MASK DETECTION AND RECOGNATION

Samet Sarıal – 64170037 – Computer Engineering

Emre Erkan – 64170039 – Computer Engineering

Abstract

Nowadays, we should be careful about health in our daily lives due to the pandemic. For this, people should be more careful about themselves and their environment by wearing a mask. Unfortunately, there are people who put themselves and their environment in danger by not wearing a mask. For this, officers make the necessary warnings by walking around the streets, avenues and crowded areas and impose penal sanctions on those who do not obey. In our imaginary scenario, our technique can be used in any area of contemporary life in the healthcare field: people's faces can be scanned with cameras placed on the streets or in the streets. The scanned faces will be scanned for masks and the officers assigned to certain regions for them will be able to see them from the cameras. Thus, the number of people assigned to the regions will be reduced and a more risk-free work will be done. With this project, it will be possible to have a better control in hospitals, streets, streets, in short, in crowded areas, and it will be ensured that this pandemic period is overcome with more care and with as little damage as possible.

Our method basically has two steps: First one is creating the mask detection modal and applying to the face, second one is detecting faces from the given input. Detected faces will given to the mask modal, modal will provide the with mask and without mask information. End of the process, real values and predicted values will be displayed to the user.

Introduction

The first case of covid-19 was seen in 2020 and now it threatens our lives with 144 million active cases and 3 million deaths. In order to prevent the increase of these cases, we will be able to check mask people with basic technology items. According to research, when we, the source of infection, wear the face mask, we minimize the number of droplets emitted by 99 percent. It also reduces the number of droplets consumed by an uninfected person, though not as effectively. It is really necessary for wearing masks by people in public places and should really be made mandatory rather than being based on individual decisions, as a significant portion of people with infection lack coronavirus symptoms.

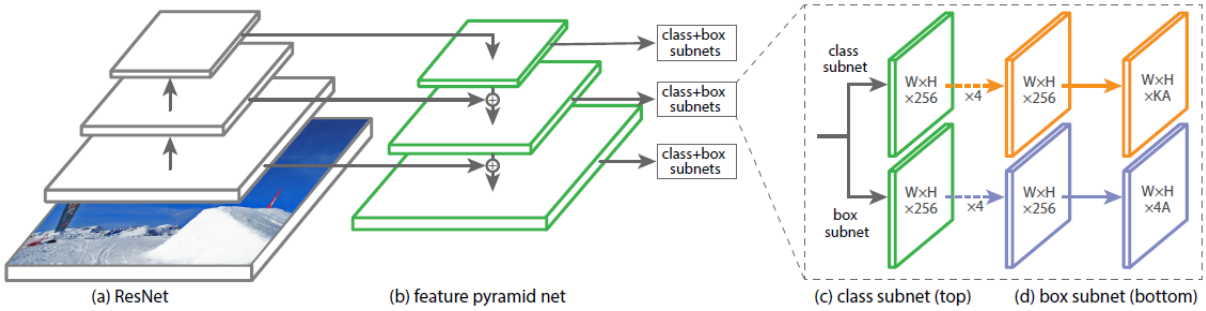
```
image_path = '/content/image6.jpg'

for i in glob(image_path):
  image = cv2.cvtColor(cv2.imread(i), cv2.COLOR_BGR2RGB)

  result = RetinaFace.detect_faces(image)

  boxes = []
  for i in result:
    boxes.append(result[i]['facial_area'])
  boxes = np.array(boxes)

  visualize_detections(image, boxes)
```

We want to contribute to end the coronavirus worldwide by producing this project at low cost and with the best results in order to minimize the cases of coronavirus.

The modal we are working on is designed to give the most accurate and best results whether people are wearing mask in crowded environments. The results that our modal gave us and the outputs we got proved the accuracy of the modal. Results obtained on the classification modal:
Loss: 0.0182 - F1 score: 0.99

Related Works

Our modal has 2 different part one of them is recognation to the face then recognated face goes to the mask detection modal and then, its returning to the result to the user. The modal working fine in single person or crowded areas. For other literature works, our modal approach similar to the literature different ways. For example, we took face recognation modal from the RetinaNet.



Featured image pyramids have been used in computer vision to detect items in images with varied scales. Featurized image pyramids means one would take an image and subsample it into lower resolution and smaller size image. RetinaNet is one of the best one-stage object detection modals that has proven to work well with dense and small-scale objects. For this reason, it has become a popular object detection and also face recognation.

For face-mask detection we used Xception modal with a modified final fully connected layer. Our dataset for classification has limited amount of data (around 16K) available for training the classifier, I was inclined to use transfer learning for the purpose of our task of classifying an input image into the categories of whether the subject is wearing a mask or not.

The difference between the other literature works with our work is, other ones work with the small datasets because of the avaliable images and some many of them works with the artificially created datasets. So, they are taking high accuracy but in real examples, modal predict different results from the real results. So, our modal is working fine in real life according to the crowded and single person face-mask detection.

Data

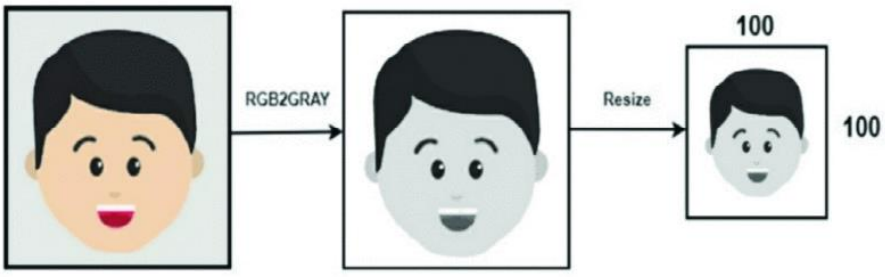
A mask detection modal is a hybrid of a classification and a face detection modal. To provide a practical test scenario for our mask detection modal, the photographs and video sequences obtained for the purpose of testing our modal had to come from a crowded environment. Our dataset is divided into two categories: images of people wearing masks and images of people without masks.

This dataset was created by scraping the web for practical examples suitable for inference in a real-world observation setting using Google image search and accessible YouTube videos. It has been collected for use on a variety of masks and different types of human faces. We will develop a modal to get the most accurate information in crowded areas.The total number of images in the training set were divided into the two categories as follows:

```
print('TOTAL IMAGE NUMBER = '+str(len(images)))
print('TRAIN IMAGES  = '+str(len(train_images)))
print('TOTAL IMAGES = '+str(len(test_images)))
```

```
TOTAL IMAGE NUMBER = 16159
TRAIN IMAGES  = 14543
TOTAL IMAGES = 1616
```

Our dataset was created with real images from around the world over the internet, so our images were not ready for classification and human faces were not specified. The face detection modal was used for data processing and classification, and then faces were classified into masked and unmasked categories. Our datasets contain many different faces with mask or witout mask so this variety becomes a problem because different faces and different camera angles gives bigger or smaller photos so we applied resizing and categorization for mask or no-mask to all photos. After categorization we divide into the train and test dataset and modals determined if photo is wearing mask photo will become a green, if its not it will be a red.

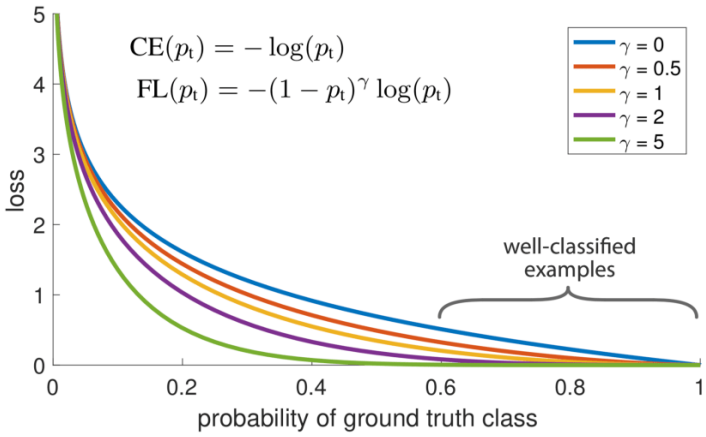


Method

Our problem is coronavirus and wearing masks have big role against the coronavirus. So, our modal purpose is helping people for getting rid of from the virus. For doing that we used face detection modal and developed mask detection modal.

We used RetinaNet face detection modal. RetinaNet uses focal loss which focuses training on hard negatives, while down-weighting easy examples, using parameters alpha and gamma which provide the hyper-parameters for focusing on hard negatives and offset class imbalance of the number of examples. As a backbone, the ResNet architecture is used to detect features from input images and a feature pyramid network is used as a fast and multiscale feature extractor on top of

this backbone to create a multi-scale pyramid of features. Focal Loss (FL) is an improved version of Cross-Entropy Loss (CE) that tries to handle the class imbalance problem by assigning more weights to hard or easily misclassified examples (i.e. background with noisy texture or partial object or the object of our interest) and to down-weight easy examples.(ResNet is modal of the RetinaNet)



VGGNet – 1st Runner Up in ILSVRC 2014

ResNet – Winner in ILSVRC 2015

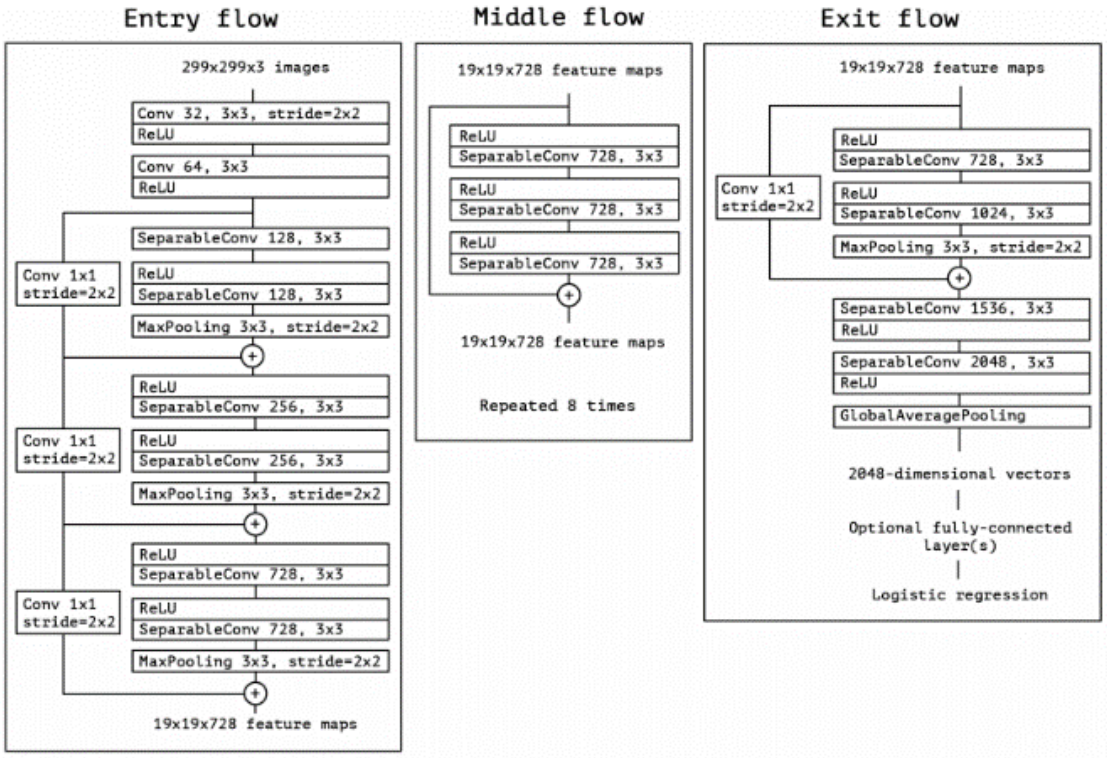
Inception-v3 – 1st Runner Up in ILSVRC 2015

	Top-1 accuracy	Top-5 accuracy
VGG-16	0.715	0.901
ResNet-152	0.770	0.933
Inception V3	0.782	0.941
Xception	**0.790**	**0.945**

For the mask detection and classification modal is Xception. It is called an "extreme version of its predecessor Inception V3" for the reason that an efficient use of modal parameters by using depth-wise separable convolutions gives better performance on the ImageNet datasets. XCeption is an efficient architecture that relies on two main points, Depthwise Separable Convolution and Shortcuts between Convolution blocks as in ResNet. Xception modals remain expensive to train but are pretty good improvements compared to Inception. Transfer learning brings part of the solution when it comes to adapting such algorithms to your specific task.

Implementation details for classification architecture:

- Batch Size: 32
- Epochs: 2
- Learning rate: 1e-4 (with decay of 1e-4 / epoch)
- Gradient Descent Optimizer: Adam
- Loss function: Sparse categorical cross entropy
- Criterion for evaluation: F1-score

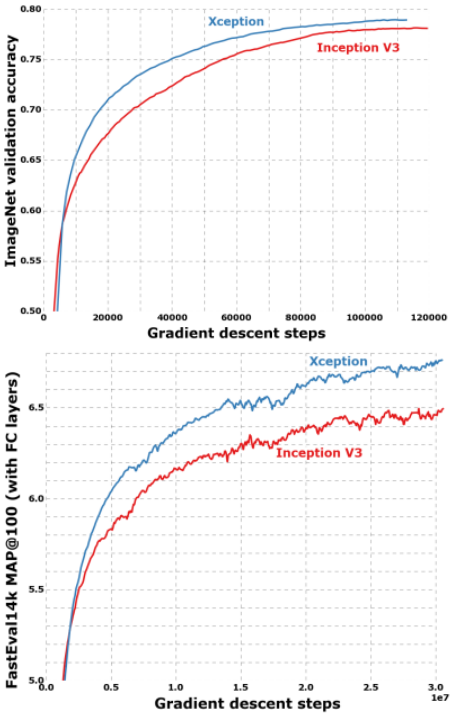


Experiment

As we talked previous parts our project contains 2 main parts one of them xception modal which we used for determine and train the mask detection modal and the other one is for face-detection. We used for real testing samples which we used our images. We talked about the Xception modal in the modal part and its contains 3 main parts as you can see on the figure. Also as you can see on the figure Xception modal contains SparableConv layer so why seperableConv is better than traditional one ?

If we were to use a normal convolution on the input tensor, and we use a filter/kernel size of 3x3x3 and the total number of filters we want is 64. So a total is 3x3x3x64.In separable convolution, we first use 3x3x1x3 in depthwise convolution and 1x1x3x64 in pointwise convolution. Traditional Convolutional layer = 3x3x3x64 = 1,728. Separable Convolutional layer = (3x3x1x3)+(1x1x3x64) = 27+192 = 219.

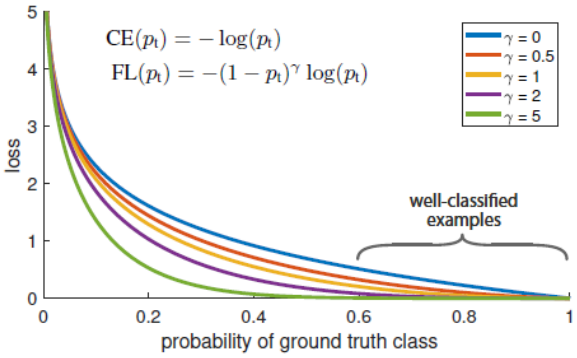
As you can see, separable convolution layers are way more advantageous than traditional convolutional layers, both in terms of computation cost as well as memory. The main difference is that in the normal convolution, we are transforming the image multiple times. And every transformation uses up 1,728 multiplications. In the separable convolution, we only transform the image once. Then, we take the transformed image and simply elongate it to 64 channels without having to transform the image over and over again, we can save up on computational

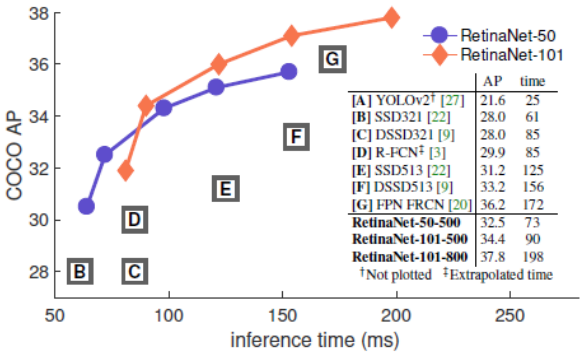


Xception performance vs Inception on JFT and ImageNet

all details about the Xception modal from the link all architecture, Regularization configuration, Training infrastructure and so on all other informations avaliable on the local documentation. For improve the accuracy different methods can be try but we took f1-scores for criterion for evaluation. F1-score is the harmonic mean of precision and recall. It is chosen as the criterion for evaluation for the classification modal. F1 score: 0.99 so we got the best result using the parameters we mentioned.

So why we chose RetinaNet? While deciding upon the face detection algorithm to be used as part of my proposed solution of face detection, a pre-trained RetinaNet was chosen as the one which could, with the highest recall and precision, predict the number of faces in a crowded setting. RetinaNet has been formed by making two improvements over existing single stage object detection modals. Feature Pyramid Networks for Object Detection, Focal Loss for Dense Object Detection. Feature pyramid was shown on the previous figure. Focal Loss is an improvement on cross-entropy loss that helps to reduce the relative loss for well-classified examples and putting more focus on hard, misclassified examples.





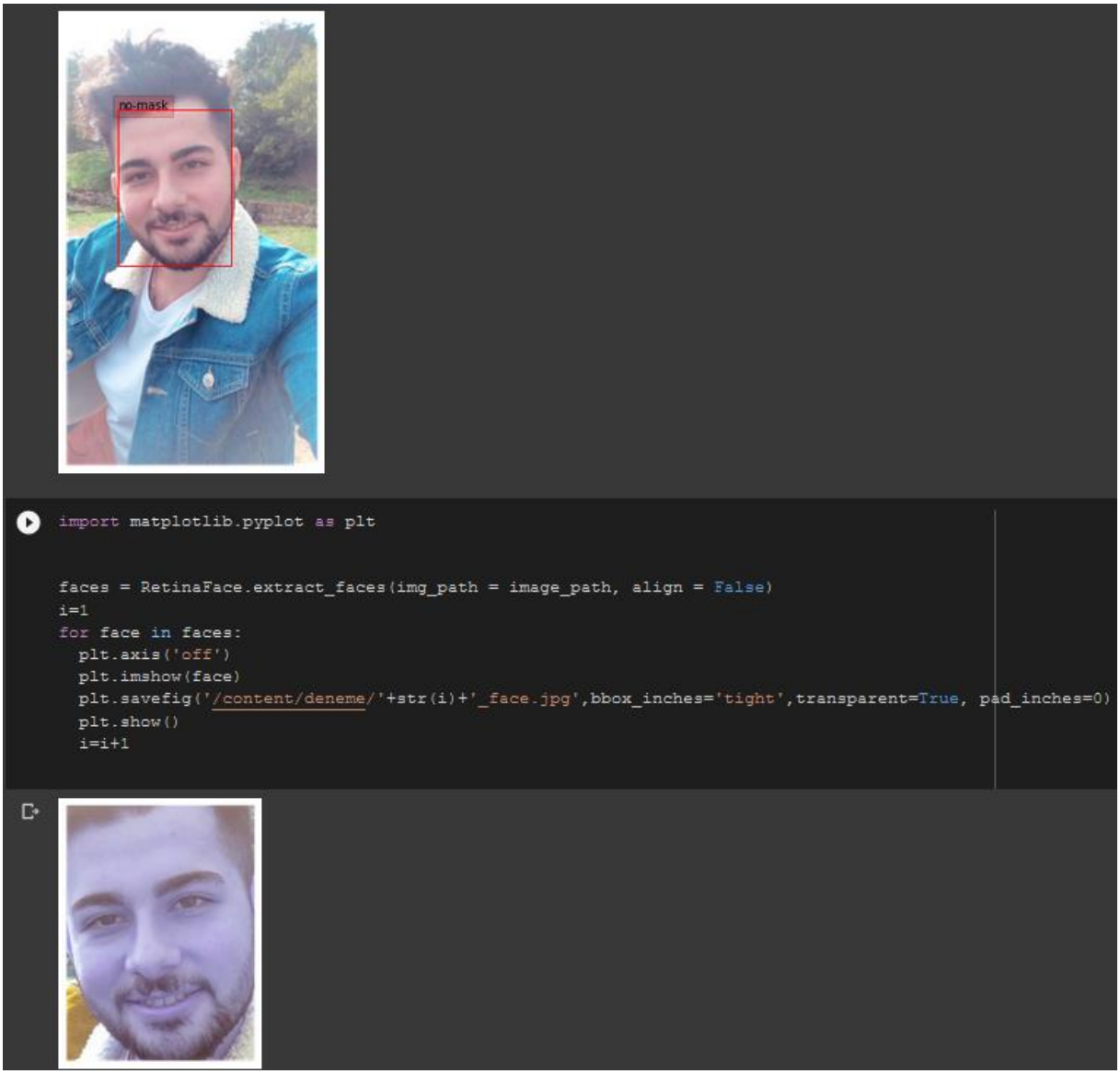
Conclusion

In the previous parts of our article, information about the learning and performance of the modal was given. The techniques and method used provided a high rate of accuracy, but of course a number of problems were encountered. Chief among these was that the modal had difficulty trying to distinguish whether the people in the ensemble images used were wearing masks. The blurring in the picture, the distant and small faces of the people reduced the accuracy. But when the pictures were clear, the accuracy was quite satisfactory. We thought improvements could be made to Face Deteciton to fix this.

We know that the CoronaVirus crisis has made it mandatory for all people to wear masks. Now, every time we go out in public, we need to have a mask on our faces, and public institutions display an extremely sensitive attitude on these issues. Our modal will make a great contribution to the public health system in this direction. Improvements that can be made on the modal in this direction may be to enable the person to perceive whether he or she is wearing the mask in the right position. For this, we can work on the face coordinates to search for the mask in certain areas, or

to check whether the mask used is an effective mask against the virus, that is, whether it is a surgical mask. These features can be studied and developed for the modal.

Supplementary Material:



The face algorithm working perfectly on inputs we developed this code for selection of faces and saving into the specific folder so you can find all faces.

```
print(train_dataset.take(1))
plt.figure(figsize=(15,15))
for i in range(8):
  for val in train_dataset.take(1):
    plt.subplot(4,2,i+1)
    img  = (val[0][i]+1)*127.5
    plt.imshow(tf.cast(img,tf.uint8))
    y_pred = model.predict(np.expand_dims(val[0][i],axis=0))
    y_pred = np.argmax(y_pred,axis=1)
    plt.title('Truth {}, Predicted {}'.format(val[1][i],y_pred))
    plt.subplots_adjust(wspace=1, hspace=1)
plt.show()
```

```
<TakeDataset shapes: ((None, 128, 128, 3), (None,)), types: (tf.float32, tf.int32
```

The mask detection modal whic we developed -training example its gives predicted data 0 means with mask 1 means no-mask you can see its perfectly returning right results.

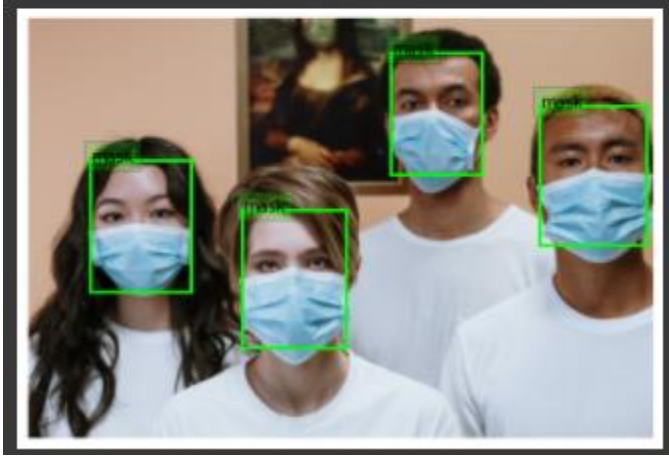
```
pred_labels = []
for image_path in (test_images):
  image = read_img(image_path)
  y_pred = model.predict(np.expand_dims(image,axis=0))
  y_pred = np.argmax(y_pred,axis=1)
  pred_labels.append(y_pred)

print(classification_report(test_labels, pred_labels))
```

```
              precision    recall  f1-score   support

           0       1.00      0.99      1.00       807
           1       0.99      1.00      1.00       809

    accuracy                           1.00      1616
   macro avg       1.00      1.00      1.00      1616
weighted avg       1.00      1.00      1.00      1616
```

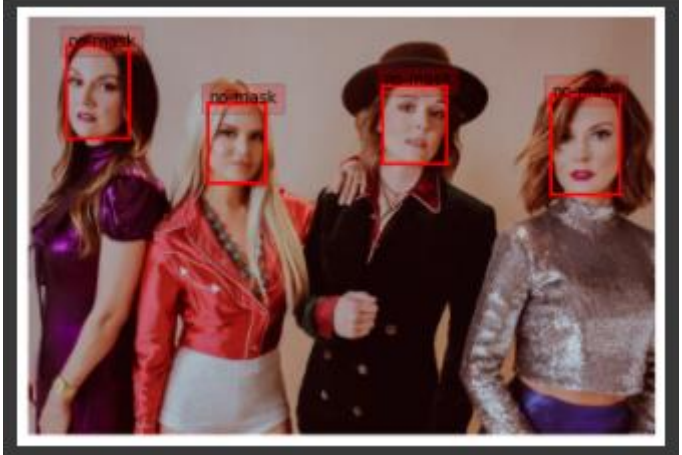
```
block13_sepconv2 (SeparableConv (None, None, None, 1 752024      block13_sepconv2_act[0][0]
__________________________________________________________________________________________________
block13_sepconv2_bn (BatchNorma (None, None, None, 1 4096        block13_sepconv2[0][0]
__________________________________________________________________________________________________
conv2d_3 (Conv2D)               (None, None, None, 1 745472      add_82[0][0]
__________________________________________________________________________________________________
block13_pool (MaxPooling2D)     (None, None, None, 1 0           block13_sepconv2_bn[0][0]
__________________________________________________________________________________________________
batch_normalization_3 (BatchNor (None, None, None, 1 4096        conv2d_3[0][0]
__________________________________________________________________________________________________
add_83 (Add)                    (None, None, None, 1 0           block13_pool[0][0]
                                                                 batch_normalization_3[0][0]
__________________________________________________________________________________________________
block14_sepconv1 (SeparableConv (None, None, None, 1 1582080     add_83[0][0]
__________________________________________________________________________________________________
block14_sepconv1_bn (BatchNorma (None, None, None, 1 6144        block14_sepconv1[0][0]
__________________________________________________________________________________________________
block14_sepconv1_act (Activatio (None, None, None, 1 0           block14_sepconv1_bn[0][0]
__________________________________________________________________________________________________
block14_sepconv2 (SeparableConv (None, None, None, 2 3159552     block14_sepconv1_act[0][0]
__________________________________________________________________________________________________
block14_sepconv2_bn (BatchNorma (None, None, None, 2 8192        block14_sepconv2[0][0]
__________________________________________________________________________________________________
block14_sepconv2_act (Activatio (None, None, None, 2 0           block14_sepconv2_bn[0][0]
__________________________________________________________________________________________________
global_average_pooling2d (Globa (None, 2048)         0           block14_sepconv2_act[0][0]
__________________________________________________________________________________________________
dense (Dense)                   (None, 1024)         2098176     global_average_pooling2d[0][0]
__________________________________________________________________________________________________
dropout (Dropout)               (None, 1024)         0           dense[0][0]
__________________________________________________________________________________________________
dense_1 (Dense)                 (None, 2)            2050        dropout[0][0]
==================================================================================================
Total params: 22,961,706
Trainable params: 2,100,226
Non-trainable params: 20,861,480
__________________________________________________________________________________________________
```

End of the modal.summary code its returning this for more check ipyb notebook file

```
image_path = '/content/image8.jpg'

for i in glob(image_path):
  image = cv2.cvtColor(cv2.imread(i), cv2.COLOR_BGR2RGB)

  result = RetinaFace.detect_faces(image)

  boxes = []
  for i in result:
    boxes.append(result[i]['facial_area'])
  boxes = np.array(boxes)

  visualize_detections(image, boxes)
```



```
image_path = '/content/image7.jpg'

for i in glob(image_path):
  image = cv2.cvtColor(cv2.imread(i), cv2.COLOR_BGR2R

  result = RetinaFace.detect_faces(image)

  boxes = []
  for i in result:
    boxes.append(result[i]['facial_area'])
  boxes = np.array(boxes)

  visualize_detections(image, boxes)
```




```
image_path = '/content/image4.jpg'

for i in glob(image_path):
  image = cv2.cvtColor(cv2.imread(i), cv2.COLOR_BGR2R

  result = RetinaFace.detect_faces(image)

  boxes = []
  for i in result:
    boxes.append(result[i]['facial_area'])
  boxes = np.array(boxes)

  visualize_detections(image, boxes)
```

Evaluation The Project:

The hardest points were working with the colab because retinaNet modal files needed to be installing manually to the root/.cache file so until find the correct direction we suffered too much but all things put together perfectly Xception modal and our datasets worked fine. But resizing datasets and preparing data was hard one. This project was fun and instructive. The perfect results followed this process.

References:

[1] https://arxiv.org/abs/1610.02357v3

https://www.kaggle.com/andrewmvd/face-mask-detection

https://towardsdatascience.com/object-detection-on-aerial-imagery-using-retinanet-626130ba2203

https://github.com/smtsarial/mask-detection/tree/main/Dataset

https://github.com/fizyr/keras-retinanet

https://keras.io/examples/vision/retinanet/

https://www.pyimagesearch.com/2020/05/04/covid-19-face-mask-detector-with-opencv-keras-tensorflow-and-deep-learning/

https://www.kaggle.com/alpertemel/mask-detection

https://www.kaggle.com/delllectron/face-mask-detection-transfer-learning-xception

https://www.kaggle.com/chaitanya99/face-mask-detection-xception-haar-cascade

https://github.com/ksvbka/face-mask-detector