Corona Lung Dataset-CNN

In [24]:

```python
import numpy as np
import cv2
import os

import warnings
warnings.filterwarnings("ignore")

from concurrent.futures import ThreadPoolExecutor
```

In [25]:

```python
images=[os.path.join(os.getcwd(),"train",image)
       for image in os.listdir("./train/")
       if image.endswith("jpg")
       or image.endswith(".png")
       or image.endswith(".jpeg")]

print(images[10])

covid=[]
normal=[]
for image in images:
    if image.split("\\")[-1].startswith("covid"):
        covid.append(image)
    else:
        normal.append(image)

print(f"covid:- {covid[10]}")
print(f"covid:-{normal[10]}")
```

```
C:\Users\User\train\covid 108.jpg
covid:- C:\Users\User\train\covid 108.jpg
covid:-C:\Users\User\train\normal 108.jpeg
```

In [13]:

```python
len(covid)
```

Out[13]:

```
151
```

In [26]:

```python
%time
print("100")
```

```
Wall time: 0 ns
100
```

In [27]:

```python
# covid
import time
start=time.time()
def covid_save(img,label):
    path=os.path.join("Data_", "train", label, img.split("\\")[-1])
    img=cv2.imread(img)
    cv2.imwrite(f"{path}", img)


for cov in covid:
    covid_save(cov, "cov")

print(time.time()-start)
```

```
10.249018669128418
```

In [28]:

```python
# normal
start=time.time()

def normal_save(img):
    path=os.path.join("Data_", "train", "normal", img.split("\\")[-1])
    img=cv2.imread(img)
    cv2.imwrite(f"{path}", img)

with ThreadPoolExecutor(max_workers=100) as executor:
    executor.map(normal_save,normal)


print(time.time()-start)
```

```
7.142252445220947
```

