In [2]:

```
!unzip '/content/drive/My Drive/Covid19Pred/Dataset_kaggle.zip'
```

```
  inflating: Dataset_kaggle/COVID/Covid (1032).png
  inflating: Dataset_kaggle/COVID/Covid (1033).png
  inflating: Dataset_kaggle/COVID/Covid (1034).png
  inflating: Dataset_kaggle/COVID/Covid (1035).png
  inflating: Dataset_kaggle/COVID/Covid (1036).png
  inflating: Dataset_kaggle/COVID/Covid (1037).png
  inflating: Dataset_kaggle/COVID/Covid (1038).png
  inflating: Dataset_kaggle/COVID/Covid (1039).png
  inflating: Dataset_kaggle/COVID/Covid (104).png
  inflating: Dataset_kaggle/COVID/Covid (1040).png
  inflating: Dataset_kaggle/COVID/Covid (1041).png
  inflating: Dataset_kaggle/COVID/Covid (1042).png

  inflating: Dataset_kaggle/COVID/Covid (1043).png
  inflating: Dataset_kaggle/COVID/Covid (1044).png
  inflating: Dataset_kaggle/COVID/Covid (1045).png
  inflating: Dataset_kaggle/COVID/Covid (1046).png
  inflating: Dataset_kaggle/COVID/Covid (1047).png
  inflating: Dataset_kaggle/COVID/Covid (1048).png
  inflating: Dataset_kaggle/COVID/Covid (1049).png
```

In [5]:

```
!pip install split_folders
```

```
Collecting split_folders
  Downloading https://files.pythonhosted.org/packages/20/67/29dda743e6d23ac1ea3
d16704d8bbb48d65faf3f1b1eaf53153b3da56c56/split_folders-0.3.1-py3-none-any.whl
 (https://files.pythonhosted.org/packages/20/67/29dda743e6d23ac1ea3d16704d8bbb4
8d65faf3f1b1eaf53153b3da56c56/split_folders-0.3.1-py3-none-any.whl)
Installing collected packages: split-folders
Successfully installed split-folders-0.3.1
```

In [7]:

```
import split_folders
split_folders.ratio('/content/Dataset_kaggle', output="output", seed=1337, ratio
```

```
Copying files: 0 files [00:00, ? files/s]
Copying files: 415 files [00:00, 4145.79 files/s]
Copying files: 929 files [00:00, 4400.15 files/s]
Copying files: 1230 files [00:00, 3591.63 files/s]
Copying files: 1527 files [00:00, 3377.15 files/s]
Copying files: 1821 files [00:00, 3230.08 files/s]
Copying files: 2094 files [00:00, 3059.96 files/s]
Copying files: 2481 files [00:00, 3328.57 files/s]
```

In [11]:

```
import pandas as pd
import numpy as np
import os
import tensorflow as tf
import keras
import matplotlib.pyplot as plt
from tensorflow.keras.layers import Dense, GlobalAveragePooling2D
from tensorflow.keras.applications.vgg16 import VGG16
from tensorflow.keras.preprocessing import image
from tensorflow.keras.applications.vgg16 import preprocess_input
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.models import Model
from tensorflow.keras.optimizers import Adam

image_size = [224,224]
data_path = '/content/output'

vgg = VGG16(input_shape= image_size+[3],weights='imagenet',include_top=False)

x = vgg.output
x = GlobalAveragePooling2D()(x)

x = Dense(1024,activation='relu')(x)
x = Dense(1024,activation='relu')(x)
x = Dense(512, activation='relu')(x)

preds = Dense(2,activation='softmax')(x)

model = Model(inputs = vgg.input,outputs=preds)

for layer in vgg.layers:
    layer.trainable = False

train_datagen=ImageDataGenerator(preprocessing_function=preprocess_input) #inclu
test_datagen=ImageDataGenerator(preprocessing_function=preprocess_input)
train_generator=train_datagen.flow_from_directory('/content/output/train', # thi
                                                 target_size=(224,224),
                                                 color_mode='rgb',
                                                 batch_size=32,
                                                 class_mode='categorical',
                                                 shuffle=True)
test_generator=test_datagen.flow_from_directory('/content/output/val', # this is
                                                 target_size=(224,224),
                                                 color_mode='rgb',
                                                 batch_size=32,
                                                 shuffle=False)


model.compile(optimizer='Adam',
              loss='categorical_crossentropy',
              metrics=['accuracy'])



print(train_generator.n)
print(train_generator.batch_size)
print(746//32)
```

```
step_size_train=train_generator.n//train_generator.batch_size
r = model.fit_generator(generator=train_generator,
                    validation_data=test_generator,
                   steps_per_epoch=step_size_train,
                   epochs=20)
```

```
Found 1984 images belonging to 2 classes.
Found 497 images belonging to 2 classes.
1984
32
23
Epoch 1/20
62/62 [==============================] - 16s 255ms/step - loss: 1.0543 - accu
racy: 0.7455 - val_loss: 0.2845 - val_accuracy: 0.8732
Epoch 2/20
62/62 [==============================] - 16s 255ms/step - loss: 0.2334 - accu
racy: 0.9022 - val_loss: 0.5626 - val_accuracy: 0.8109
Epoch 3/20
62/62 [==============================] - 16s 255ms/step - loss: 0.2009 - accu
racy: 0.9118 - val_loss: 0.1496 - val_accuracy: 0.9356
Epoch 4/20
62/62 [==============================] - 16s 252ms/step - loss: 0.0977 - accu
racy: 0.9688 - val_loss: 0.1129 - val_accuracy: 0.9557
Epoch 5/20
62/62 [==============================] - 16s 252ms/step - loss: 0.0780 - accu
racy: 0.9713 - val_loss: 0.0945 - val_accuracy: 0.9638
Epoch 6/20
62/62 [==============================] - 15s 248ms/step - loss: 0.0616 - accu
racy: 0.9753 - val_loss: 0.0962 - val_accuracy: 0.9577
Epoch 7/20
62/62 [==============================] - 15s 248ms/step - loss: 0.0405 - accu
racy: 0.9854 - val_loss: 0.1152 - val_accuracy: 0.9537
Epoch 8/20
62/62 [==============================] - 15s 246ms/step - loss: 0.0402 - accu
racy: 0.9864 - val_loss: 0.1018 - val_accuracy: 0.9638
Epoch 9/20
62/62 [==============================] - 15s 246ms/step - loss: 0.0309 - accu
racy: 0.9879 - val_loss: 0.1348 - val_accuracy: 0.9577
Epoch 10/20
62/62 [==============================] - 15s 246ms/step - loss: 0.0390 - accu
racy: 0.9854 - val_loss: 0.2001 - val_accuracy: 0.9457
Epoch 11/20
62/62 [==============================] - 15s 243ms/step - loss: 0.0303 - accu
racy: 0.9889 - val_loss: 0.1115 - val_accuracy: 0.9598
Epoch 12/20
62/62 [==============================] - 15s 244ms/step - loss: 0.0312 - accu
racy: 0.9889 - val_loss: 0.0814 - val_accuracy: 0.9638
Epoch 13/20
62/62 [==============================] - 15s 245ms/step - loss: 0.0129 - accu
racy: 0.9955 - val_loss: 0.1104 - val_accuracy: 0.9738
Epoch 14/20
62/62 [==============================] - 15s 247ms/step - loss: 0.1173 - accu
racy: 0.9567 - val_loss: 0.2338 - val_accuracy: 0.9356
Epoch 15/20
62/62 [==============================] - 15s 244ms/step - loss: 0.0418 - accu
racy: 0.9889 - val_loss: 0.3270 - val_accuracy: 0.9276
```

```
Epoch 16/20
62/62 [==============================] - 15s 242ms/step - loss: 0.0174 - accu
racy: 0.9934 - val_loss: 0.1800 - val_accuracy: 0.9537
Epoch 17/20
62/62 [==============================] - 15s 245ms/step - loss: 0.0468 - accu
racy: 0.9839 - val_loss: 0.1127 - val_accuracy: 0.9557
Epoch 18/20
62/62 [==============================] - 15s 244ms/step - loss: 0.0090 - accu
racy: 0.9965 - val_loss: 0.0986 - val_accuracy: 0.9759
Epoch 19/20
62/62 [==============================] - 15s 244ms/step - loss: 0.0068 - accu
racy: 0.9965 - val_loss: 0.0986 - val_accuracy: 0.9658
Epoch 20/20
62/62 [==============================] - 15s 243ms/step - loss: 0.0097 - accu
racy: 0.9970 - val_loss: 0.0960 - val_accuracy: 0.9718
```

In [12]:

```
acc=model.evaluate_generator(test_generator)
print(acc[1])
```

```
0.9718309640884399
```

In [ ]: