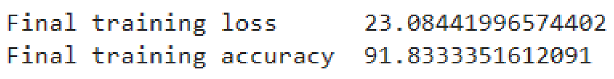
1. For epoch=5



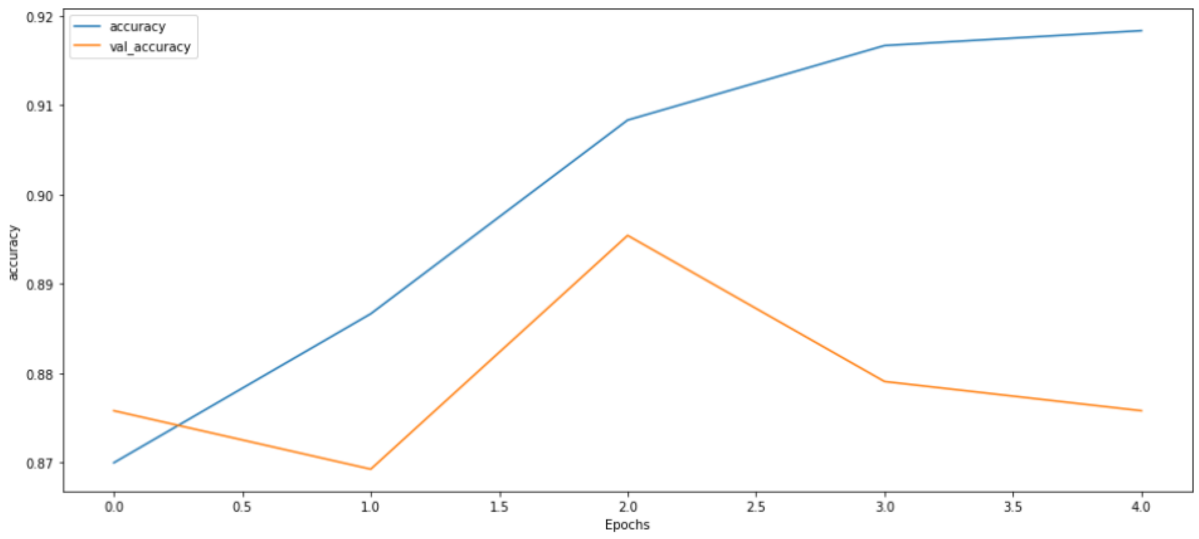
```
Epoch 1/5
19/19 [==============================] - 22s 1s/step - loss: 0.3333 - accuracy: 0.8700 - val_loss: 0.4013 - val_accuracy: 0.8758
Epoch 2/5
19/19 [==============================] - 20s 1s/step - loss: 0.2935 - accuracy: 0.8867 - val_loss: 0.4478 - val_accuracy: 0.8693
Epoch 3/5
19/19 [==============================] - 20s 1s/step - loss: 0.2694 - accuracy: 0.9083 - val_loss: 0.3792 - val_accuracy: 0.8954
Epoch 4/5
19/19 [==============================] - 21s 1s/step - loss: 0.2310 - accuracy: 0.9167 - val_loss: 0.3253 - val_accuracy: 0.8791
Epoch 5/5
19/19 [==============================] - 20s 1s/step - loss: 0.2308 - accuracy: 0.9183 - val_loss: 0.3660 - val_accuracy: 0.8758
```

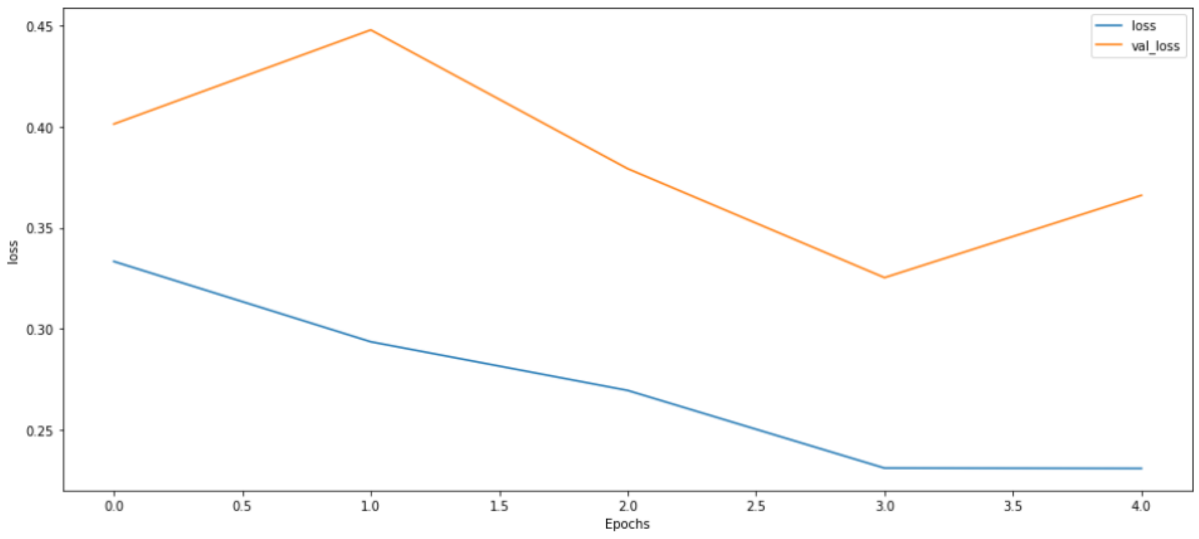
Figure 5

Training and Validation accuracy and loss:

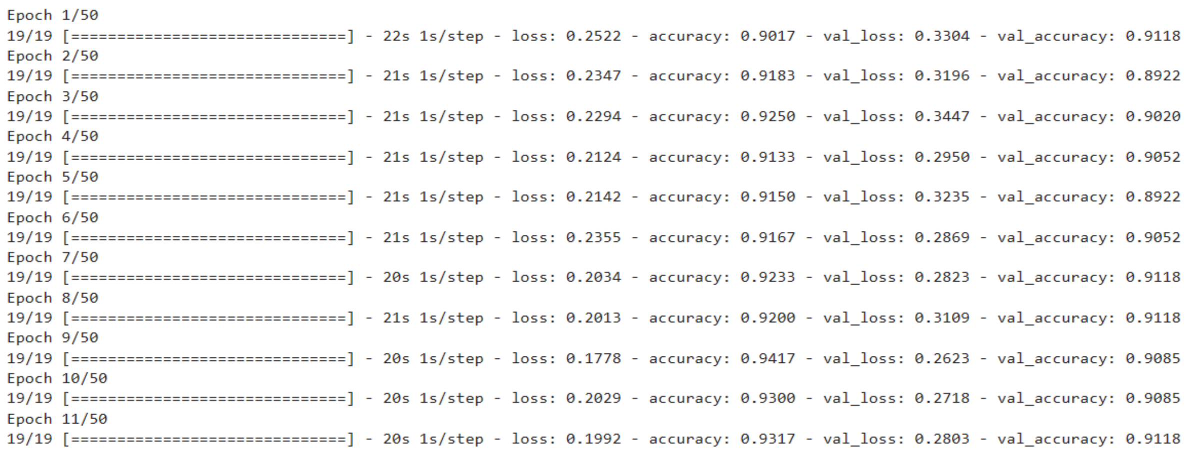


```
Final training loss       23.08441996574402
Final training accuracy   91.8333351612091
```





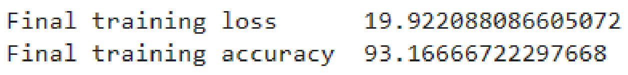
2. For epoch=50



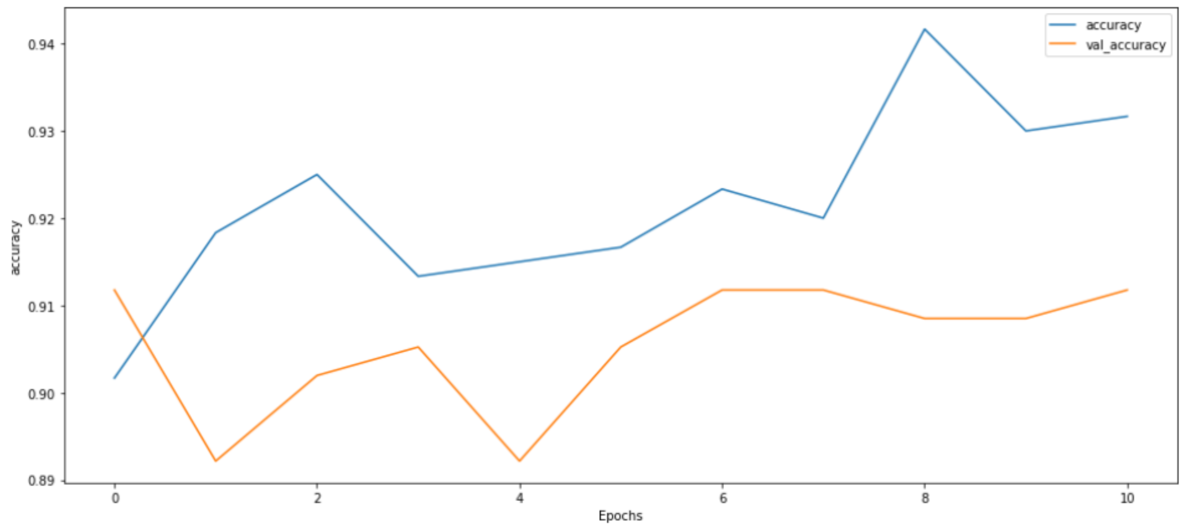
```
Epoch 1/50
19/19 [==============================] - 22s 1s/step - loss: 0.2522 - accuracy: 0.9017 - val_loss: 0.3304 - val_accuracy: 0.9118
Epoch 2/50
19/19 [==============================] - 21s 1s/step - loss: 0.2347 - accuracy: 0.9183 - val_loss: 0.3196 - val_accuracy: 0.8922
Epoch 3/50
19/19 [==============================] - 21s 1s/step - loss: 0.2294 - accuracy: 0.9250 - val_loss: 0.3447 - val_accuracy: 0.9020
Epoch 4/50
19/19 [==============================] - 21s 1s/step - loss: 0.2124 - accuracy: 0.9133 - val_loss: 0.2950 - val_accuracy: 0.9052
Epoch 5/50
19/19 [==============================] - 21s 1s/step - loss: 0.2142 - accuracy: 0.9150 - val_loss: 0.3235 - val_accuracy: 0.8922
Epoch 6/50
19/19 [==============================] - 21s 1s/step - loss: 0.2355 - accuracy: 0.9167 - val_loss: 0.2869 - val_accuracy: 0.9052
Epoch 7/50
19/19 [==============================] - 20s 1s/step - loss: 0.2034 - accuracy: 0.9233 - val_loss: 0.2823 - val_accuracy: 0.9118
Epoch 8/50
19/19 [==============================] - 21s 1s/step - loss: 0.2013 - accuracy: 0.9200 - val_loss: 0.3109 - val_accuracy: 0.9118
Epoch 9/50
19/19 [==============================] - 20s 1s/step - loss: 0.1778 - accuracy: 0.9417 - val_loss: 0.2623 - val_accuracy: 0.9085
Epoch 10/50
19/19 [==============================] - 20s 1s/step - loss: 0.2029 - accuracy: 0.9300 - val_loss: 0.2718 - val_accuracy: 0.9085
Epoch 11/50
19/19 [==============================] - 20s 1s/step - loss: 0.1992 - accuracy: 0.9317 - val_loss: 0.2803 - val_accuracy: 0.9118
```