In [29]:

```python
import keras
from keras.layers import Dense, Activation, Conv2D, MaxPool2D, Dropout
from keras.models import Sequential
from keras.optimizers import RMSprop, SGD, Adam
from keras.preprocessing import image
from keras.layers import BatchNormalization, Flatten
from keras.layers import ZeroPadding2D

#DataGenerator
train_gen=image.ImageDataGenerator(rescale=1./255,
                                   featurewise_center=True,
                                   samplewise_center=True,
                                   featurewise_std_normalization=True,
                                   samplewise_std_normalization=True,
                                   zca_whitening=False, zca_epsilon=1e-06,
                                   rotation_range=40, width_shift_range=0.0,
                                   height_shift_range=0.0,brightness_range=(0.2,0.2),
                                   shear_range=0.2, zoom_range=0.2,
                                   channel_shift_range=0.0, fill_mode='nearest',
                                   cval=0.0, horizontal_flip=True, vertical_flip=True,
                                   data_format=None, validation_split=0.0, dtype=None)

val_gen=image.ImageDataGenerator(rescale=1./255,
                                   featurewise_center=True,
                                   samplewise_center=True,
                                   featurewise_std_normalization=True,
                                   samplewise_std_normalization=True,
                                   zca_whitening=False, zca_epsilon=1e-06,
                                   rotation_range=40, width_shift_range=0.0,
                                   height_shift_range=0.0,brightness_range=(0.2,0.2),
                                   shear_range=0.2, zoom_range=0.2,
                                   channel_shift_range=0.0, fill_mode='nearest',
                                   cval=0.0, horizontal_flip=True, vertical_flip=True,
                                   data_format=None, validation_split=0.0, dtype=None)

#Data Train
data_train=train_gen.flow_from_directory(r"C:\Users\User\Data\train\\",
                                         target_size=(32,32),
                                         classes=["covid","dog"],
                                         class_mode="categorical",
                                         batch_size=64,
                                         seed=1)

#Data val
data_val=val_gen.flow_from_directory(r"C:\Users\User\Data\val\\",
                                     target_size=(32,32),
                                         classes=["covid","dog"],
                                         class_mode="categorical",
                                         batch_size=64,
                                         seed=1)


#Model
model=Sequential()

#Conv1
model.add(ZeroPadding2D(padding=(2,2), input_shape=(32,32,3)))
model.add(Conv2D(5,kernel_size=(3,3),strides=(1,1),padding="SAME",
                activation="relu"))

model.add(MaxPool2D(pool_size=(2,2), strides=(1,1),padding="SAME"))
model.add(BatchNormalization(axis=-1))
model.add(Dropout(0.3))


#conv2
model.add(ZeroPadding2D(padding=(2,2)))
model.add(Conv2D(5,kernel_size=(3,3),strides=(1,1),padding="SAME",
                activation="relu"))

model.add(MaxPool2D(pool_size=(2,2), strides=(1,1),padding="SAME"))
model.add(BatchNormalization(axis=-1))
model.add(Dropout(0.3))

#Conv 3
model.add(ZeroPadding2D(padding=(2,2)))
model.add(Conv2D(3,kernel_size=(3,3),strides=(1,1),padding="SAME",
                activation="relu"))

model.add(MaxPool2D(pool_size=(2,2), strides=(1,1),padding="SAME"))
model.add(BatchNormalization(axis=-1))
model.add(Dropout(0.3))

#conv 4
model.add(ZeroPadding2D(padding=(2,2)))
model.add(Conv2D(3,kernel_size=(2,2),strides=(1,1),padding="SAME",
                activation="relu"))

model.add(MaxPool2D(pool_size=(2,2), strides=(1,1),padding="SAME"))
model.add(BatchNormalization(axis=-1))
model.add(Dropout(0.3))


#conv 5
model.add(ZeroPadding2D(padding=(2,2)))
model.add(Conv2D(3,kernel_size=(2,2),strides=(1,1),padding="SAME",
                activation="relu"))

model.add(MaxPool2D(pool_size=(2,2), strides=(1,1),padding="SAME"))
model.add(BatchNormalization(axis=-1))
model.add(Dropout(0.3))

#FLATTEN
model.add(Flatten())

#Dense 1
model.add(Dense(128, activation='relu'))
model.add(BatchNormalization(axis=-1))
model.add(Dropout(0.2))

#Dense 2
model.add(Dense(64, activation='relu'))
model.add(BatchNormalization(axis=-1))
model.add(Dropout(0.3))

#Dense 3
model.add(Dense(32, activation='relu'))
model.add(BatchNormalization(axis=-1))
model.add(Dropout(0.5))

#Densed
model.add(Dense(2, activation="sigmoid"))

#model summary
model.summary()

#optimizers
ad=Adam(lr=0.01)

#compile model
model.compile(optimizer=Ad,
             loss="categorical_crossentropy",
             metrics=["accuracy"])

#fitting Model
history=model.fit_generator(data_train, steps_per_epcoh=100,
                           epcohs=3, validation_data=data_val,
                           validation_steps=5)
```

```
Found 151 images belonging to 2 classes.
Found 0 images belonging to 2 classes.
Model: "sequential_5"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 zero_padding2d_21 (ZeroPadd  (None, 36, 36, 3)        0
 ing2D)

 conv2d_20 (Conv2D)          (None, 36, 36, 5)         140

 max_pooling2d_20 (MaxPoolin  (None, 36, 36, 5)        0
 g2D)

 batch_normalization_25 (Bat  (None, 36, 36, 5)        20
 chNormalization)

 dropout_23 (Dropout)        (None, 36, 36, 5)         0

 zero_padding2d_22 (ZeroPadd  (None, 40, 40, 5)        0
 ing2D)

 conv2d_21 (Conv2D)          (None, 40, 40, 5)         230

 max_pooling2d_21 (MaxPoolin  (None, 40, 40, 5)        0
 g2D)

 batch_normalization_26 (Bat  (None, 40, 40, 5)        20
 chNormalization)

 dropout_24 (Dropout)        (None, 40, 40, 5)         0

 zero_padding2d_23 (ZeroPadd  (None, 44, 44, 5)        0
 ing2D)

 conv2d_22 (Conv2D)          (None, 44, 44, 3)         138

 max_pooling2d_22 (MaxPoolin  (None, 44, 44, 3)        0
 g2D)

 batch_normalization_27 (Bat  (None, 44, 44, 3)        12
 chNormalization)

 dropout_25 (Dropout)        (None, 44, 44, 3)         0

 zero_padding2d_24 (ZeroPadd  (None, 48, 48, 3)        0
 ing2D)

 conv2d_23 (Conv2D)          (None, 48, 48, 3)         39

 max_pooling2d_23 (MaxPoolin  (None, 48, 48, 3)        0
 g2D)

 batch_normalization_28 (Bat  (None, 48, 48, 3)        12
 chNormalization)

 dropout_26 (Dropout)        (None, 48, 48, 3)         0

 zero_padding2d_25 (ZeroPadd  (None, 52, 52, 3)        0
 ing2D)

 conv2d_24 (Conv2D)          (None, 52, 52, 3)         39

 max_pooling2d_24 (MaxPoolin  (None, 52, 52, 3)        0
 g2D)

 batch_normalization_29 (Bat  (None, 52, 52, 3)        12
 chNormalization)

 dropout_27 (Dropout)        (None, 52, 52, 3)         0

 flatten_4 (Flatten)         (None, 8112)              0

 dense_7 (Dense)             (None, 128)               1038464

 batch_normalization_30 (Bat  (None, 128)              512
 chNormalization)

 dropout_28 (Dropout)        (None, 128)               0

 dense_8 (Dense)             (None, 64)                8256

 batch_normalization_31 (Bat  (None, 64)               256
 chNormalization)

 dropout_29 (Dropout)        (None, 64)                0

 dense_9 (Dense)             (None, 32)                2080

 batch_normalization_32 (Bat  (None, 32)               128
 chNormalization)

 dropout_30 (Dropout)        (None, 32)                0

 dense_10 (Dense)            (None, 2)                 66

=================================================================
Total params: 1,050,424
Trainable params: 1,049,938
Non-trainable params: 486
_________________________________________________________________
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_8116/950378390.py in <module>
    129
    130 #compile model
--> 131 model.compile(optimizer=Ad,
    132              loss="categorical_crossentropy",
    133              metrics=["accuracy"])

NameError: name 'Ad' is not defined
```

In [1]:

```python
import matplotlib.pyplot as plt
history=model.fit()
print(history.history.keys())

#summarize history for accuracy
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('model_accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['train','test'],loc='upper left')
plt.show()

#Summarize history for loss
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('model_loss')
plt.ylabel('loss')
plt.legend(["train","test"],loc="upper left")
plt.show()
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_8888/1681388505.py in <module>
      1 import matplotlib.pyplot as plt
----> 2 history=model.fit()
      3 print(history.history.keys())
      4
      5 #summarize history for accuracy

NameError: name 'model' is not defined
```

In [31]:

```python
test_images=[os.path.join(os.getcwd(),"test",image)
                    for image in os.listdir("./test/")
                    if image.endswith(".jpg") or image.endswith(".png")
                    or image.endswith(".jpeg")]
```

In [1]:

```python
import os.path
if os.path.isfile("covid_normal_model.h5") is False:
    model.save("covid_normal_model.h5")
```

In [ ]:

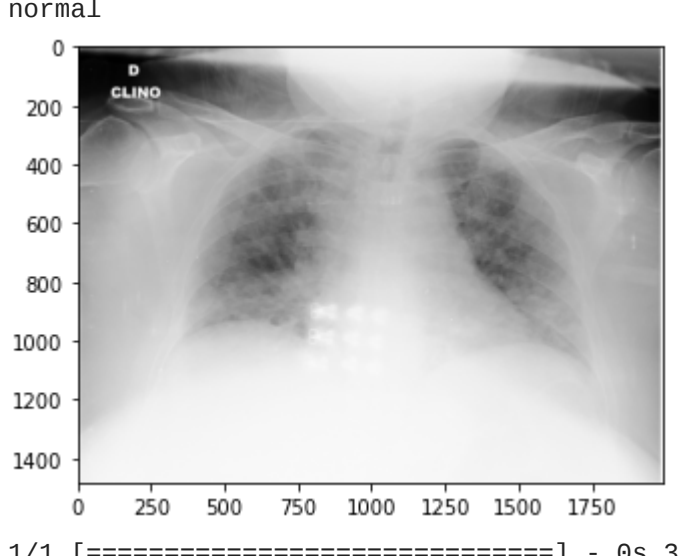
```python
from tensor
```

In [32]:

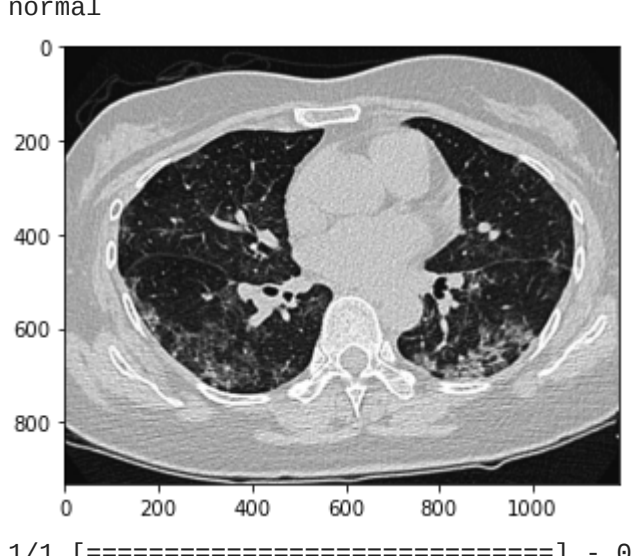
```python
import matplotlib.pyplot as plt
classes=["covid","normal"]

for i in range(0,10):
    img=cv2.imread(test_images[i])
    img=cv2.resize(img, (32, 32))
    img=np.expand_dims(img, axis=0)
    print(model.predict(img),np.argmax(model.predict(img)))
    print(classes[np.argmax(model.predict(img))])
    plt.imshow(cv2.imread(test_images[i]))
    plt.show()
```

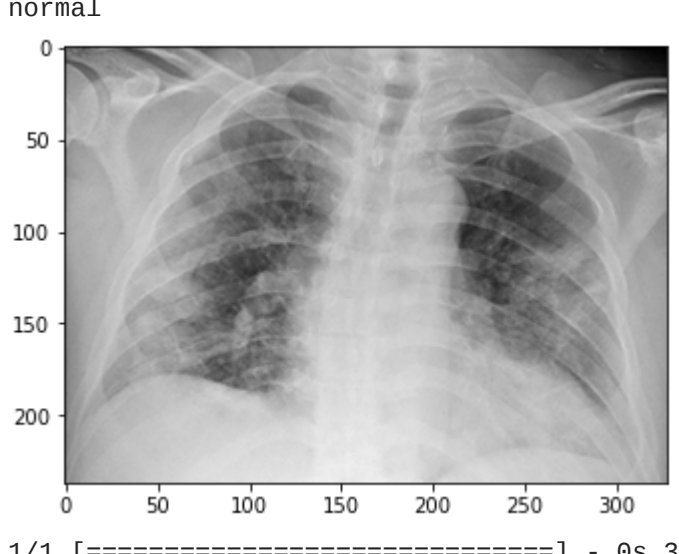
```
1/1 [==============================] - 0s 334ms/step
1/1 [==============================] - 0s 36ms/step
[[0.6305605  0.94976306]] 1
1/1 [==============================] - 0s 35ms/step
normal
```



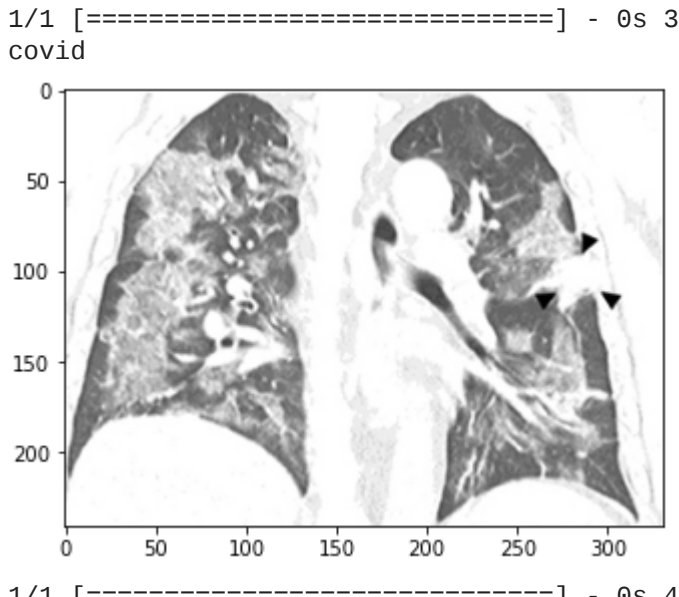

```
1/1 [==============================] - 0s 33ms/step
1/1 [==============================] - 0s 33ms/step
[[0.21272689 0.85885704]] 1
1/1 [==============================] - 0s 34ms/step
normal
```



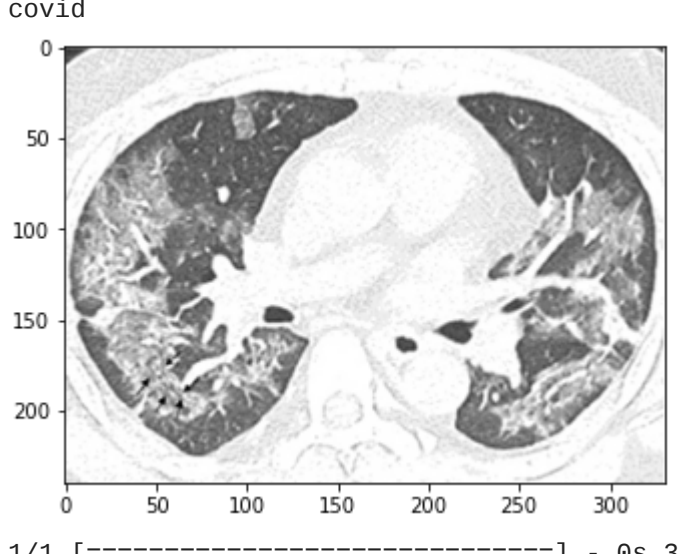
```
1/1 [==============================] - 0s 33ms/step
1/1 [==============================] - 0s 33ms/step
[[0.8097797 0.9284845]] 1
1/1 [==============================] - 0s 33ms/step
normal
```



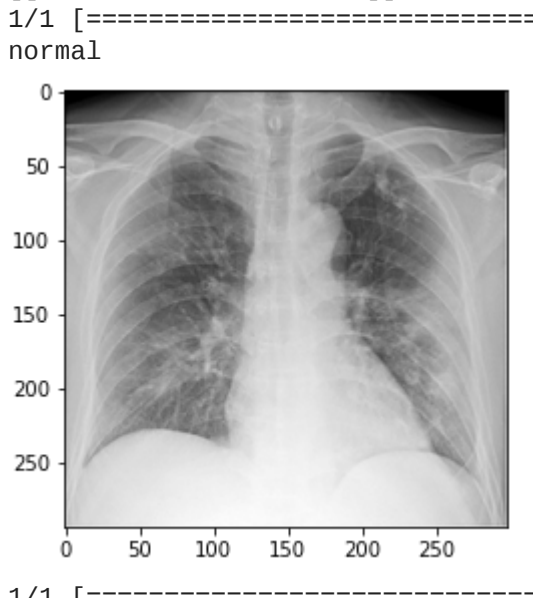
```
1/1 [==============================] - 0s 34ms/step
1/1 [==============================] - 0s 34ms/step
[[0.9941465 0.980127 ]] 0
1/1 [==============================] - 0s 34ms/step
covid
```



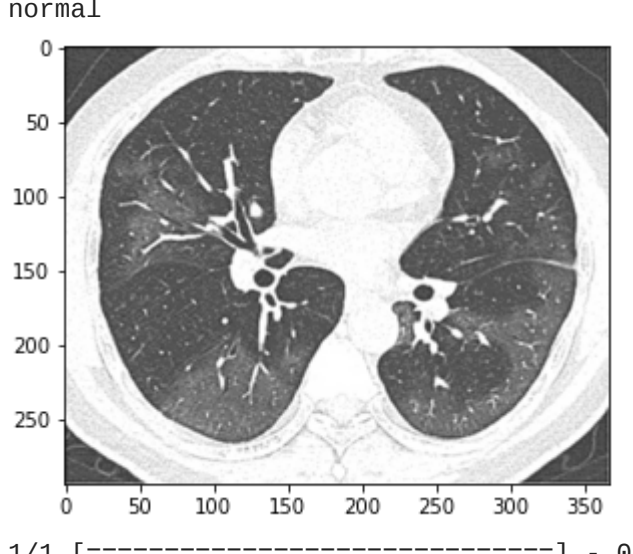
```
1/1 [==============================] - 0s 41ms/step
1/1 [==============================] - 0s 38ms/step
[[0.94431317 0.90047544]] 0
1/1 [==============================] - 0s 33ms/step
covid
```



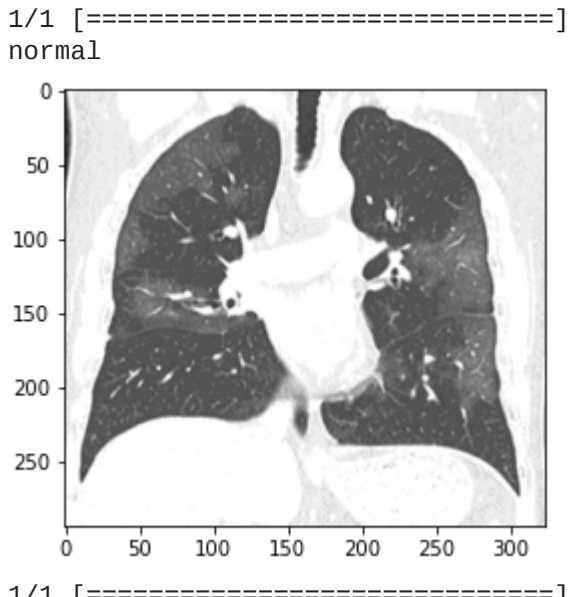
```
1/1 [==============================] - 0s 35ms/step
1/1 [==============================] - 0s 33ms/step
[[0.22269739 0.9258754 ]] 1
1/1 [==============================] - 0s 33ms/step
normal
```



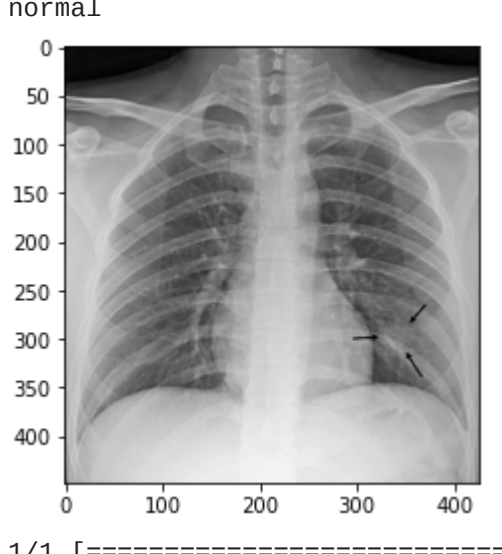
```
1/1 [==============================] - 0s 35ms/step
1/1 [==============================] - 0s 33ms/step
[[0.31283256 0.8605097 ]] 1
1/1 [==============================] - 0s 32ms/step
normal
```



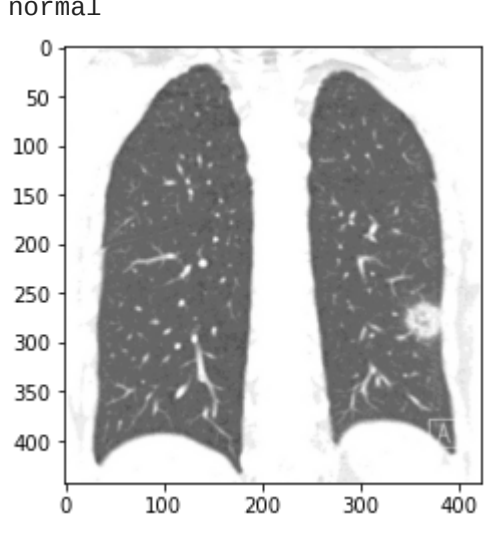
```
1/1 [==============================] - 0s 35ms/step
1/1 [==============================] - 0s 35ms/step
[[0.26781756 0.9589037 ]] 1
1/1 [==============================] - 0s 35ms/step
normal
```



```
1/1 [==============================] - 0s 34ms/step
1/1 [==============================] - 0s 34ms/step
[[0.25042903 0.8952273 ]] 1
1/1 [==============================] - 0s 36ms/step
normal
```



```
1/1 [==============================] - 0s 33ms/step
1/1 [==============================] - 0s 33ms/step
[[0.942094  0.9835716]] 1
1/1 [==============================] - 0s 35ms/step
normal
```



In [ ]: