# FACE MASK DETECTION USING NEURAL NETWORKS (CNN, VGG16, RESNET50, INCEPTIONV3)

Submited To : Sir Dr Syed M.Naqi

CV Project report

Submited by: moeez kokab
Moeez.kokab@gmail.com

# Face Mask Detection using Neural Networks

## Cnn, Vgg16, ResNet50, InceptionV3

**Moeez Kokab**

**Quaid-i-Azam University,**

**Islamabad ,**

**Pakistan.**

# Contents

# Face Mask Detection using Neural Networks ( Cnn, Vgg16, ResNet50, InceptionV3)


## Abstract

As we have been seeing the exponential growth in the spread of covid-19, one of the major causes was the direct contact with the infected ones. For example, a family living in the same building has meals together and does their things altogether. And also a huge mass of people visiting the shopping malls and other public places. It was set necessary for everyone to put on face masks while being entered in public places. It is quite impossible to check each and every person at the entrance if they have their face masks put on or not. So, there was a need to build such a system that could automatically detect the people with or without the face masks.


## Introduction

Coronavirus disease 2019, also known as COVID-19, is a major issue for everyone around the world. It is an infectious disease that has had an impact on human life all across the world. According to health experts, the virus can spread through direct or indirect contact with an infected person, which is why medical bodies have imposed rigorous measures such as the wearing of face masks as depicted. Even if a person is not sick, several studies recommend wearing a face mask. It isn't the first time that wearing face masks has been emphasized as a means of preventing transmission during COVID-19. Face masks have been used to survive several pandemics throughout history. Furthermore, several studies have shown that using face masks properly, rather than just wearing them, reduces the virus's transmission to a significant level. The necessity for an automatic face mask detector arose from the discovery that the higher the proportion of the population wearing face masks in a country, the lower the number of instances of COVID-19 in the country.

**Table 1.** Illustrated country-wise information about COVID-19 such as confirmed cases, recovered cases, and total number of deaths (accessed on 2 September 2021).

| | Worldometer [8] | | | Google News COVID-19 [7] | |
|---|---|---|---|---|---|
| Countries | Confirmed | Recovered | Deaths | Confirmed | Deaths |
| Worldwide | 216,918,733 | 193,849,589 | 4,511,302 | 216,026,420 | 4,495,014 |
| United States | 39,617,417 | 30,812,242 | 654,381 | 38,830,051 | 637,066 |
| Italy | 4,524,292 | 4,255,808 | 129,056 | 4,524,292 | 129,056 |
| Germany | 3,933,569 | 3,726,700 | 92,631 | 3,933,585 | 92,136 |
| India | 32,695,030 | 31,888,642 | 437,860 | 32,695,030 | 437,830 |
| Saudi Arabia | 543,796 | 531,733 | 8526 | 543,318 | 8512 |
| UAE | 716,381 | 702,102 | 2038 | 715,394 | 2036 |
| China | 94,819 | 88,924 | 4636 | 94,765 | 4636 |
| Pakistan | 1,152,481 | 1,033,373 | 25,604 | 1,152,481 | 25,604 |

COVID-19 signs can indicate a condition that ranges from mild to life-threatening. The incubation period for COVID-19 symptoms lasts between 2 and 15 days following infection. Anyone with COVID-19 symptoms is maintained in quarantine (isolation wards) under active medical supervision. Shortness of breath, sore throat, cough, rapid heartbeat, chest pain, and fever are the most common symptoms of the condition. The virus is commonly transmitted from one person to another through respiratory droplets created during coughing, as well as physical contact, such as touching dirty surfaces and subsequently touching someone's face (Centers for Disease Control and Prevention, 2020).

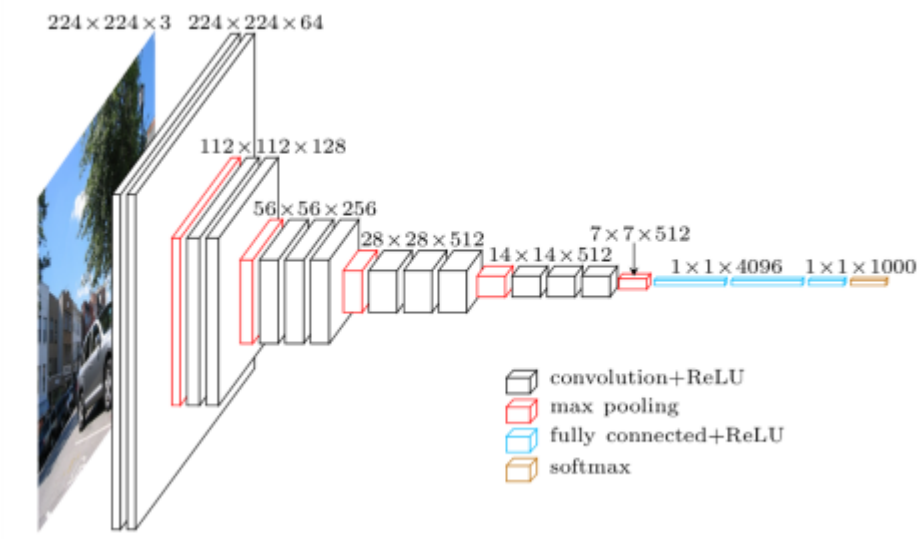
The problem was raised by the capitalists and the health ministers around the world after carrying out the research that the use of face masks could lessen the spread of the epidemic. The main thing was how to make the people do it. Because it was not possible for the security guards or other authorities to force them to do so. Keeping in mind this scenario, we came up with the idea, “what if we make an application to detect the cases automatically?” So, here we propose our different approaches to handle the situation.

## Literature Review

Convolutional Neural Networks (CNN) is now used in a variety of ways to assist humans and protect their lives from harm, including fire disasters, facial feature analysis, healthcare, and many other fields. In addition, a resource-constrained device has been used in several studies to interact in real-time for human life facilitation. The main focus of this research is on using CNN and different other models like ResNet50, Vgg16, and InceptionV3 to detect face masks for COVID-19 prevention. This study, on the other hand, is focused on detecting people who do not wear face masks in public places in order to help reduce COVID-19 transmission. COVID-19 spread is significantly reduced when face masks are worn in public places, according to scientists and researchers.

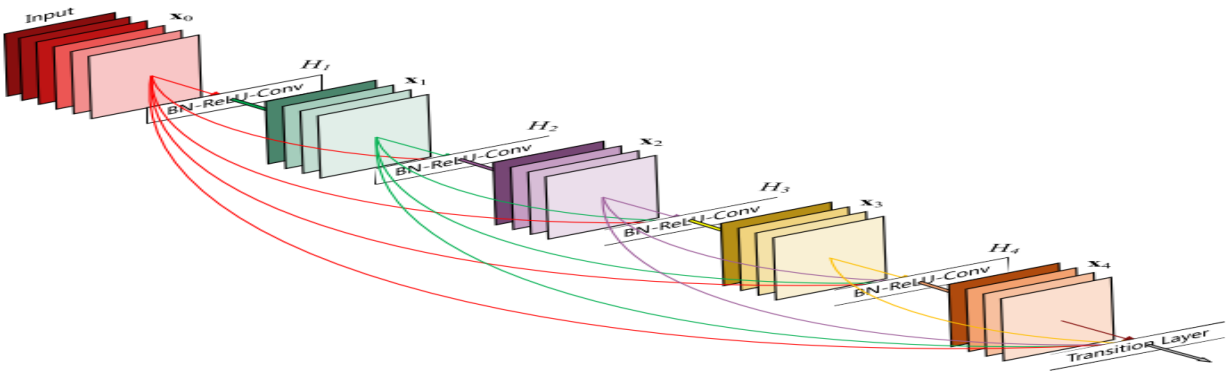
## VGG16

VGGNet-16 consists of 16 convolutional layers and is very appealing because of its very uniform Architecture. Similar to AlexNet, it has only 3x3 convolutions, but lots of filters. It can be trained on 4 GPUs for 2–3 weeks. It is currently the most preferred choice in the community for extracting features from images. The weight configuration of the VGGNet is publicly available and has been used in many other applications and challenges as a baseline feature extractor.



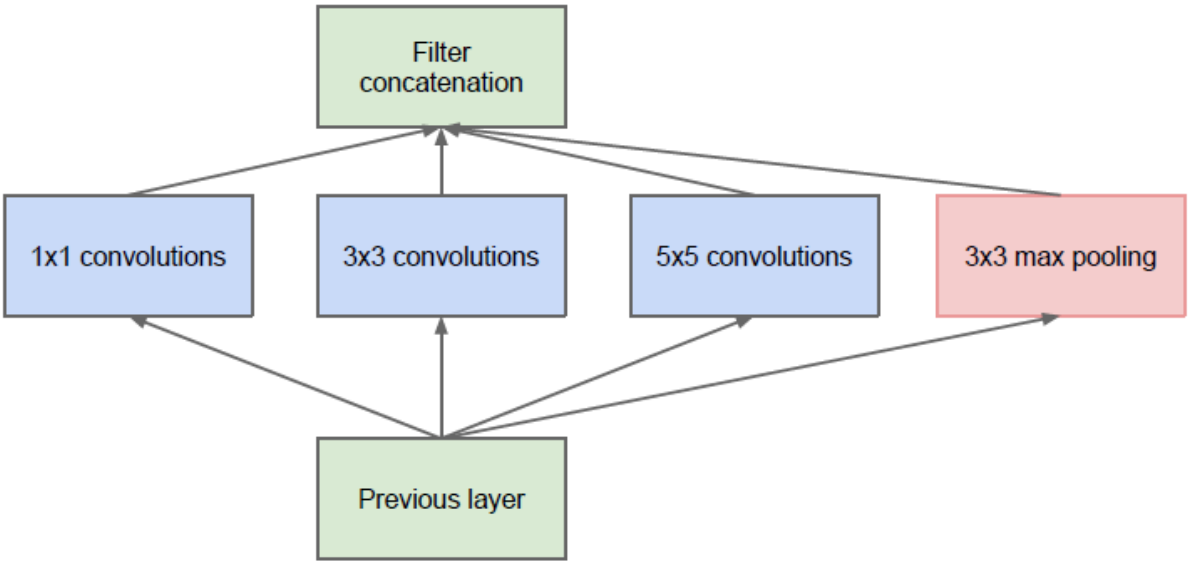

## ResNet50

ResNet, short for Residual Networks is a classic neural network used as a backbone for many computer vision tasks. The original model was the winner of the ImageNet challenge in 2015. ResNet50 v1.5 is the modified version of the original ResNet 50. The fundamental breakthrough with ResNet was it allowed us to train extremely deep neural networks with 150+layers successfully. Prior to ResNet training very deep neural networks were difficult due to the problem of vanishing gradients.The authors addressed this problem by introducing deep residual learning framework so for this they introduce shortcut connections that simply perform identity mappings. The benefit of these shortcut identity mapping was that there was no additional parameters added to the model and also the computational time was kept in check.



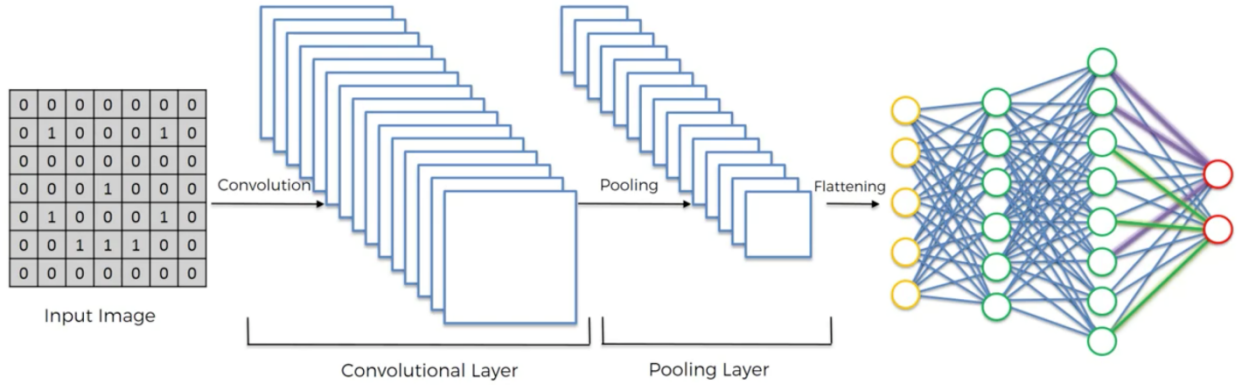

## InceptionV3

The Inception V3 is a deep learning model based on Convolutional Neural Networks, which is used for image classification. The inception V3 is a superior version of the basic model Inception V1 which was introduced as GoogLeNet in 2014. As the name suggests it was developed by a team at Google.




## CNN

Convolutional Neural Network (CNN) is a neural network that extracts or identifies a feature in a particular image. This forms one of the most fundamental operations in Machine Learning and is widely used as a base model in majority of Neural Networks like GoogleNet, VGG19, and others for various tasks such as Object Detection, Image Classification, and others.




## Methods

We gathered the dataset. 5000 images labeled 'With Mask' and 5000 images labeled as 'Without Mask'. The next step was to choose the tool so we used Jupiter Notebook to work. We used *CNN*, *ResNet50*, *Vgg16,* and *InceptionV3* to handle this problem. Following are the steps carried out.

1. Import all the libraries (all libraries are given below )
2. Upload the data
3. Preprocess the data
   - Set the size of the input images (10,000 images)
   - Split the dataset i.e. 80% training (8,000 images) and 20% testing (2,000 images)
4. Train the model {model are cnn,vgg16,resnet50,inceotion}
5. Run the model
6. Plot the accuracies and losses
7. Take the image from the directory to test
8. Make a prediction for the image

## Dataset

This dataset is used for face mask detection The dataset is composed of 10k images with almost 300 MB in size. This dataset is divided into two classes. One class is “with mask” and another one is “without a mask”. Some examples of data sets are given below.(Link: dataset link of Kaggle )

### Example of dataset “ with mask ”



### Example of dataset “ without mask ”



## Programing Language Tool

- Name of Language : Python
- Version of Language : Python 3.9.7
- IDE Name: Anaconda -> Jupiter notebook
- Jupiter notebook Version : notebook 6.4.5
- Package / library : Numpy (np) , Matplotlib (plt) , Tesnsorflow(tf) , Tkinter (tk) , pathlib , keras , layers , Sequential , filedialog

## Epochs

Given below are the screenshots of the epochs running on the above-described architectures(epochs =7 )

### CNN Epochs

```
Epoch 1/7
500/500 [==============================] - 34s 56ms/step - loss: 0.1220 - accuracy: 0.9515 - val_loss: 0.0457 - val_accuracy:
0.9855
Epoch 2/7
500/500 [==============================] - 28s 55ms/step - loss: 0.0563 - accuracy: 0.9794 - val_loss: 0.0444 - val_accuracy:
0.9850
Epoch 3/7
500/500 [==============================] - 26s 53ms/step - loss: 0.0416 - accuracy: 0.9859 - val_loss: 0.0698 - val_accuracy:
0.9780
Epoch 4/7
500/500 [==============================] - 28s 56ms/step - loss: 0.0325 - accuracy: 0.9881 - val_loss: 0.0329 - val_accuracy:
0.9885
Epoch 5/7
500/500 [==============================] - 27s 54ms/step - loss: 0.0274 - accuracy: 0.9901 - val_loss: 0.0285 - val_accuracy:
0.9915
Epoch 6/7
500/500 [==============================] - 27s 54ms/step - loss: 0.0214 - accuracy: 0.9921 - val_loss: 0.0279 - val_accuracy:
0.9925
Epoch 7/7
500/500 [==============================] - 28s 55ms/step - loss: 0.0212 - accuracy: 0.9925 - val_loss: 0.0218 - val_accuracy:
0.9935
```

This is the CNN model that was implemented on the given dataset. I found a very little loss for both the training and validation set and get 99% accuracy for both the training and validation in this architecture.

### Resnet50 Epoch

```
: # Running the epochs on ResNet50 model
mymodel=resnet_50_model.fit(train_data,validation_data=valid_data,epochs=noepochs)
```

```
Epoch 1/7
500/500 [==============================] - 769s 2s/step - loss: 0.2227 - accuracy: 0.9249 - val_loss: 0.1624 - val_accuracy: 0.
9425
Epoch 2/7
500/500 [==============================] - 755s 2s/step - loss: 0.0712 - accuracy: 0.9771 - val_loss: 0.2213 - val_accuracy: 0.
8915
Epoch 3/7
500/500 [==============================] - 760s 2s/step - loss: 0.0633 - accuracy: 0.9801 - val_loss: 0.7896 - val_accuracy: 0.
7340
Epoch 4/7
500/500 [==============================] - 760s 2s/step - loss: 0.0374 - accuracy: 0.9871 - val_loss: 0.0608 - val_accuracy: 0.
9815
Epoch 5/7
500/500 [==============================] - 767s 2s/step - loss: 0.0444 - accuracy: 0.9854 - val_loss: 0.0400 - val_accuracy: 0.
9855
Epoch 6/7
500/500 [==============================] - 736s 1s/step - loss: 0.0379 - accuracy: 0.9871 - val_loss: 0.1014 - val_accuracy: 0.
9665
Epoch 7/7
500/500 [==============================] - 734s 1s/step - loss: 0.0344 - accuracy: 0.9875 - val_loss: 0.0950 - val_accuracy: 0.
9680
```

The Resnet 50 model's execution is attached here and we can see this architecture also performed good but not as good as compared to the previous one model

## VGG16 Epochs

```
]: # Running the epochs on VGG16 model
   mymodel = vgg_16_model.fit(train_data,validation_data=valid_data,epochs=noepochs)
```

```
Epoch 1/7
250/250 [==============================] - 1031s 4s/step - loss: 3.9861 - accuracy: 0.5013 - val_loss: 0.6935 - val_accuracy:
0.4950
Epoch 2/7
250/250 [==============================] - 957s 4s/step - loss: 0.6933 - accuracy: 0.5013 - val_loss: 0.6933 - val_accuracy: 0.
4950
Epoch 3/7
250/250 [==============================] - 946s 4s/step - loss: 0.6932 - accuracy: 0.5013 - val_loss: 0.6932 - val_accuracy: 0.
4950
Epoch 4/7
250/250 [==============================] - 973s 4s/step - loss: 0.6932 - accuracy: 0.5013 - val_loss: 0.6932 - val_accuracy: 0.
4950
Epoch 5/7
250/250 [==============================] - 961s 4s/step - loss: 0.6932 - accuracy: 0.5013 - val_loss: 0.6932 - val_accuracy: 0.
4950
Epoch 6/7
250/250 [==============================] - 939s 4s/step - loss: 0.6932 - accuracy: 0.5013 - val_loss: 0.6932 - val_accuracy: 0.
4950
Epoch 7/7
250/250 [==============================] - 941s 4s/step - loss: 0.6932 - accuracy: 0.5013 - val_loss: 0.6932 - val_accuracy: 0.
4950
```

Above is given the image of vgg16 describing the details of every epoch run. Vgg16 have a huge loss for both the training and validation set.This model performance is very bad as compared all three models.

## InceptionV3 Epochs

```
: # Running the epochs on inceptionV3 model
  mymodel = inceptionV3_model.fit(train_data,validation_data=valid_data,epochs=noepochs)
```

```
Epoch 1/7
250/250 [==============================] - 387s 1s/step - loss: 0.0785 - accuracy: 0.9749 - val_loss: 0.1164 - val_accuracy: 0.
9770
Epoch 2/7
250/250 [==============================] - 368s 1s/step - loss: 0.0213 - accuracy: 0.9936 - val_loss: 0.1524 - val_accuracy: 0.
9815
Epoch 3/7
250/250 [==============================] - 368s 1s/step - loss: 0.0231 - accuracy: 0.9945 - val_loss: 0.0242 - val_accuracy: 0.
9945
Epoch 4/7
250/250 [==============================] - 366s 1s/step - loss: 0.0446 - accuracy: 0.9879 - val_loss: 0.2469 - val_accuracy: 0.
9660
Epoch 5/7
250/250 [==============================] - 365s 1s/step - loss: 0.0450 - accuracy: 0.9910 - val_loss: 0.0348 - val_accuracy: 0.
9865
Epoch 6/7
250/250 [==============================] - 365s 1s/step - loss: 0.0375 - accuracy: 0.9900 - val_loss: 0.0401 - val_accuracy: 0.
9900
Epoch 7/7
250/250 [==============================] - 365s 1s/step - loss: 0.0156 - accuracy: 0.9961 - val_loss: 0.0782 - val_accuracy: 0.
9705
```

The last architecture that has given the 2nd best results is inceptionV3. Shown in a above image. Only CNN model perform better than is inceptionV3 architecture

## Results

The recorded results of the proposed approaches for this specific dataset are as follows:

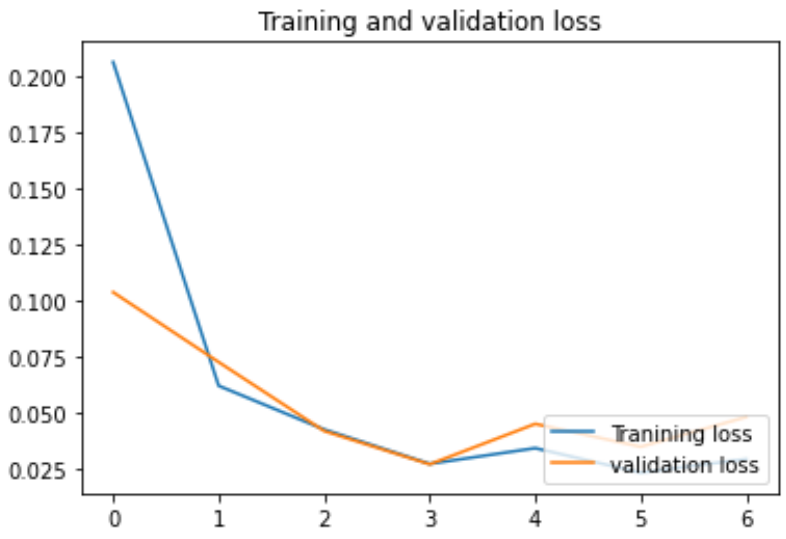
| Architecture | Accuracy |
|---|---|
| 1. CNN | 99% |
| 2. ResNet50 | 96% |
| 3. Vgg16 | 49% |
| 4. InceptionV3 | 97% |

### Plotting of the graphs

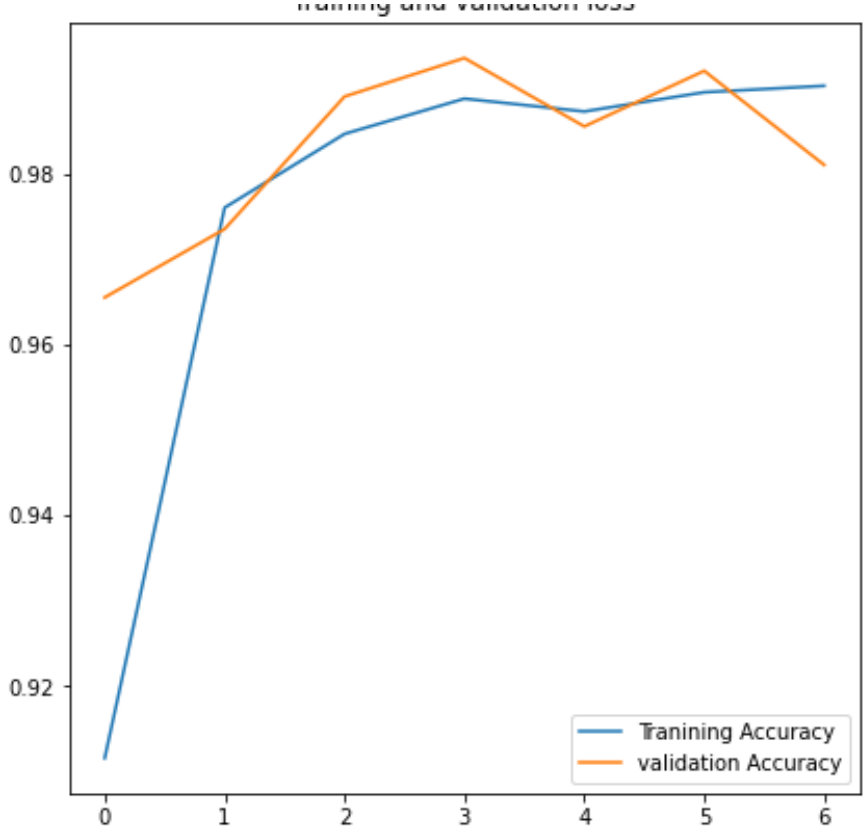
The results of each architecture applied are taken separately and represented by the plotting of the graphs and is shown below, in terms of 'accuracy' and 'loss'. CNN's ,VGG16's, ResNet50 and Inception loss and accuracy is show below

#### CNN accuracy and loss

```
]: Text(0.5, 1.0, 'Training and validation loss')
```

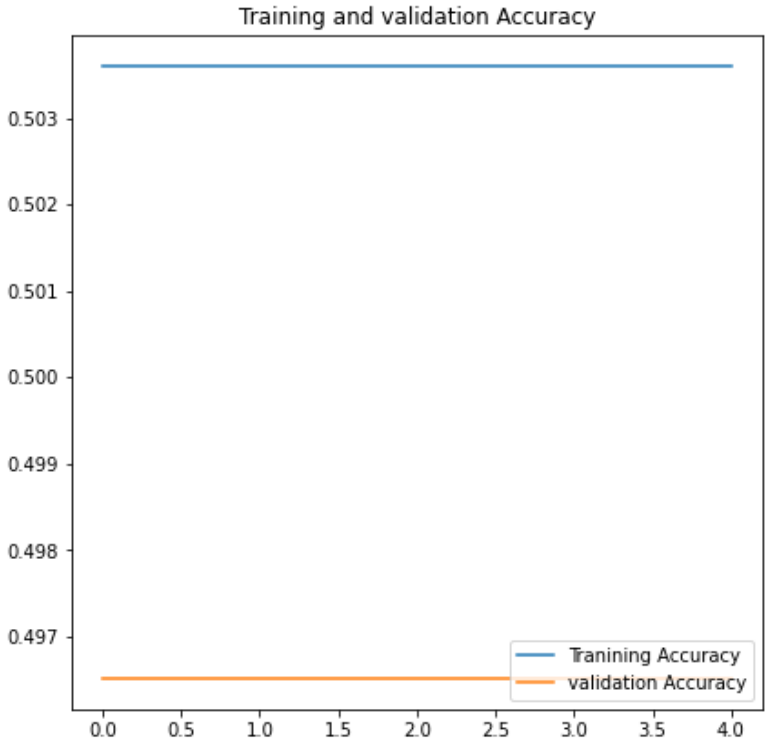


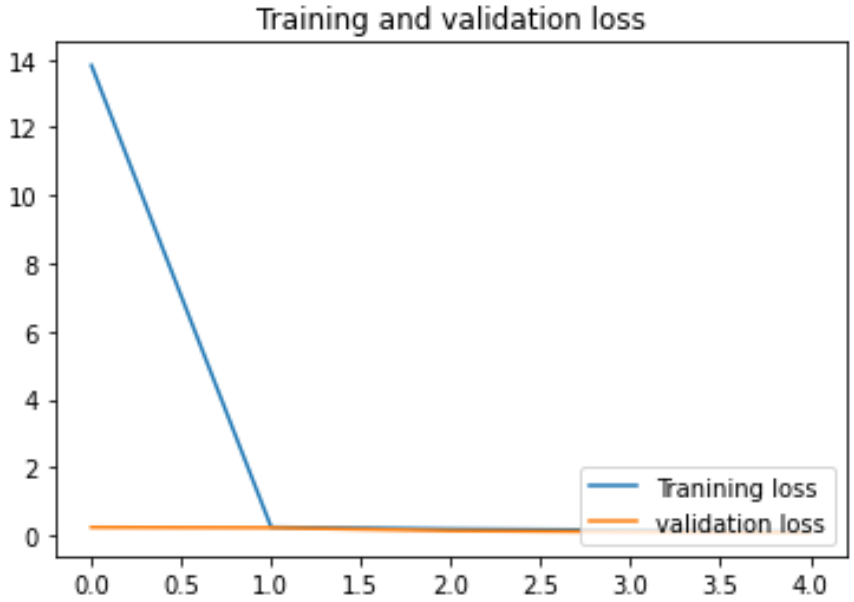

Training and validation loss



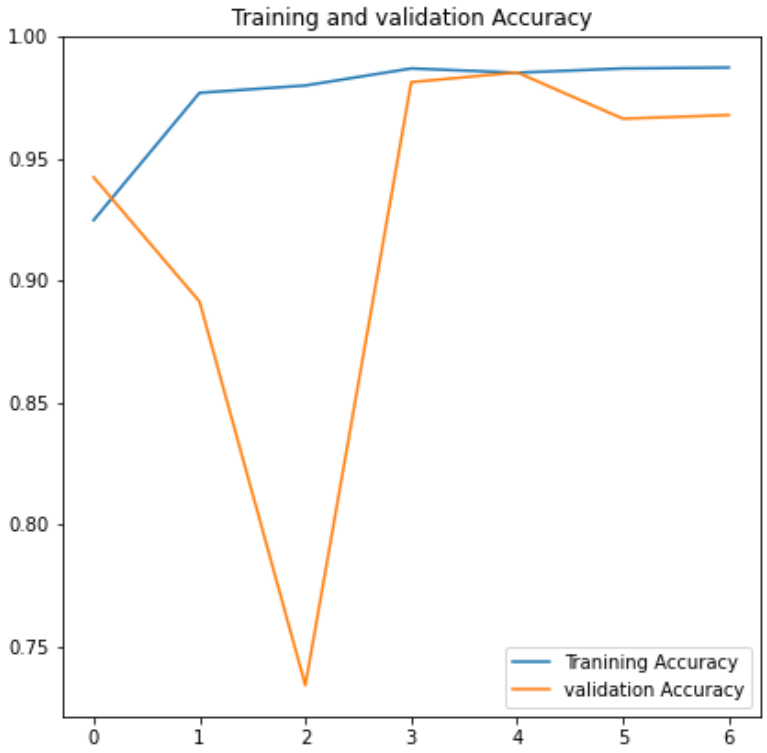

Training and validation Accuracy

## VGG16 accuracy and loss

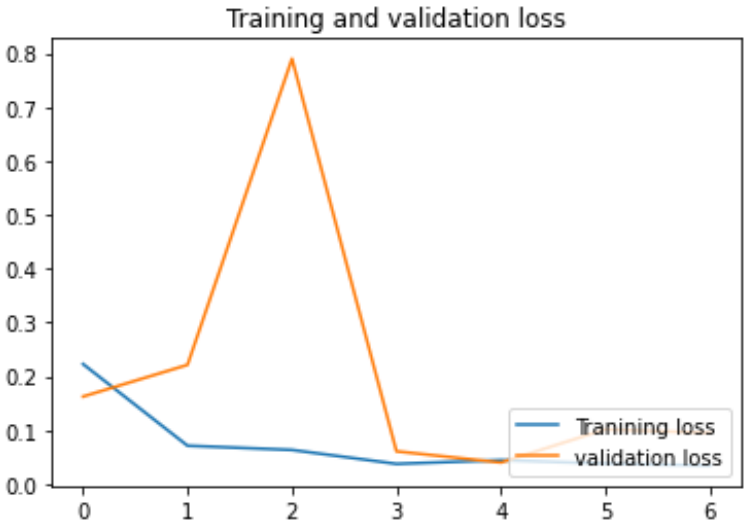




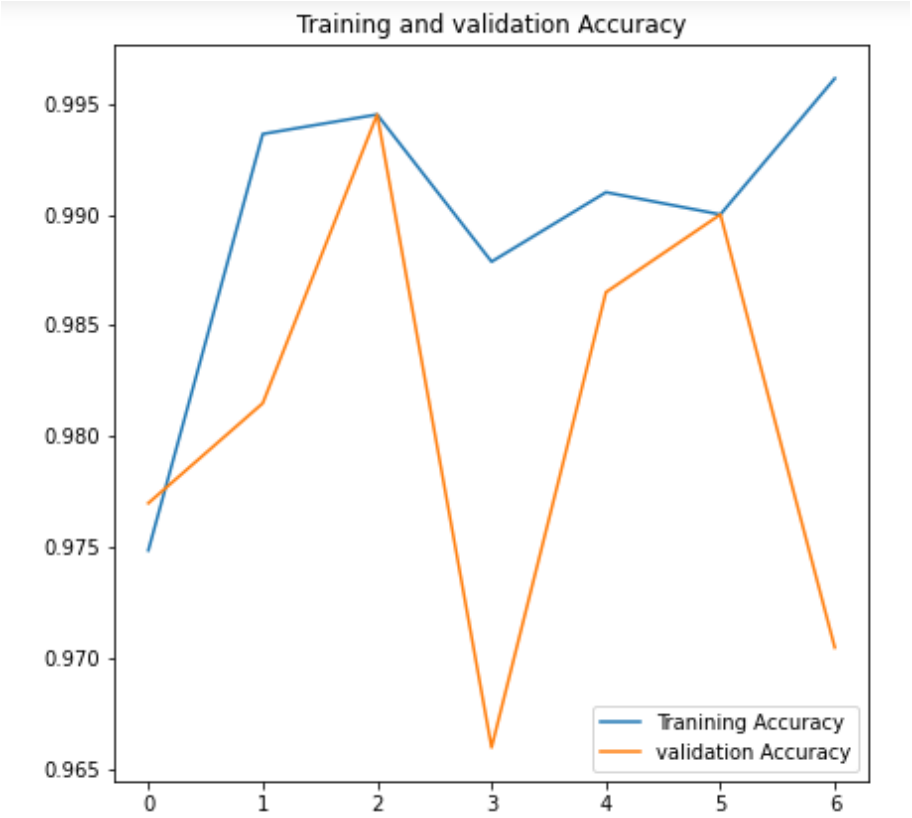

## ResNet50 accuracy and loss



```
Text(0.5, 1.0, 'Training and validation loss')
```




InceptionV3 accuracy and loss




## Code Link

I uploaded all the models on my GitHub profile.

- Project github link ( https://github.com/MoeezKokab/Face-Mask-Detection-using-Neural-Networks/tree/master)
- Cnn code github link ( https://github.com/MoeezKokab/Face-Mask-Detection-using-Neural-Networks/blob/master/CCN%20algo.ipynb)
- ResNet50 code github link (https://github.com/MoeezKokab/Face-Mask-Detection-using-Neural-Networks/blob/master/ResNet50.ipynb)
- VGG16 code github link ( https://github.com/MoeezKokab/Face-Mask-Detection-using-Neural-Networks/blob/master/VGG16.ipynb)
- InceptionV3 code github link (https://github.com/MoeezKokab/Face-Mask-Detection-using-Neural-Networks/blob/master/InceptionV3.ipynb)
- My profile link MoeezKokab ( https://github.com/MoeezKokab )

## Conclusion

Different results were archived after the processing of the proposed four architectures. Out of which, CNN showed the best results with an accuracy of 99%. Hence for this specific problem, we can apply CNN to apply for the 'Face Mask Detection'. In the future, we want to extend our work by expanding this dataset into three classes labeled: 'With Mask', 'Without Mask' and 'Improper Mask', and also we will be applying the latest Machine Learning approaches at that time

# References


1. Real-Time Face Mask Detection to Ensure COVID-19 Precautionary Measures in the Developing Countries
2. Prediction of COVID-19 active cases using exponential and non-linear growth models
3. Face Mask Detection System using CNN
4. Deep Neural Architecture for Face mask Detection on Simulated Masked Face Dataset against Covid-19 Pandemic
5. Face Mask Detection with alert system using Tensorflow, Keras and OpenCV
6. IRJET- A Survey: Different Techniques of Face Mask Detection
7. A FACEMASK DETECTOR USING MACHINE LEARNING AND IMAGE PROCESSING TECHNIQUES.
8. https://www.anaconda.com/
9. https://www.tensorflow.org/
10. https://www.tensorflow.org/api_docs/python/tf/keras
11. https://www.tensorflow.org/guide/keras/sequential_model
12. https://www.youtube.com/watch?v=WvoLTXIjBYU&t=42s
13. https://www.youtube.com/watch?v=-HwvjlHuUZg&t=1070s
14. https://www.kaggle.com/datasets/ashishjangra27/face-mask-12k-images-dataset
15. https://www.youtube.com/watch?v=wu5bATXIzGI
16. https://www.youtube.com/watch?v=JcU72smpLJk
17. https://www.youtube.com/watch?v=ybVV7iu8jis
18. https://www.youtube.com/watch?v=bIDnxmmj_so
19. https://github.com/MoeezKokab
20. https://github.com/MoeezKokab/Face-Mask-Detection-using-Neural-Networks/blob/master