**A PROJECT REPORT**

**OF**

**FACE MASK DETECTION**

**Submitted in partial fulfillment for the requirement of the completion of**

**SUMMER INTERNSHIP**

**IN**

**Artificial Intelligence and Machine Learning**



*Submitted by*

**RISHIKESH PATHAK**

(Karunya institute of technology and sciences, Tamil Nadu)

**Project: Real-time Face Mask detection**

**Business Problem:**

Most organizations and institutions are following stringent norms for prevention of the spread of Corona Virus such as social distancing, hygiene, etc. One important aspect is for all employees to be wearing face masks to contain this spread. Organizations are looking for applications that can detect if their employees or visitors are wearing face mask while entering their premises. They would like to use a network of cameras that can detect any breach of such instance and trigger a security alarm.

**Input provided for the project:**

Dummy dataset of pictures (about 500 images) with different faces and facemasks will be provided. Students can also download similar pictures from internet

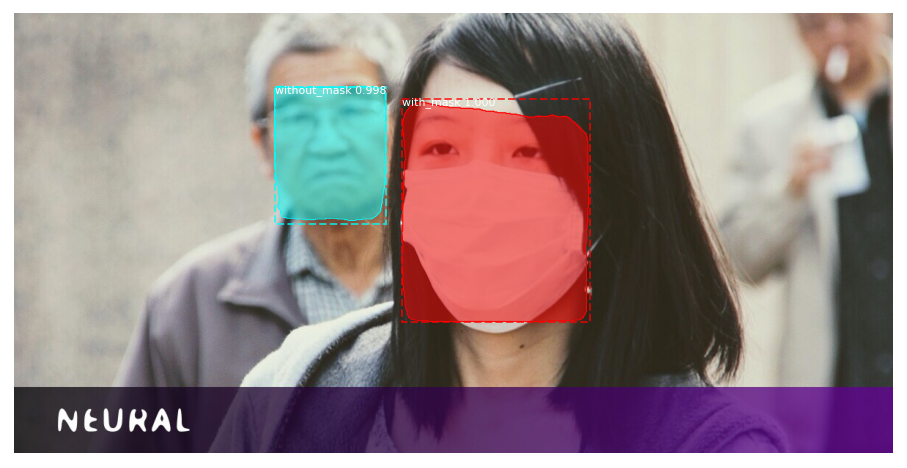
**DESCRIPTION:**

This project Face Mask Detection is done using Mask RCNN, Mask R-CNN is a sophisticated model to implement, especially as compared to a simple or even state-of-the-art deep convolutional neural network model. Instead of developing an implementation of the R-CNN or Mask R-CNN model from scratch, I have used a reliable third-party implementation built on top of the Keras deep learning framework. The third-party implementations of Mask R-CNN is the  [Mask R-CNN Project](https://github.com/matterport/Mask_RCNN) developed by [Matterport](https://matterport.com/). The project is open source released under a permissive license (e.g. MIT license) and the code has been widely used on a variety of projects and Kaggle competitions.

The data used was provided by Cognitel, consisting of 853 images of people with and without masks with their respective XML file as the images are annotated. As this was the data collection step moving on to the data preparation part, a dataset object must be made which then be used as an input for the model. Thus I extracted the information of annotations from the XML files and used it to make the dataset object.The model takes this dataset object and uses a pre-trained model (transfer learning) so that it takes less time to train for a better accuracy.

After the model is trained several times over every image is the dataset (epochs=10) with was 80% (20% for testing the model) of the whole dataset (642 images). In the testing phase the accuracy metric used is ‘Average Precision (AP)’ which turns out to be over 90% score, which means the model has trained well.

Using effective visualizations we can easily see the model detecting faces with and without masks.

****

**Mask RCNN :**



Mask RCNN is a deep neural network aimed to solve instance segmentation problem in machine learning or computer vision. In other words, it can separate different objects in a image or a video. You give it a image, it gives you the object bounding boxes, classes and masks.

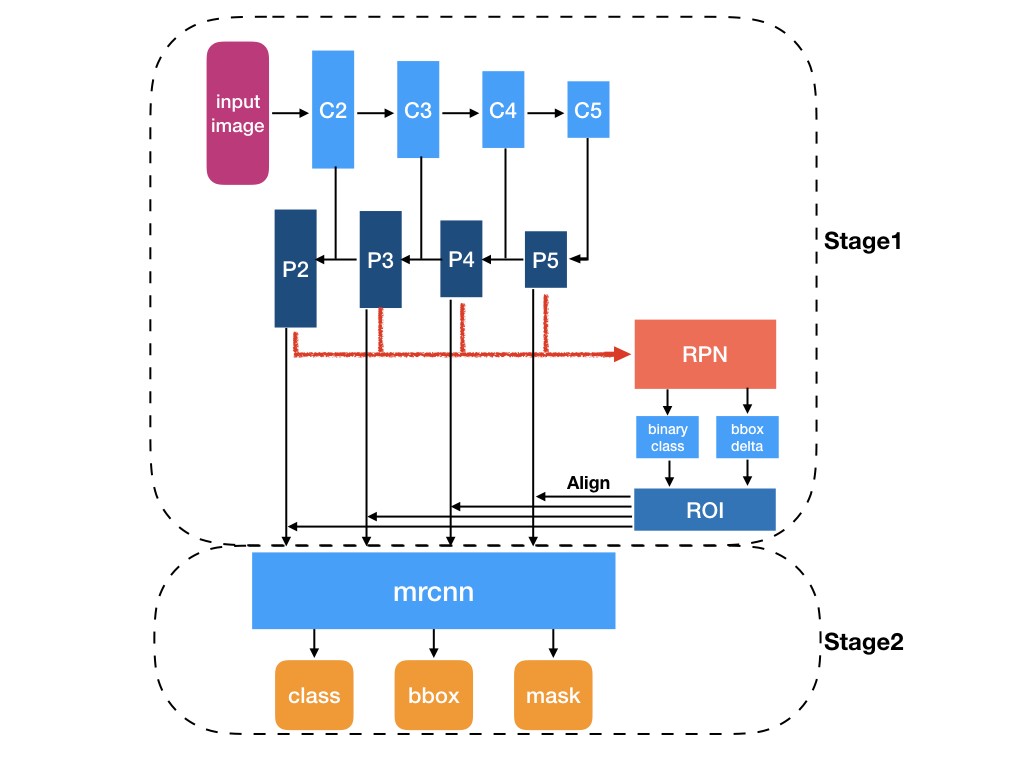
There are two stages of Mask RCNN. First, it generates proposals about the regions where there might be an object based on the input image. Second, it predicts the class of the object, refines the bounding box and generates a mask in pixel level of the object based on the first stage proposal. Both stages are connected to the backbone structure.

What is backbone? Backbone is a FPN style deep neural network. It consists of a bottom-up pathway , a top-bottom pathway and lateral connections. Bottom-up pathway can be any ConvNet, usually ResNet or VGG, which extracts features from raw images. Top-bottom pathway generates feature pyramid map which is similar in size to bottom-up pathway. Lateral connections are convolution and adding operations between two corresponding levels of the two pathways. FPN outperforms other single ConvNets mainly for the reason that it maintains strong semantically features at various resolution scales.

Now let’s look at the first stage. A light weight neural network called RPN scans all FPN top-bottom pathway( hereinafter referred to feature map) and proposes regions which may contain objects. That’s all it is. While scaning feature map is an efficient way, we need a method to bind features to its raw image location. Here come the anchors. Anchors are a set of boxes with predefined locations and scales relative to images.

Ground-truth classes( only object or background binary classified at this stage) and bounding boxes are assigned to individual anchors according to some IoU value. As anchors with different scales bind to different levels of feature map, RPN uses these anchors to figure out where of the feature map ‘should’ get an object and what size of its bounding box is. Here we may agree that convolving, downsampling and upsampling would keep features staying the same relative locations as the objects in original image, and wouldn’t mess them around.