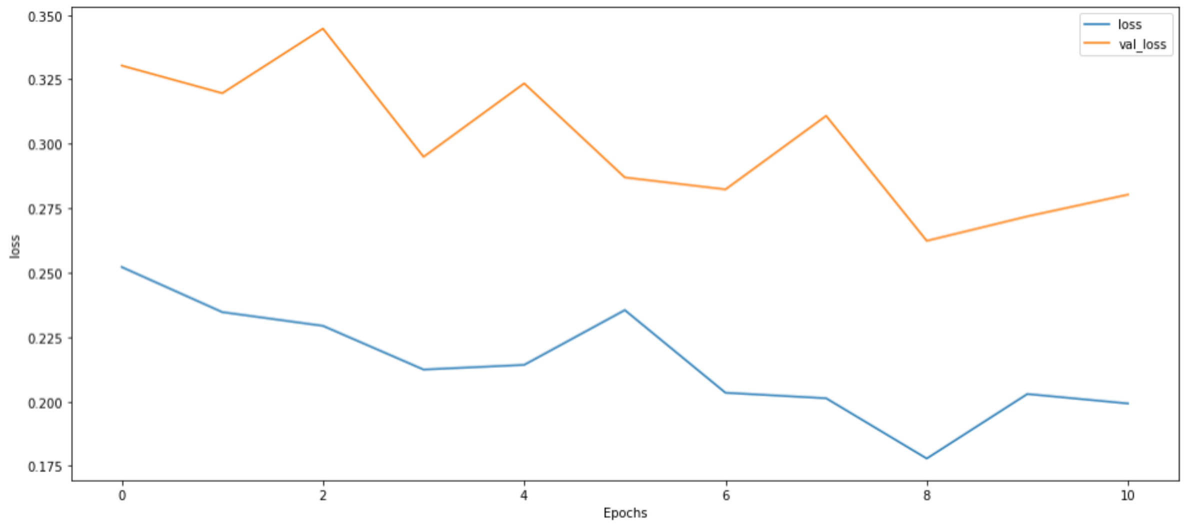
Figure 6

Training and Validation accuracy and loss:



```
Final training loss       19.922088086605072
Final training accuracy   93.16666722297668
```





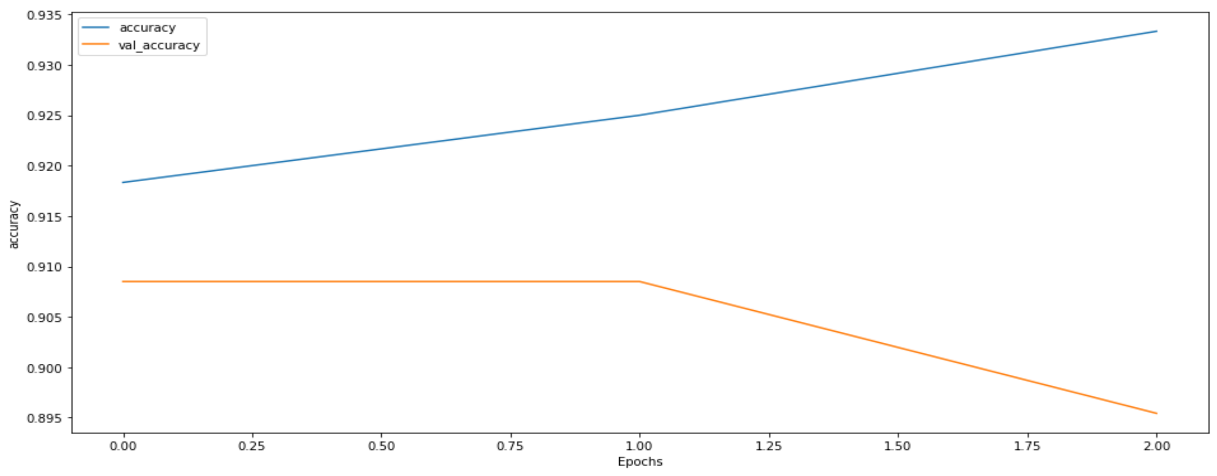
3. For epoch=100

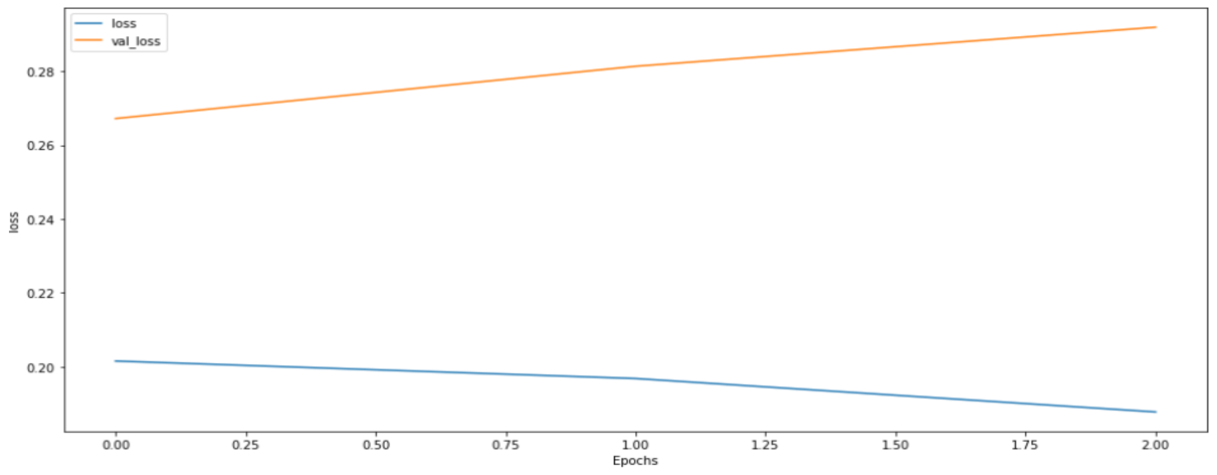
```
Epoch 1/100
19/19 [==============================] - 21s 1s/step - loss: 0.2015 - accuracy: 0.9183 - val_loss: 0.2672 - val_accuracy: 0.9085
Epoch 2/100
19/19 [==============================] - 20s 1s/step - loss: 0.1968 - accuracy: 0.9250 - val_loss: 0.2813 - val_accuracy: 0.9085
Epoch 3/100
19/19 [==============================] - 20s 1s/step - loss: 0.1878 - accuracy: 0.9333 - val_loss: 0.2919 - val_accuracy: 0.8954
```

Figure 7

Training and Validation accuracy and loss:

```
Final training loss      18.77649277448654
Final training accuracy  93.33333373069763
```





Result:

CNN MODEL with	Accuracy Score	Loss Score
2 layers Cnn	88.99	28.18
Adding Max Pooling	90.16	26.95
Early Stopping	93.66	19.00
Adding Dropout	87.00	33.24
Drop out + Early Stopping		
1. Epoch = 5	91.833	23.08
2. Epoch = 50	93.166	19.92
3. Epoch = 100	93.333	18.77

As the model trains, the final training loss and accuracy metrics are displayed. This model reaches an highest accuracy of about 93.66% on the training data and the loss of 19.00%. Accuracy keeps increasing with addition of max pooling layer and also with increase in number of epochs. Then early stopping is used to stop the model as soon as it gets overfitted where as dropout layer is one of the most popular regularization techniques to **reduce overfitting**. After doing regularization techniques the validation accuracy is kept reducing and model is getting over fit but accuracy increases with increase in epochs.

Conclusion:

In this paper, we implemented an application to develop face mask detection using the Deep Convolutional Neural Network model. That is to build a CNN-based framework to precisely match both still images and then check whether a person is wearing a mask or not.

The model should hopefully help the concerned authorities in this great pandemic situation which had largely gained roots in most of the world; other researchers can use the dataset provided in this paper for further advanced models such as those of face recognition, facial landmarks, and facial part detection process.

References:

- Abadi M., Agarwal A., Barham P., Brevdo E., Chen Z., Citro C.…Ghemawat S. 2016. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. arXiv preprint arXiv:1603.04467. [Google Scholar]
- Ahmed I., Ahmad M., Rodrigues J.J., Jeon G., Din S. A deep learning-based social distance monitoring framework for COVID-19. *Sustainable Cities and Society.* 2020 [PMC free article] [PubMed] [Google Scholar]
- Alom M.Z., Taha T.M., Yakopcic C., Westberg S., Sidike P., Nasrin M.S.…Asari V.K. 2018. The history began from alexnet: A comprehensive survey on deep learning approaches. arXiv preprint arXiv:1803.01164. [Google Scholar]
- Anisimov D., Khanova T. Towards lightweight convolutional neural networks for object detection. 2017 14th IEEE international conference on advanced video and signal based surveillance (AVSS); IEEE; 2017. pp. 1–8. August. [Google Scholar]
- Chen D., Ren S., Wei Y., Cao X., Sun J. *European conference on computer vision.* Springer; Cham: 2014. Joint cascade face detection and alignment; pp. 109–122. September. [Google Scholar]
- Chen D., Hua G., Wen F., Sun J. *European conference on computer vision.* Springer; Cham: 2016. Supervised transformer network for efficient face detection; pp. 122–138. October. [Google Scholar]
- Ge S., Li J., Ye Q., Luo Z. Detecting masked faces in the wild with lle-cnns. *Proceedings of the IEEE conference on computer vision and pattern recognition.* 2017:2682–2690. [Google Scholar]
- Ge X.Y., Pu Y., Liao C.H., Huang W.F., Zeng Q., Zhou H.…Chen H.L. Evaluation of the exposure risk of SARS-CoV-2 in different hospital environment. *Sustainable Cities and Society.* 2020;61 [PMC free article] [PubMed] [Google Scholar]
- Ghiasi G., Fowlkes C.C. Occlusion coherence: Localizing occluded faces with a hierarchical deformable part model. *Proceedings of the IEEE conference on computer vision and pattern recognition.* 2014:2385–2392. [Google Scholar]
- Gulcehre C., Moczulski M., Denil M., Bengio Y. *International conference on machine learning.* 2016. Noisy activation functions; pp. 3059–3068. June. [Google Scholar]