Load_Test

April 21, 2021

0.0.1 Importing the required libraries

[2]:
```
import tensorflow as tf
import random
import cv2
from random import shuffle
import numpy as np
import os
import matplotlib.pyplot as plt
```

0.0.2 Data Process the test dataset

[3]:
```
single=r"C:/Users/omara/test"
#quick note on data cleaning, you need to first convert the dataset to PNG,␣
 ↪then to prevent incomptiable input shapes convert from RGB to Grayscale,␣
 ↪then downsample to not fry you CPU with 1000s of images of millions of␣
 ↪arrays elements.
test=[]
cat=["healthypng","covidpng"]
Model="CT"
for i in cat:
    path_2=os.path.join(single,i)
    label=cat.index(i)
    for img in os.listdir(path_2): # for test dataset
        img_path=os.path.join(path_2,img)
        img_arr=cv2.imread(img_path)# will result in input shape(256,256)␣
 ↪instead of (256,256,3) aka RGB
        #img_arr=tf.image.resize(img_arr, [256,256])#no need
        test.append([img_arr,label])
shuffle(test)# no need in acutuality but to stay consistent
x_test=[]
y_test=[]
for features,labels in test:
    x_test.append(features)
    y_test.append(labels)
x_test=np.array(x_test)
y_test=np.array(y_test)
x_test=x_test/255
```

0.0.3 Load the model

[4]:
```
model=tf.keras.models.load_model("ULT_CT_CONV")
```

0.0.4 Test the model

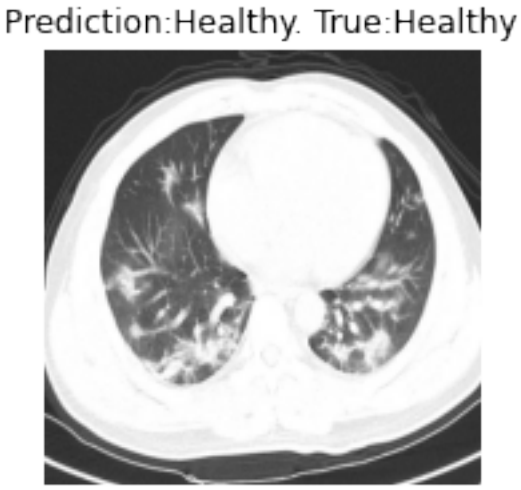
[5]:
```
loss,win=model.evaluate(x_test,y_test)# test using evaluate
model.summary()
```

```
4/4 [==============================] - 3s 108ms/step - loss: 0.7819 - accuracy:
0.8416
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 256, 256, 64)      1792
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 128, 128, 64)      0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 128, 128, 64)      36928
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 64, 64, 64)        0
_________________________________________________________________
flatten (Flatten)            (None, 262144)            0
_________________________________________________________________
dense (Dense)                (None, 64)                16777280
_________________________________________________________________
dense_1 (Dense)              (None, 64)                4160
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 65
=================================================================
Total params: 16,820,225
Trainable params: 16,820,225
Non-trainable params: 0
_________________________________________________________________
```

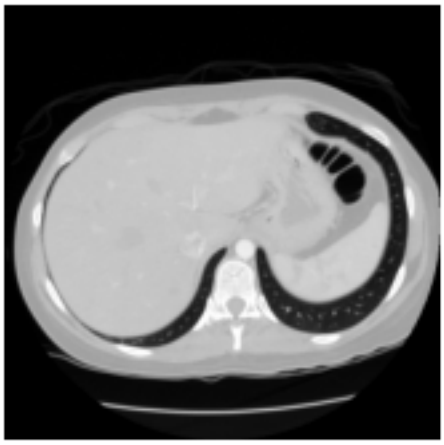
0.0.5 Some stats

[6]:
```
predict=model.predict([x_test])
import random
def stuff(n):
    if n==0:
        return "Healthy"
    elif n==1:
        return "Sick"
    return "Healthy"
for i in range(25):
    plt.figure(figsize = (3,3))
```

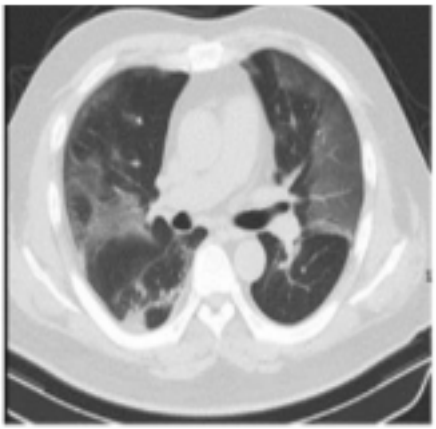
```
    val=random.randint(0,len(x_test)-1)
    plt.axis("off")
    plt.imshow(x_test[val], cmap='gray',)
    plt.title("Prediction:{}. True:{}".
↪format(stuff(predict[val]),stuff(y_test[val])))
    plt.show()
# varification
```



Prediction:Sick. True:Sick

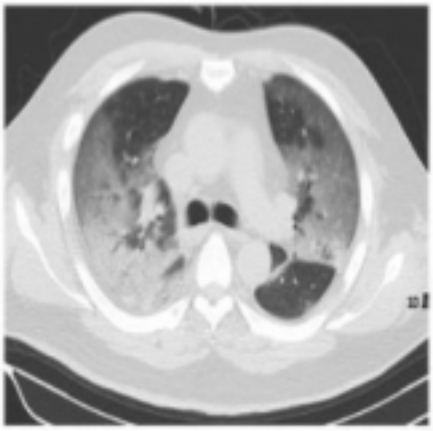


Prediction:Healthy. True:Healthy

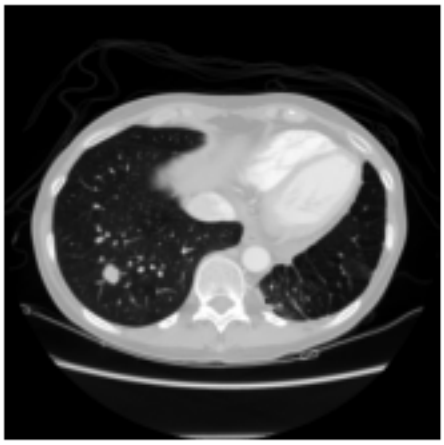


Prediction:Healthy. True:Healthy

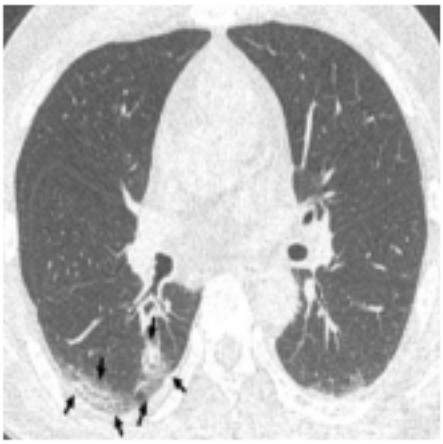
Prediction:Healthy. True:Healthy



Prediction:Sick. True:Sick

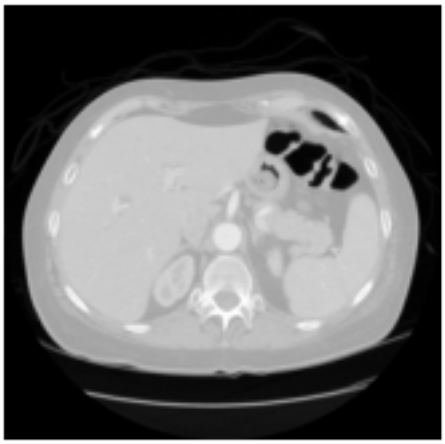


Prediction:Healthy. True:Healthy

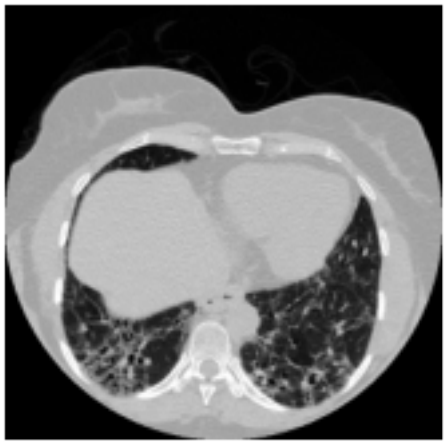


Prediction:Sick. True:Sick

Prediction:Healthy. True:Sick



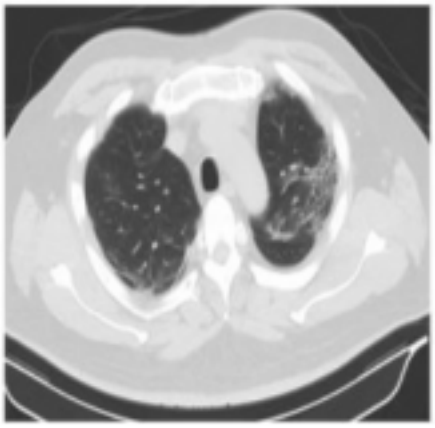
Prediction:Healthy. True:Sick



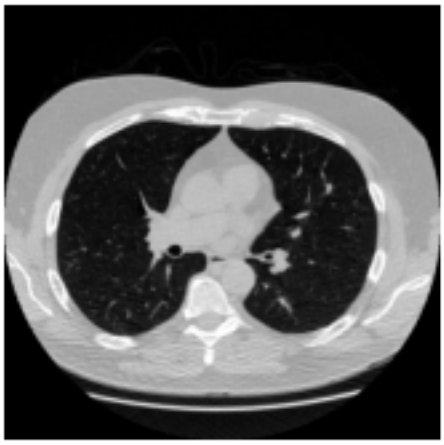
Prediction:Sick. True:Sick

Prediction:Sick. True:Sick

Prediction:Healthy. True:Healthy



Prediction:Sick. True:Sick

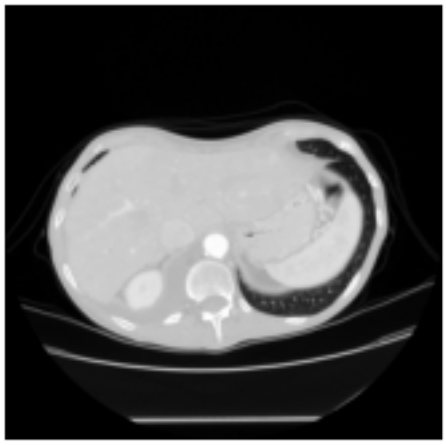


Prediction:Sick. True:Sick

Prediction:Healthy. True:Healthy

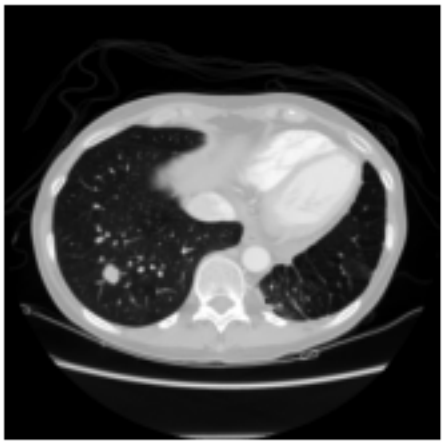
Prediction:Sick. True:Sick

Prediction:Healthy. True:Sick

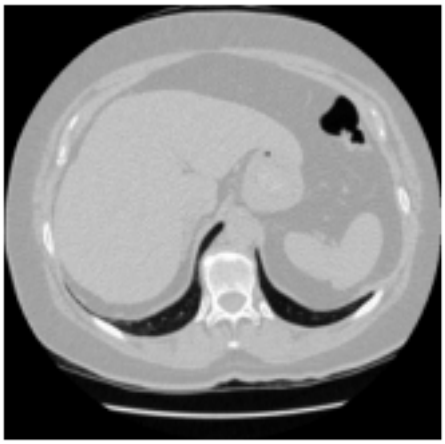


Prediction:Sick. True:Sick

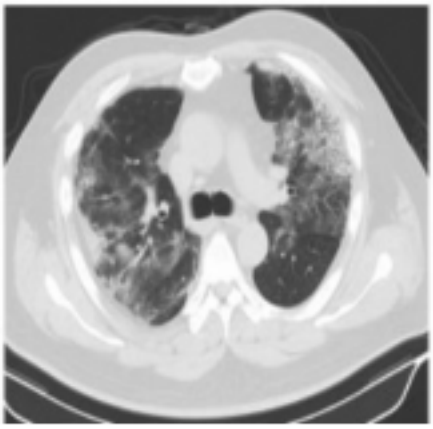
Prediction:Sick. True:Sick



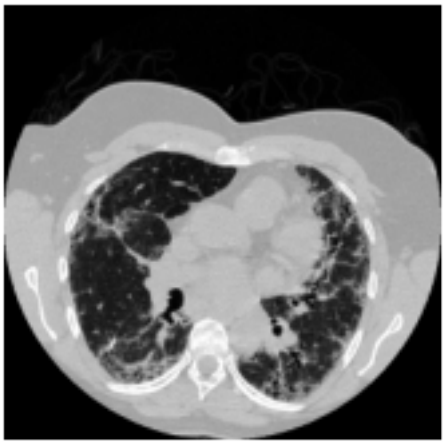
Prediction:Healthy. True:Sick



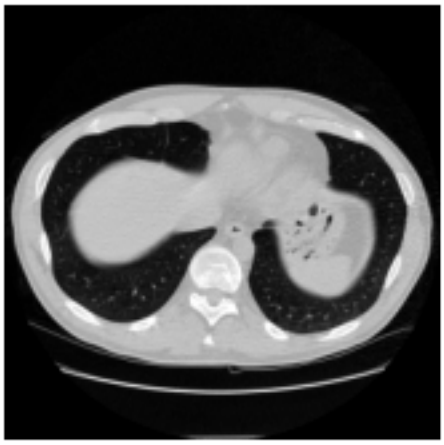
Prediction:Healthy. True:Healthy



Prediction:Healthy. True:Sick



Prediction:Sick. True:Sick



[ ]: