```
!pip install kaggle
```

```
Requirement already satisfied: kaggle in /usr/local/lib/python3.10/dist-packages (1.6.17)
Requirement already satisfied: six>=1.10 in /usr/local/lib/python3.10/dist-packages (from kaggle) (1.16.0)
Requirement already satisfied: certifi>=2023.7.22 in /usr/local/lib/python3.10/dist-packages (from kaggle) (2024.8.30)
Requirement already satisfied: python-dateutil in /usr/local/lib/python3.10/dist-packages (from kaggle) (2.9.0.post0)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-packages (from kaggle) (2.32.3)
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages (from kaggle) (4.66.5)
Requirement already satisfied: python-slugify in /usr/local/lib/python3.10/dist-packages (from kaggle) (8.0.4)
Requirement already satisfied: urllib3 in /usr/local/lib/python3.10/dist-packages (from kaggle) (2.2.3)
Requirement already satisfied: bleach in /usr/local/lib/python3.10/dist-packages (from kaggle) (6.1.0)
Requirement already satisfied: webencodings in /usr/local/lib/python3.10/dist-packages (from bleach->kaggle) (0.5.1)
Requirement already satisfied: text-unidecode>=1.3 in /usr/local/lib/python3.10/dist-packages (from python-slugify->kaggle) (1.3)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests->kaggle) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests->kaggle) (3.10)
```

```
!mkdir -p ~/.kaggle
!cp kaggle.json ~/.kaggle/
!chmod 600 ~/.kaggle/kaggle.json
```

```
!kaggle datasets download -d omkargurav/face-mask-dataset
```

```
Dataset URL: https://www.kaggle.com/datasets/omkargurav/face-mask-dataset
License(s): unknown
Downloading face-mask-dataset.zip to /content
 93% 151M/163M [00:00<00:00, 258MB/s]
100% 163M/163M [00:00<00:00, 237MB/s]
```

```
from zipfile import ZipFile
dataset = '/content/face-mask-dataset.zip'

with ZipFile(dataset,'r') as zip:
  zip.extractall()
  print('The Dataset is Extracted')
```

```
The Dataset is Extracted
```

Importing the Libraries

```
import os # to access the file in folder
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
import cv2
```

```
from google.colab.patches import cv2_imshow
from PIL import Image
from sklearn.model_selection import train_test_split
```

```
with_mask_files = os.listdir('/content/data/with_mask')
print(with_mask_files[0:5])
print(with_mask_files[-5:])
```

```
['with_mask_2030.jpg', 'with_mask_1493.jpg', 'with_mask_417.jpg', 'with_mask_2099.jpg', 'with_mask_3204.jpg']
['with_mask_2619.jpg', 'with_mask_874.jpg', 'with_mask_3041.jpg', 'with_mask_2484.jpg', 'with_mask_2241.jpg']
```

```
without_mask_files = os.listdir('/content/data/without_mask')
print(without_mask_files[0:5])
print(without_mask_files[-5:])
```

```
['without_mask_1858.jpg', 'without_mask_1204.jpg', 'without_mask_1174.jpg', 'without_mask_3785.jpg', 'without_mask_3520.jpg']
['without_mask_1307.jpg', 'without_mask_1166.jpg', 'without_mask_1902.jpg', 'without_mask_1828.jpg', 'without_mask_675.jpg']
```

```
print('Number of with mask images:', len(with_mask_files))
print('Number of without mask images:', len(without_mask_files))
```

```
Number of with mask images: 3725
Number of without mask images: 3828
```

Creating Labels for the two class of Images

with mask --> 1

without mask --> 0

```
with_mask_labels = [1]*3725
without_mask_labels = [0]*3828
```

```
print(with_mask_labels[0:5])
print(without_mask_labels[-5:])
```

```
[1, 1, 1, 1, 1]
[0, 0, 0, 0, 0]
```

```
print(len(with_mask_labels))
print(len(without_mask_labels))
```

```
3725
3828
```

```
labels = with_mask_labels + without_mask_labels
print(len(labels))
```

```
7553
```

Displaying the Images

```
img = mpimg.imread('/content/data/with_mask/with_mask_2619.jpg')
imgplot = plt.imshow(img)
plt.show()
```

```
img = mpimg.imread('/content/data/without_mask/without_mask_1307.jpg')
imgplot = plt.imshow(img)
plt.show()
```

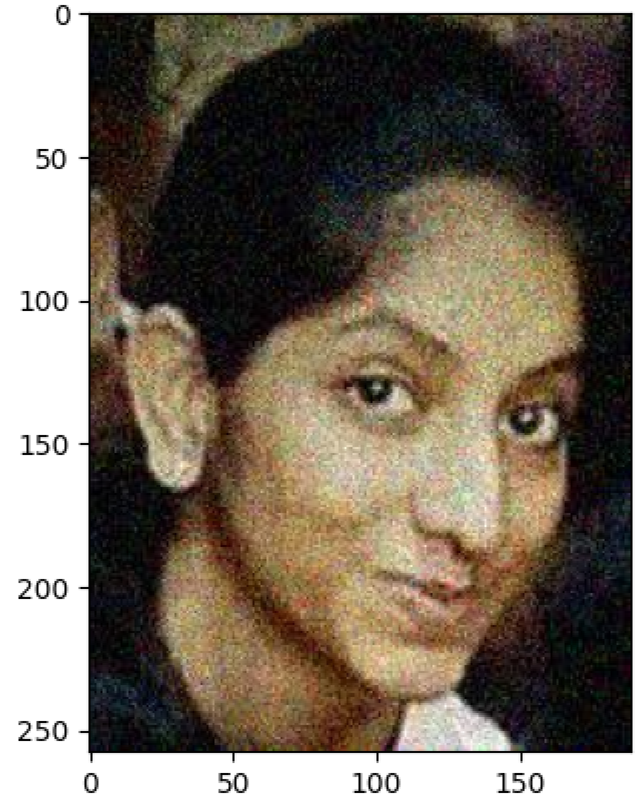


Image Processing

```
with_mask_path = '/content/data/with_mask/'
data = []
for img_file in with_mask_files:
  image = Image.open(with_mask_path + img_file)
  image = image.resize((128,128))
  image = image.convert('RGB')
  image = np.array(image)
  data.append(image)



without_mask_path = '/content/data/without_mask/'
for img_file in without_mask_files:
  image = Image.open(without_mask_path + img_file)
  image = image.resize((128,128))
  image = image.convert('RGB')
  image = np.array(image)
  data.append(image)
```

```
/usr/local/lib/python3.10/dist-packages/PIL/Image.py:1056: UserWarning: Palette images with Transparency expressed in bytes should be converted to RGBA images
  warnings.warn(
type(data)
```

```
list
```

```
len(data)
```

```
7553
```

```
data[0]
```

```
ndarray (128, 128, 3) show data
```

```
data[0].shape
```

```
(128, 128, 3)
```

```
X = np.array(data)
y = np.array(labels)
```

```
type(X)
```

```
numpy.ndarray
```

```
print(X.shape)
print(y.shape)
```

```
(7553, 128, 128, 3)
(7553,)
```

Train Test Split

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=2)
```

```
print(X.shape, X_train.shape, X_test.shape )
```

```
(7553, 128, 128, 3) (6042, 128, 128, 3) (1511, 128, 128, 3)
```

```
X_train_scaled = X_train/255
X_test_scaled = X_test/255
```

```
X_train_scaled[0]
```

```
array([[[0.33333333, 0.19215686, 0.12941176],
        [0.33333333, 0.19215686, 0.12941176],
        [0.33333333, 0.19215686, 0.12941176],
        ...,
        [0.79607843, 0.64705882, 0.59607843],
        [0.93333333, 0.78431373, 0.73333333],
        [0.98823529, 0.83921569, 0.78823529]],

       [[0.3372549 , 0.19607843, 0.13333333],
        [0.3372549 , 0.19607843, 0.13333333],
        [0.3372549 , 0.19607843, 0.13333333],
        ...,
        [0.74901961, 0.6       , 0.54901961],
        [0.8745098 , 0.7254902 , 0.6745098 ],
        [0.9254902 , 0.77647059, 0.7254902 ]],

       [[0.35294118, 0.21176471, 0.14901961],
        [0.34901961, 0.20784314, 0.14901961],
        [0.34509804, 0.20784314, 0.14509804],
        ...,
        [0.63137255, 0.47843137, 0.42745098],
        [0.73333333, 0.58431373, 0.52941176],
        [0.77254902, 0.61960784, 0.56862745]],

       ...,

       [[0.10196078, 0.07058824, 0.02745098],
        [0.10196078, 0.07058824, 0.02745098],
        [0.10196078, 0.0745098 , 0.02745098],
        ...,
        [0.11372549, 0.07843137, 0.05882353],
        [0.11372549, 0.08235294, 0.05882353],
        [0.10980392, 0.07843137, 0.05882353]],

       [[0.11764706, 0.08627451, 0.04313725],
        [0.11372549, 0.08235294, 0.03921569],
        [0.10980392, 0.07843137, 0.03529412],
        ...,
        [0.11764706, 0.08235294, 0.0627451 ],
        [0.12156863, 0.08627451, 0.06666667],
        [0.12156863, 0.08627451, 0.06666667]],

       [[0.12156863, 0.09019608, 0.04705882],
        [0.11764706, 0.08627451, 0.04313725],
        [0.10980392, 0.07843137, 0.03529412],
        ...,
        [0.11764706, 0.08235294, 0.0627451 ],
        [0.1254902 , 0.08627451, 0.06666667],
        [0.1254902 , 0.08627451, 0.06666667]]]])
```

Building a Convolution Neural Network

```
import tensorflow as tf
from tensorflow import keras
```

```
num_of_classes =2
model = keras.Sequential()
model.add(keras.layers.Conv2D(32, kernel_size = (3,3), activation= 'relu', input_shape = (128, 128, 3)))
model.add(keras.layers.MaxPooling2D(pool_size = (2,2)))


model.add(keras.layers.Conv2D(64, kernel_size = (3,3), activation= 'relu'))
model.add(keras.layers.MaxPooling2D(pool_size = (2,2)))


model.add(keras.layers.Flatten())
model.add(keras.layers.Dense(128, activation = 'relu'))
model.add(keras.layers.Dropout(0.5))


model.add(keras.layers.Dense(64, activation = 'relu'))
model.add(keras.layers.Dropout(0.5))


model.add(keras.layers.Dense(num_of_classes, activation = 'sigmoid'))
```

```
model.compile(optimizer = 'adam',
              loss = 'sparse_categorical_crossentropy',
              metrics =  ['acc'])
```

```
history = model.fit(X_train_scaled, y_train, validation_split = 0.1, epochs = 5)
```

```
Epoch 1/5
170/170 [==============================] - 18s 97ms/step - loss: 0.4642 - acc: 0.8001 - val_loss: 0.3381 - val_acc: 0.8463
Epoch 2/5
170/170 [==============================] - 16s 95ms/step - loss: 0.2959 - acc: 0.8784 - val_loss: 0.2621 - val_acc: 0.8826
```

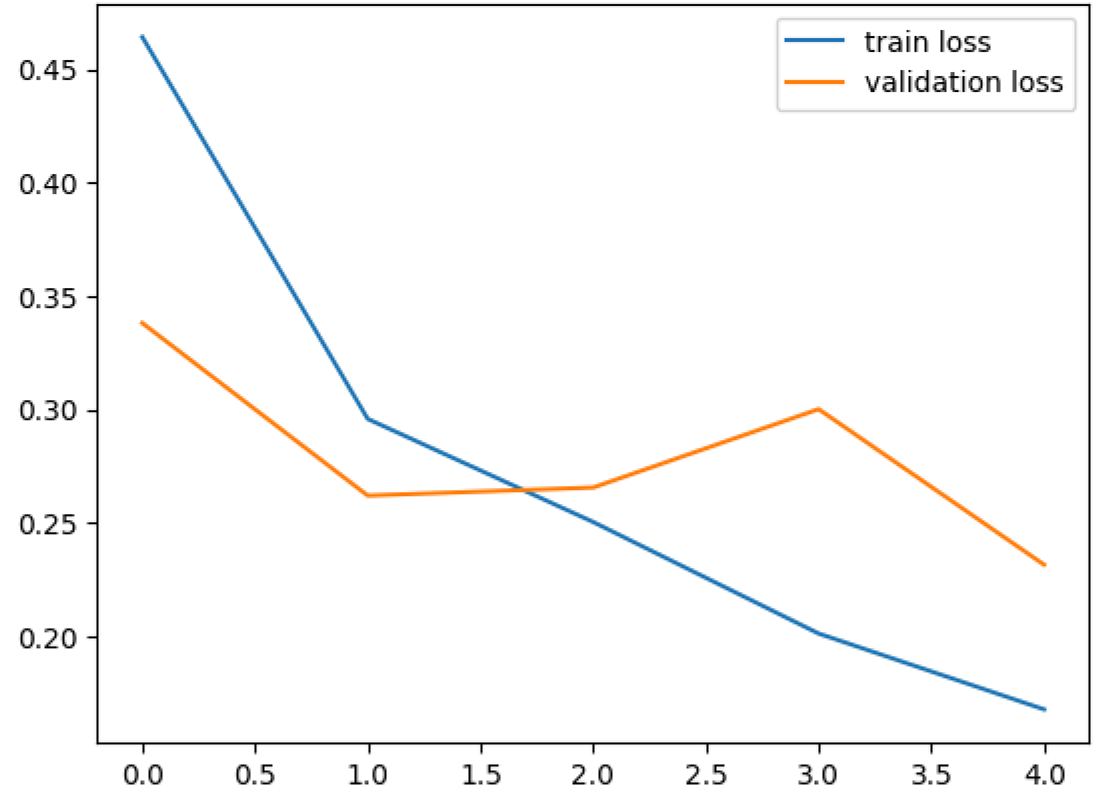
```
Epoch 3/5
170/170 [==============================] - 16s 95ms/step - loss: 0.2505 - acc: 0.8983 - val_loss: 0.2656 - val_acc: 0.8843
Epoch 4/5
170/170 [==============================] - 16s 96ms/step - loss: 0.2013 - acc: 0.9191 - val_loss: 0.3002 - val_acc: 0.8744
Epoch 5/5
170/170 [==============================] - 16s 96ms/step - loss: 0.1679 - acc: 0.9321 - val_loss: 0.2316 - val_acc: 0.9074
```

Model Evaluation

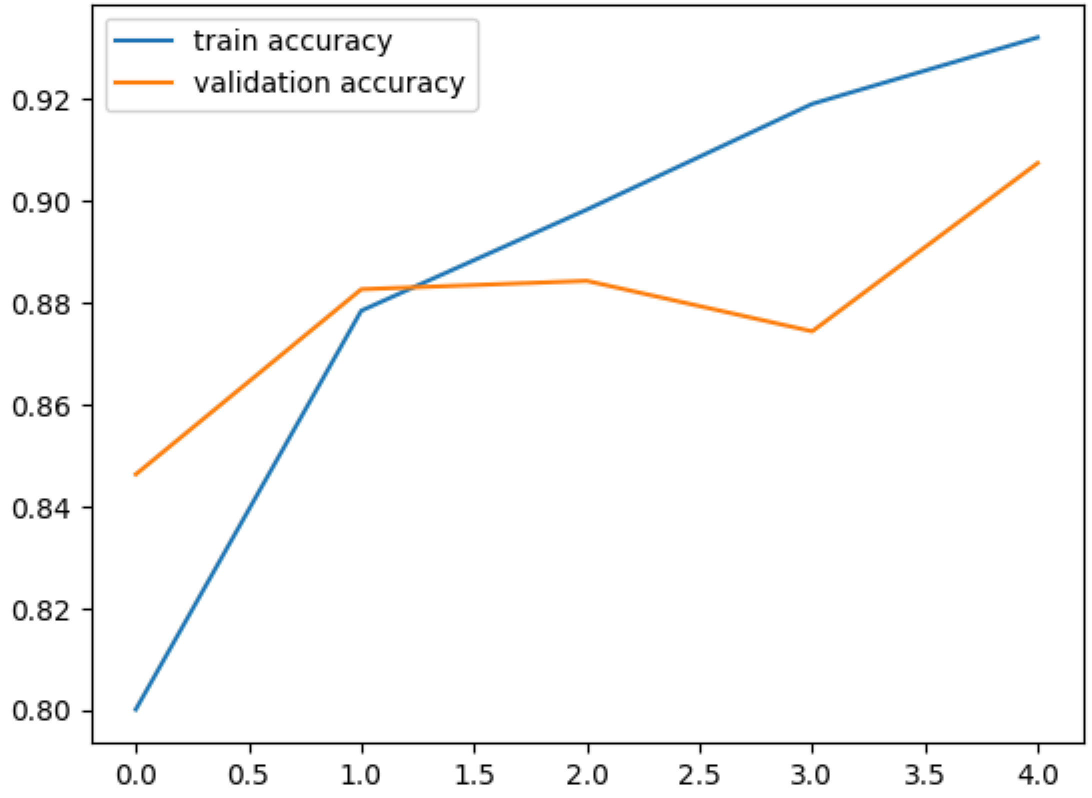
```
loss, accuracy = model.evaluate(X_test_scaled, y_test)
print('Test Accuravy =', accuracy)
```

```
48/48 [==============================] - 1s 25ms/step - loss: 0.1918 - acc: 0.9265
Test Accuravy = 0.9265387058258057
```

```
h= history
plt.plot(h.history['loss'], label = 'train loss')
plt.plot(h.history['val_loss'], label = 'validation loss')
plt.legend()
plt.show()
```



```
plt.plot(h.history['acc'], label = 'train accuracy')
plt.plot(h.history['val_acc'], label = 'validation accuracy')
plt.legend()
plt.show()
```



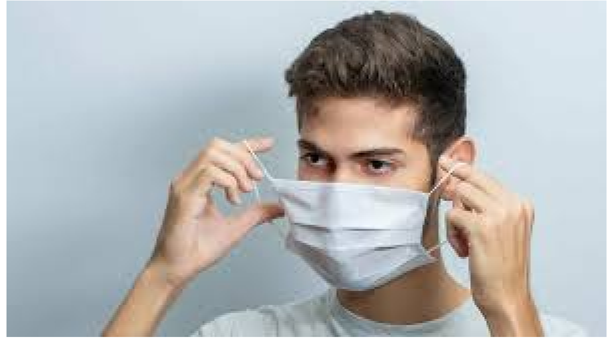
Predictive System

```
input_image_path = input('Path of the image to be predictede: ')
input_image = cv2.imread(input_image_path)
cv2_imshow(input_image)
input_image_resized = cv2.resize(input_image, (128,128))
input_image_scaled = input_image_resized/255
input_image_reshaped = np.reshape(input_image_scaled, [1,128,128,3])
input_prediction = model.predict(input_image_reshaped)
print(input_prediction)
input_pred_label = np.argmax(input_prediction)
print(input_pred_label)

if input_pred_label == 1:
 print('The person in the image is wearing mask')

else:
  print('The person in the image is not wearing mask')
```

```
Path of the image to be predictede: /content/mask wearing_1.jpg
```



```
1/1 [==============================] - 0s 32ms/step
[[0.14906374 0.7396202 ]]
1
The person in the image is wearing mask
```