At the second stage, another neural network takes proposed regions by the first stage and assign them to several specific areas of a feature map level, scans these areas, and generates objects classes(multi-categorical classified), bounding boxes and masks. The procedure looks similar to RPN. Differences are that without the help of anchors, stage-two used a trick called ROIAlign to locate the relevant areas of feature map, and there is a branch generating masks for each objects in pixel level.

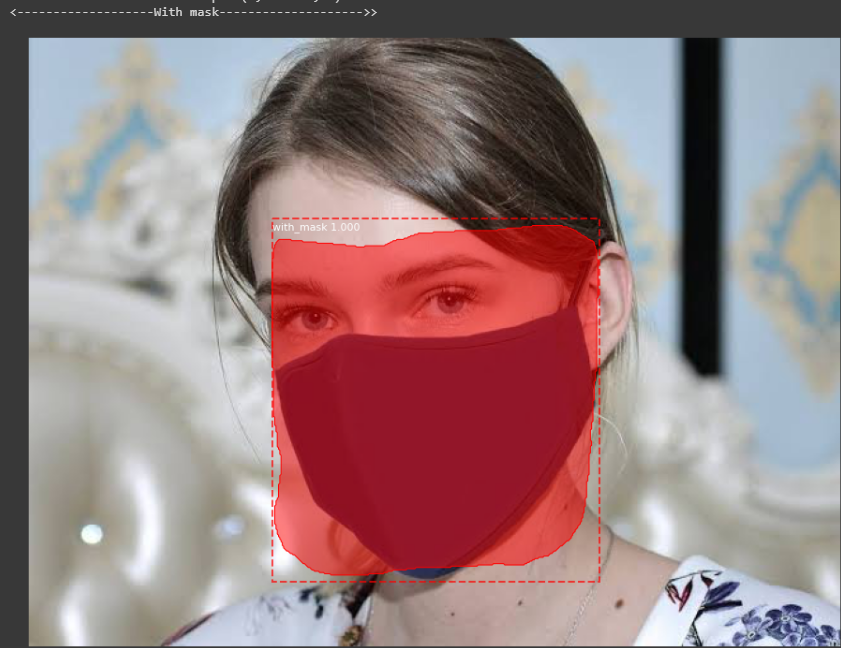
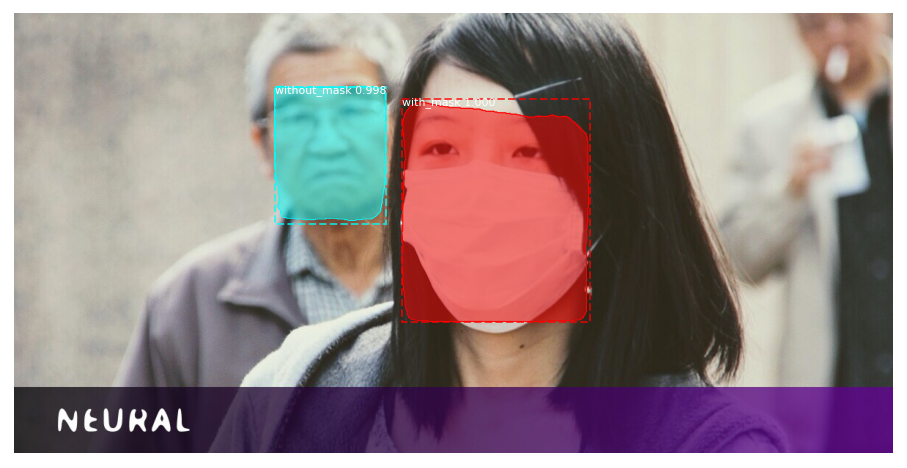


**Output:**

The aim for this project was to build an efficient face mask detector and accuracy metric depicts above 90% score which is pretty good for such small data set. I have made a video whose link will be below with some more sample image outputs from the model.

**Output Images –**



****

**Video link –**

https://drive.google.com/file/d/1p\_De\_SJ6oi2u10krl7PkXO\_5x8KD4nrZ/view?usp=sharing