task1_template

April 17, 2020

1 Class Challenge: Image Classification of COVID-19 X-rays

2 Task 1 [Total points: 30]

2.1 Setup

- This assignment involves the following packages: 'matplotlib', 'numpy', and 'sklearn'.
- If you are using conda, use the following commands to install the above packages:

```
conda install matplotlib
conda install numpy
conda install -c anaconda scikit-learn
```

- If you are using pip, use use the following commands to install the above packages:

```
pip install matplotlib
pip install numpy
pip install sklearn
```

2.2 Data

Please download the data using the following link: COVID-19.

- After downloading 'Covid_Data_GradientCrescent.zip', unzip the file and you should see the following data structure:

|–all |———train |———test |–two |———train |———test

- Put the 'all' folder, the 'two' folder and this python notebook in the **same directory** so that the following code can correctly locate the data.

2.3 [20 points] Binary Classification: COVID-19 vs. Normal

[1]:
```
import os

import tensorflow as tf
```

```
import numpy as np
import matplotlib.pyplot as plt
from tensorflow.keras.preprocessing.image import ImageDataGenerator

os.environ['OMP_NUM_THREADS'] = '1'
os.environ['CUDA_VISIBLE_DEVICES'] = '-1'
tf.__version__
```

[1]: '2.1.0'

Load Image Data

[6]:
```
DATA_LIST = os.listdir('two/train')
DATASET_PATH  = 'two/train'
TEST_DIR =  'two/test'
IMAGE_SIZE    = (224, 224)
NUM_CLASSES   = len(DATA_LIST)
BATCH_SIZE    = 10  # try reducing batch size or freeze more layers if your GPU␣
 ↪runs out of memory
NUM_EPOCHS    = 40
LEARNING_RATE = 0.0005 # start off with high rate first 0.001 and experiment␣
 ↪with reducing it gradually
```

Generate Training and Validation Batches

[7]:
```
train_datagen = ImageDataGenerator(rescale=1./
 ↪255,rotation_range=50,featurewise_center = True,
                                    featurewise_std_normalization =␣
 ↪True,width_shift_range=0.2,
                                    height_shift_range=0.2,shear_range=0.
 ↪25,zoom_range=0.1,
                                    zca_whitening = True,channel_shift_range = 20,
                                    horizontal_flip = True,vertical_flip = True,
                                    validation_split = 0.2,fill_mode='constant')

train_batches = train_datagen.
 ↪flow_from_directory(DATASET_PATH,target_size=IMAGE_SIZE,
                                                    ␣
 ↪shuffle=True,batch_size=BATCH_SIZE,
                                                     subset = "training",seed=42,
                                                     class_mode="binary")

valid_batches = train_datagen.
 ↪flow_from_directory(DATASET_PATH,target_size=IMAGE_SIZE,
                                                    ␣
 ↪shuffle=True,batch_size=BATCH_SIZE,
```

```
                                                    subset = "validation",seed=42,
                                                    class_mode="binary")
```

```
H:\Anaconda3\envs\tf\lib\site-
packages\keras_preprocessing\image\image_data_generator.py:341: UserWarning:
This ImageDataGenerator specifies `zca_whitening` which overrides setting
of`featurewise_std_normalization`.
  warnings.warn('This ImageDataGenerator specifies '
```

```
Found 104 images belonging to 2 classes.
Found 26 images belonging to 2 classes.
```

[10 points] Build Model Hint: Starting from a pre-trained model typically helps performance on a new task, e.g. starting with weights obtained by training on ImageNet.

[8]:
```
raise NotImplementedError("Build your model based on an architecture of your␣
 ↪choice "
                          "A sample model summary is shown below")
```

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
vgg16 (Model)                (None, 7, 7, 512)         14714688
_________________________________________________________________
flatten_1 (Flatten)          (None, 25088)             0
_________________________________________________________________
dense_feature (Dense)        (None, 256)               6422784
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 257
=================================================================
Total params: 21,137,729
Trainable params: 6,423,041
Non-trainable params: 14,714,688
_________________________________________________________________
None
```

[5 points] Train Model

[9]:
```
#FIT MODEL
print(len(train_batches))
print(len(valid_batches))

STEP_SIZE_TRAIN=train_batches.n//train_batches.batch_size
STEP_SIZE_VALID=valid_batches.n//valid_batches.batch_size

raise NotImplementedError("Use the model.fit function to train your network")
```

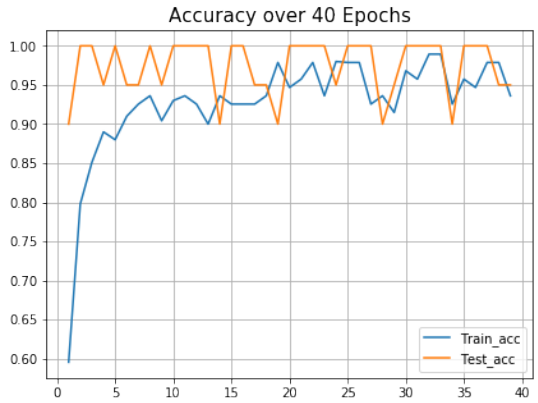
```
11
3
WARNING:tensorflow:sample_weight modes were coerced from
  ...
    to
  ['...']
WARNING:tensorflow:sample_weight modes were coerced from
  ...
    to
  ['...']
Train for 10 steps, validate for 2 steps
Epoch 1/40

H:\Anaconda3\envs\tf\lib\site-
packages\keras_preprocessing\image\image_data_generator.py:716: UserWarning:
This ImageDataGenerator specifies `featurewise_center`, but it hasn't been fit
on any training data. Fit it first by calling `.fit(numpy_data)`.
  warnings.warn('This ImageDataGenerator specifies '
H:\Anaconda3\envs\tf\lib\site-
packages\keras_preprocessing\image\image_data_generator.py:735: UserWarning:
This ImageDataGenerator specifies `zca_whitening`, but it hasn't been fit on any
training data. Fit it first by calling `.fit(numpy_data)`.
  warnings.warn('This ImageDataGenerator specifies '

10/10 [==============================] - 33s 3s/step - loss: 1.8210 - acc:
0.4362 - val_loss: 0.6237 - val_acc: 0.5500
Epoch 2/40
10/10 [==============================] - 31s 3s/step - loss: 0.6589 - acc:
0.5957 - val_loss: 0.4066 - val_acc: 0.9000
Epoch 3/40
10/10 [==============================] - 31s 3s/step - loss: 0.4876 - acc:
0.7979 - val_loss: 0.2412 - val_acc: 1.0000
Epoch 4/40
10/10 [==============================] - 30s 3s/step - loss: 0.4062 - acc:
0.8511 - val_loss: 0.2238 - val_acc: 1.0000
Epoch 5/40
10/10 [==============================] - 31s 3s/step - loss: 0.3616 - acc:
0.8900 - val_loss: 0.2199 - val_acc: 0.9500
Epoch 6/40
10/10 [==============================] - 31s 3s/step - loss: 0.3085 - acc:
0.8800 - val_loss: 0.1579 - val_acc: 1.0000
Epoch 7/40
10/10 [==============================] - 32s 3s/step - loss: 0.2819 - acc:
0.9100 - val_loss: 0.2051 - val_acc: 0.9500
Epoch 8/40
10/10 [==============================] - 33s 3s/step - loss: 0.2104 - acc:
0.9255 - val_loss: 0.1438 - val_acc: 0.9500
Epoch 9/40
10/10 [==============================] - 32s 3s/step - loss: 0.1874 - acc:
```

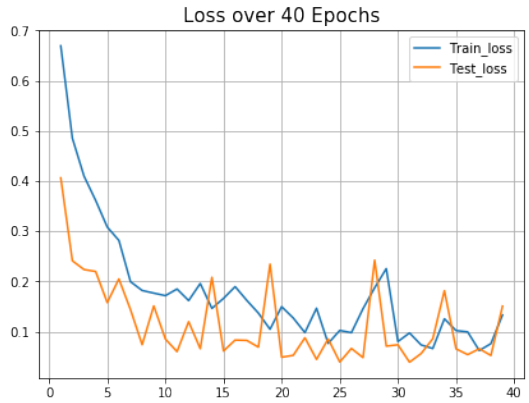
```
0.9362 - val_loss: 0.0740 - val_acc: 1.0000
Epoch 10/40
10/10 [==============================] - 33s 3s/step - loss: 0.1720 - acc:
0.9043 - val_loss: 0.1513 - val_acc: 0.9500
Epoch 11/40
10/10 [==============================] - 36s 4s/step - loss: 0.1718 - acc:
0.9300 - val_loss: 0.0854 - val_acc: 1.0000
Epoch 12/40
10/10 [==============================] - 33s 3s/step - loss: 0.1768 - acc:
0.9362 - val_loss: 0.0602 - val_acc: 1.0000
Epoch 13/40
10/10 [==============================] - 34s 3s/step - loss: 0.1566 - acc:
0.9255 - val_loss: 0.1199 - val_acc: 1.0000
Epoch 14/40
10/10 [==============================] - 35s 3s/step - loss: 0.1960 - acc:
0.9000 - val_loss: 0.0660 - val_acc: 1.0000
Epoch 15/40
10/10 [==============================] - 32s 3s/step - loss: 0.1450 - acc:
0.9362 - val_loss: 0.2083 - val_acc: 0.9000
Epoch 16/40
10/10 [==============================] - 35s 4s/step - loss: 0.1648 - acc:
0.9255 - val_loss: 0.0614 - val_acc: 1.0000
Epoch 17/40
10/10 [==============================] - 35s 3s/step - loss: 0.1979 - acc:
0.9255 - val_loss: 0.0833 - val_acc: 1.0000
Epoch 18/40
10/10 [==============================] - 34s 3s/step - loss: 0.1592 - acc:
0.9255 - val_loss: 0.0826 - val_acc: 0.9500
Epoch 19/40
10/10 [==============================] - 35s 4s/step - loss: 0.1354 - acc:
0.9362 - val_loss: 0.0692 - val_acc: 0.9500
Epoch 20/40
10/10 [==============================] - 37s 4s/step - loss: 0.1001 - acc:
0.9787 - val_loss: 0.2345 - val_acc: 0.9000
Epoch 21/40
10/10 [==============================] - 33s 3s/step - loss: 0.1449 - acc:
0.9468 - val_loss: 0.0490 - val_acc: 1.0000
Epoch 22/40
10/10 [==============================] - 32s 3s/step - loss: 0.1216 - acc:
0.9574 - val_loss: 0.0529 - val_acc: 1.0000
Epoch 23/40
10/10 [==============================] - 34s 3s/step - loss: 0.0950 - acc:
0.9787 - val_loss: 0.0878 - val_acc: 1.0000
Epoch 24/40
10/10 [==============================] - 34s 3s/step - loss: 0.1392 - acc:
0.9362 - val_loss: 0.0448 - val_acc: 1.0000
Epoch 25/40
10/10 [==============================] - 35s 4s/step - loss: 0.0765 - acc:
```

```
0.9800 - val_loss: 0.0856 - val_acc: 0.9500
Epoch 26/40
10/10 [==============================] - 33s 3s/step - loss: 0.1000 - acc:
0.9787 - val_loss: 0.0395 - val_acc: 1.0000
Epoch 27/40
10/10 [==============================] - 33s 3s/step - loss: 0.0929 - acc:
0.9787 - val_loss: 0.0665 - val_acc: 1.0000
Epoch 28/40
10/10 [==============================] - 34s 3s/step - loss: 0.1672 - acc:
0.9255 - val_loss: 0.0483 - val_acc: 1.0000
Epoch 29/40
10/10 [==============================] - 34s 3s/step - loss: 0.1796 - acc:
0.9362 - val_loss: 0.2426 - val_acc: 0.9000
Epoch 30/40
10/10 [==============================] - 33s 3s/step - loss: 0.2180 - acc:
0.9149 - val_loss: 0.0712 - val_acc: 0.9500
Epoch 31/40
10/10 [==============================] - 32s 3s/step - loss: 0.0774 - acc:
0.9681 - val_loss: 0.0739 - val_acc: 1.0000
Epoch 32/40
10/10 [==============================] - 34s 3s/step - loss: 0.0928 - acc:
0.9574 - val_loss: 0.0392 - val_acc: 1.0000
Epoch 33/40
10/10 [==============================] - 35s 4s/step - loss: 0.0719 - acc:
0.9894 - val_loss: 0.0564 - val_acc: 1.0000
Epoch 34/40
10/10 [==============================] - 32s 3s/step - loss: 0.0638 - acc:
0.9894 - val_loss: 0.0859 - val_acc: 1.0000
Epoch 35/40
10/10 [==============================] - 30s 3s/step - loss: 0.1190 - acc:
0.9255 - val_loss: 0.1817 - val_acc: 0.9000
Epoch 36/40
10/10 [==============================] - 30s 3s/step - loss: 0.0972 - acc:
0.9574 - val_loss: 0.0658 - val_acc: 1.0000
Epoch 37/40
10/10 [==============================] - 30s 3s/step - loss: 0.0952 - acc:
0.9468 - val_loss: 0.0542 - val_acc: 1.0000
Epoch 38/40
10/10 [==============================] - 29s 3s/step - loss: 0.0625 - acc:
0.9787 - val_loss: 0.0660 - val_acc: 1.0000
Epoch 39/40
10/10 [==============================] - 30s 3s/step - loss: 0.0728 - acc:
0.9787 - val_loss: 0.0525 - val_acc: 0.9500
Epoch 40/40
10/10 [==============================] - 29s 3s/step - loss: 0.1265 - acc:
0.9362 - val_loss: 0.1509 - val_acc: 0.9500
```

[5 points] Plot Accuracy and Loss During Training

[11]:
```
import matplotlib.pyplot as plt

raise NotImplementedError("Plot the accuracy and the loss during training")
```



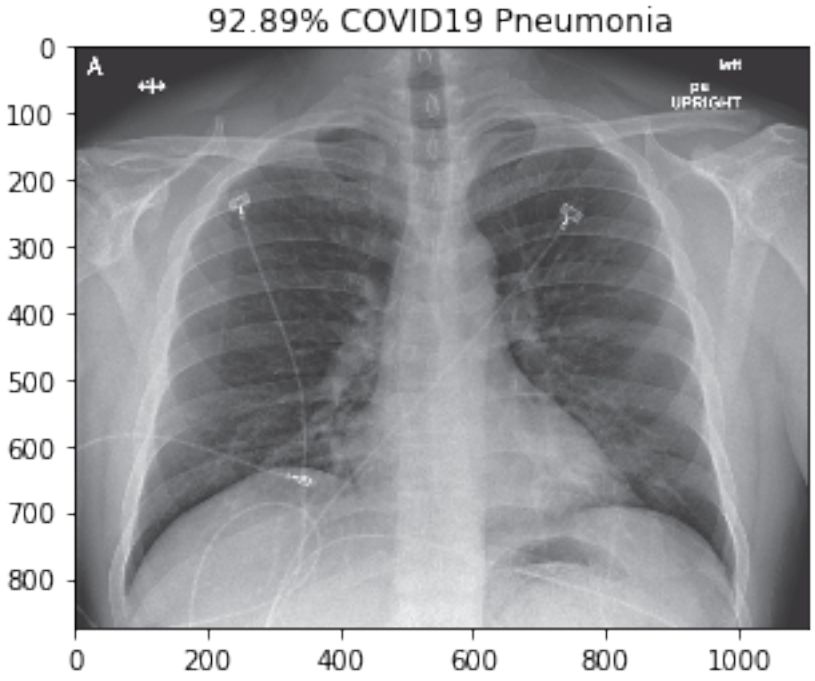


Plot Test Results

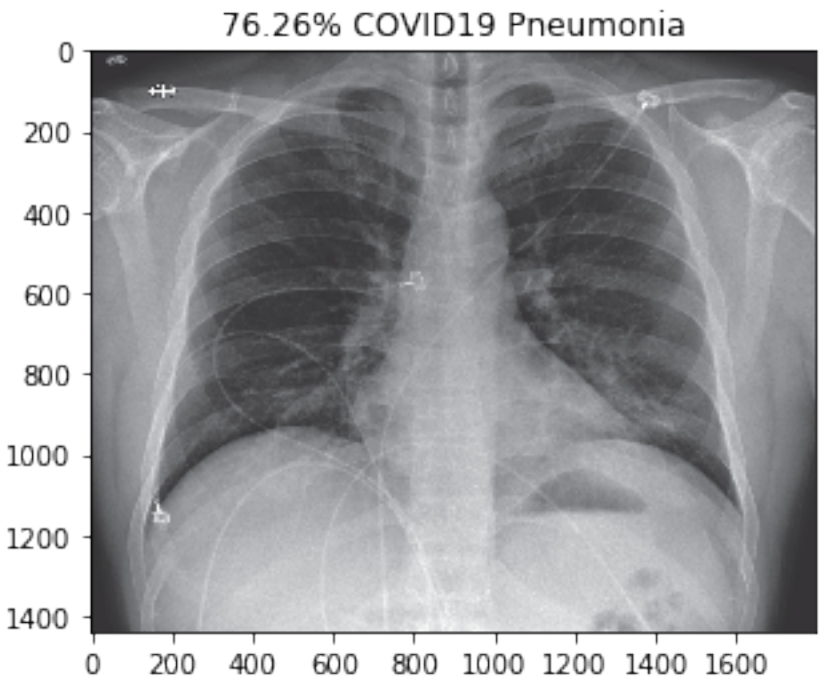
[12]:
```
import matplotlib.image as mpimg

test_datagen = ImageDataGenerator(rescale=1. / 255)
eval_generator = test_datagen.
 ↪flow_from_directory(TEST_DIR,target_size=IMAGE_SIZE,
                                                    ␣
 ↪batch_size=1,shuffle=True,seed=42,class_mode="binary")
eval_generator.reset()
pred = model.predict_generator(eval_generator,18,verbose=1)
for index, probability in enumerate(pred):
    image_path = TEST_DIR + "/" +eval_generator.filenames[index]
    image = mpimg.imread(image_path)
    if image.ndim < 3:
        image = np.reshape(image,(image.shape[0],image.shape[1],1))
        image = np.concatenate([image, image, image], 2)
#         print(image.shape)

    pixels = np.array(image)
    plt.imshow(pixels)

    print(eval_generator.filenames[index])
    if probability > 0.5:
        plt.title("%.2f" % (probability[0]*100) + "% Normal")
    else:
        plt.title("%.2f" % ((1-probability[0])*100) + "% COVID19 Pneumonia")
```

```
    plt.show()
```

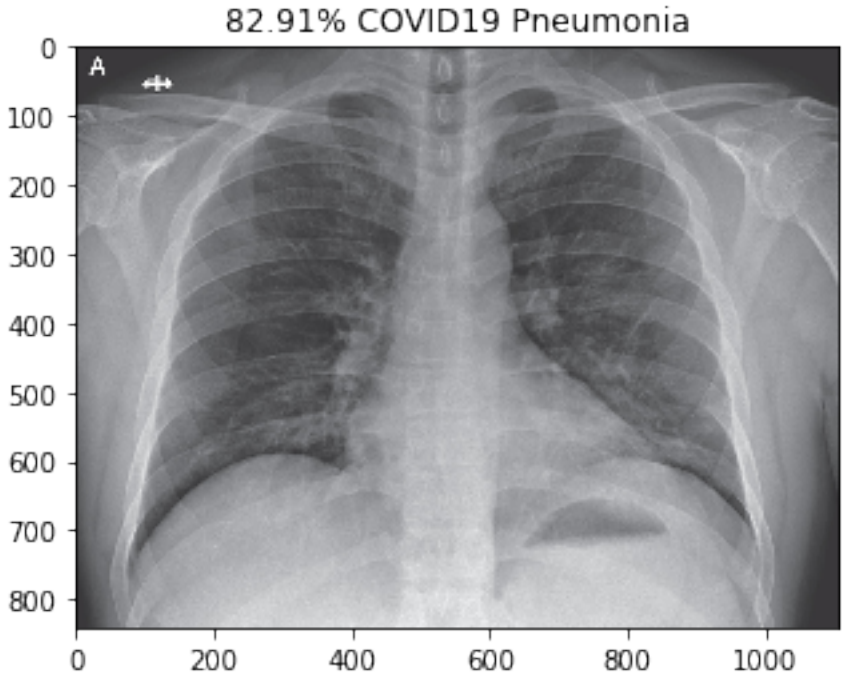
```
Found 18 images belonging to 2 classes.
WARNING:tensorflow:From <ipython-input-12-543347a5fba8>:7:
Model.predict_generator (from tensorflow.python.keras.engine.training) is
deprecated and will be removed in a future version.
Instructions for updating:
Please use Model.predict, which supports generators.
18/18 [==============================] - 5s 292ms/step
covid\nejmoa2001191_f3-PA.jpeg
```



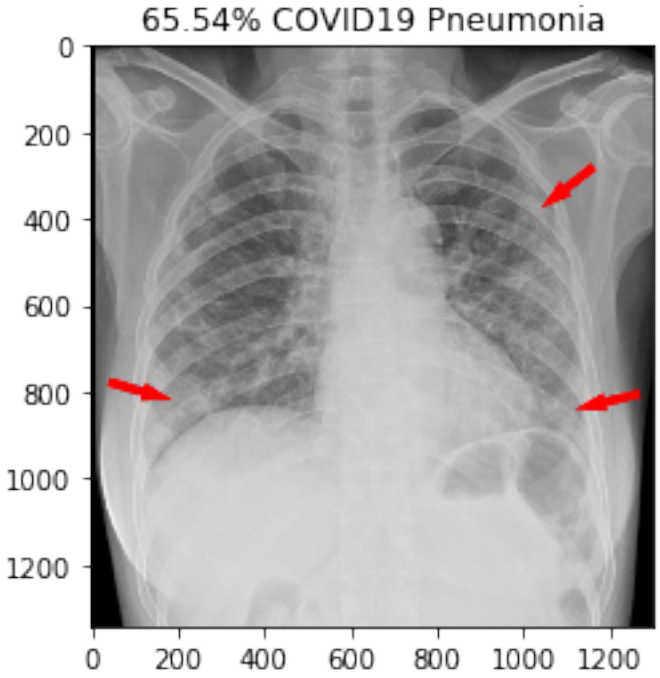

```
covid\nejmoa2001191_f4.jpeg
```



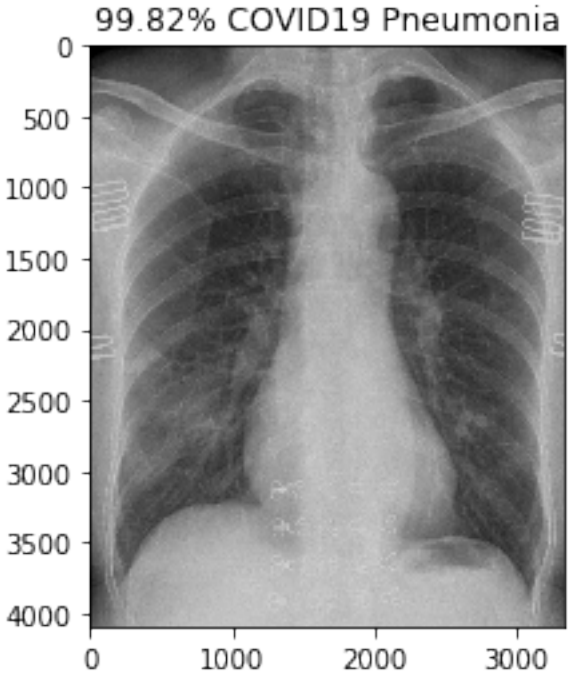
covid\nejmoa2001191_f5-PA.jpeg



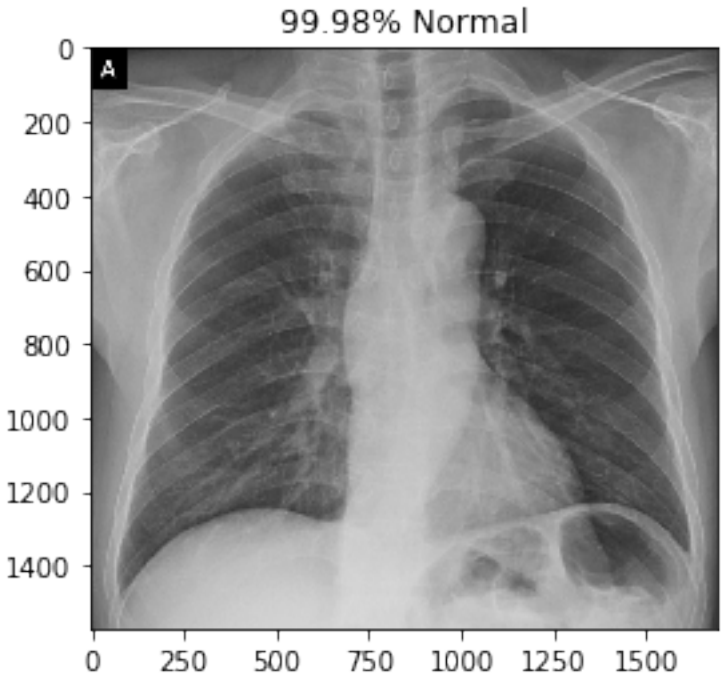
covid\radiol.2020200490.fig3.jpeg



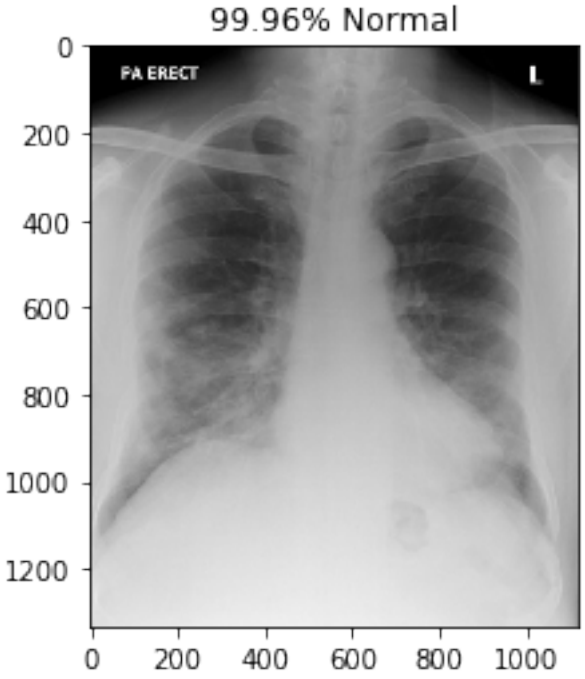
covid\ryct.2020200028.fig1a.jpeg



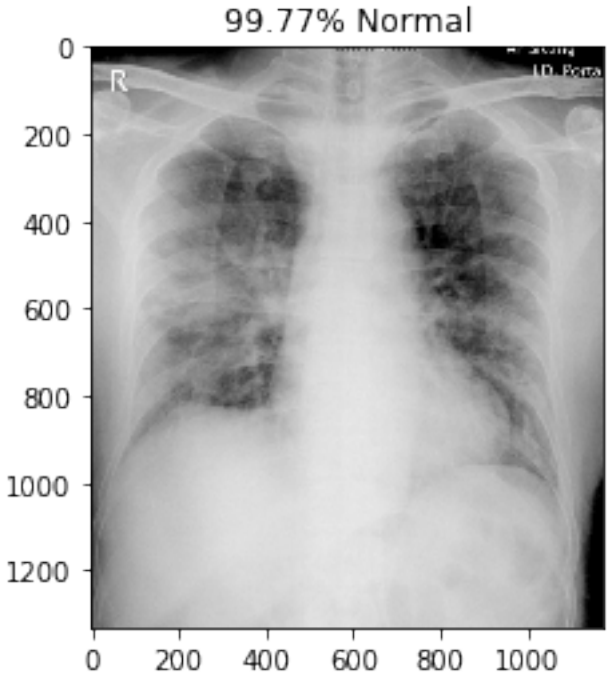
covid\ryct.2020200034.fig2.jpeg



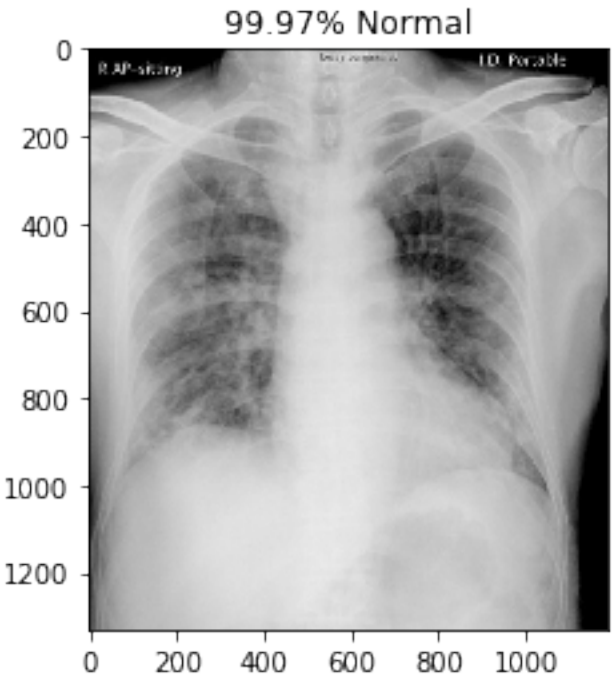
covid\ryct.2020200034.fig5-day0.jpeg



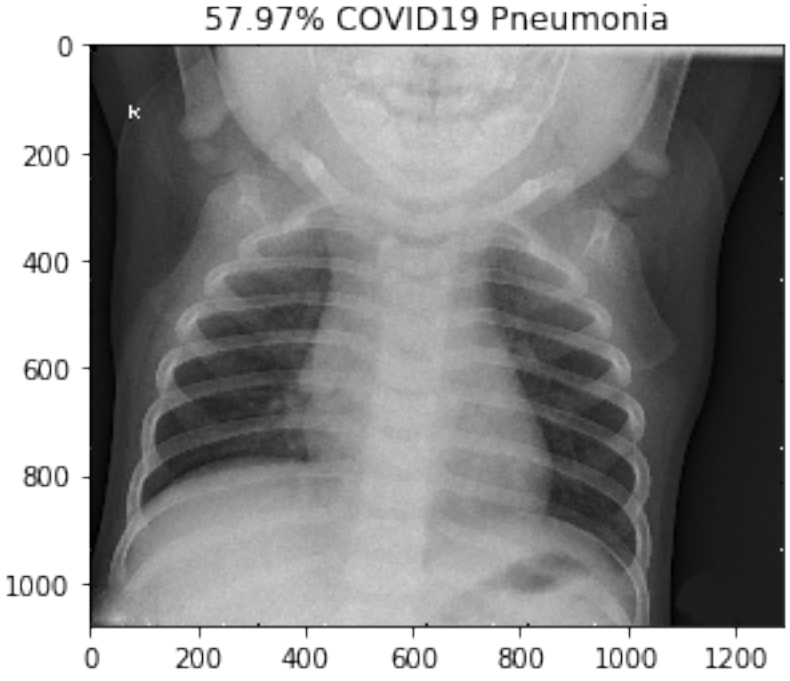
covid\ryct.2020200034.fig5-day4.jpeg



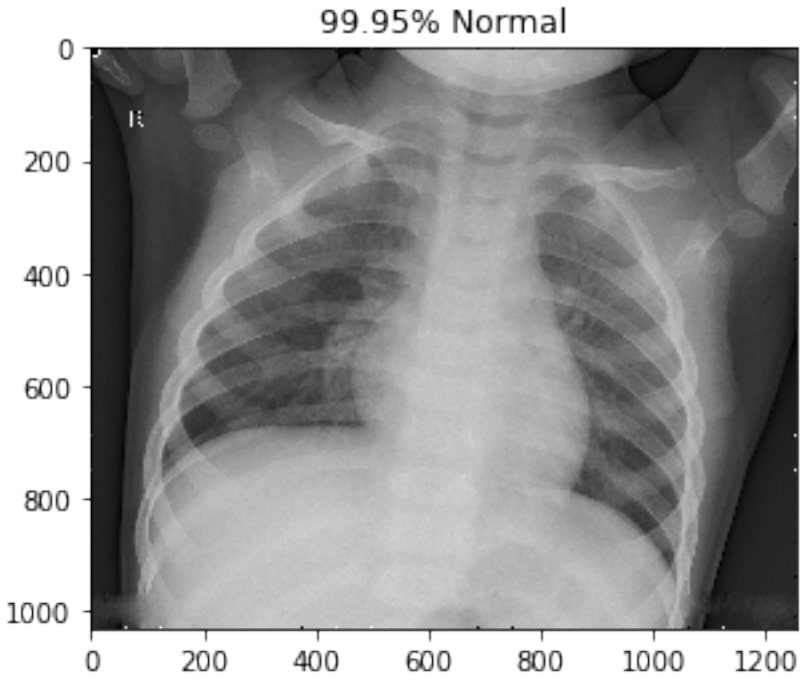
covid\ryct.2020200034.fig5-day7.jpeg



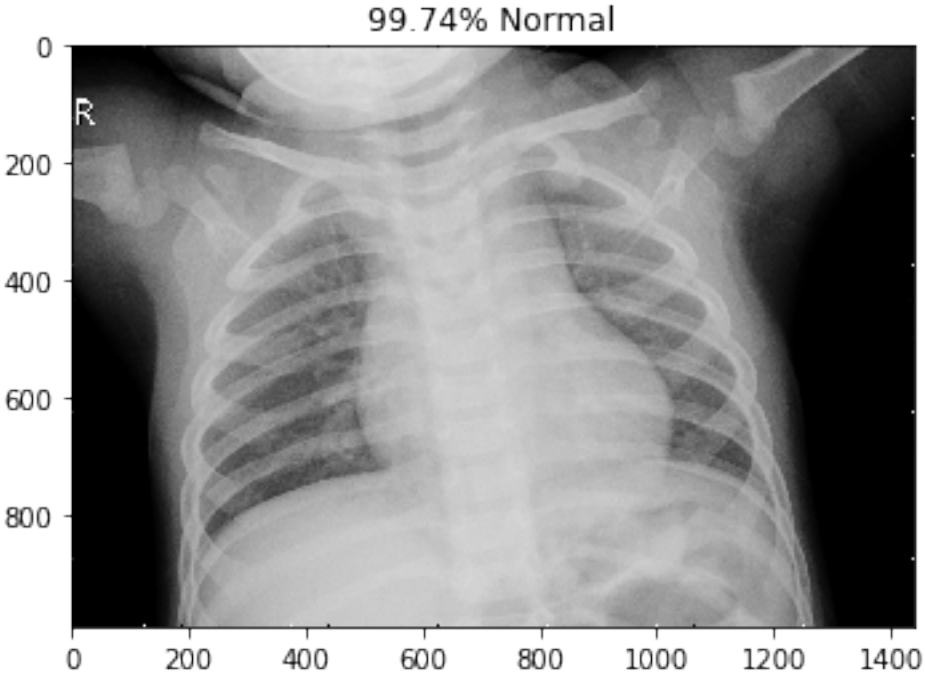
normal\NORMAL2-IM-1385-0001.jpeg



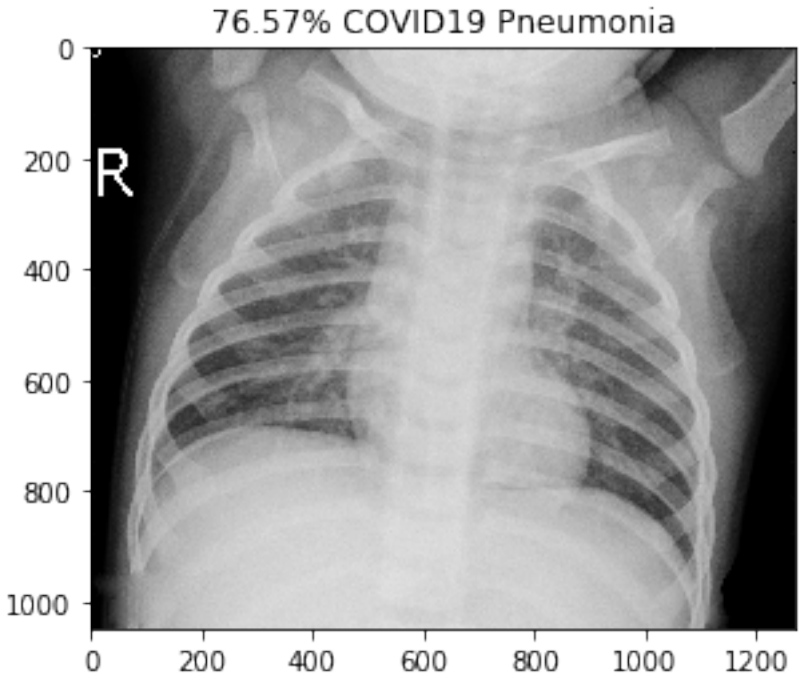
normal\NORMAL2-IM-1396-0001.jpeg



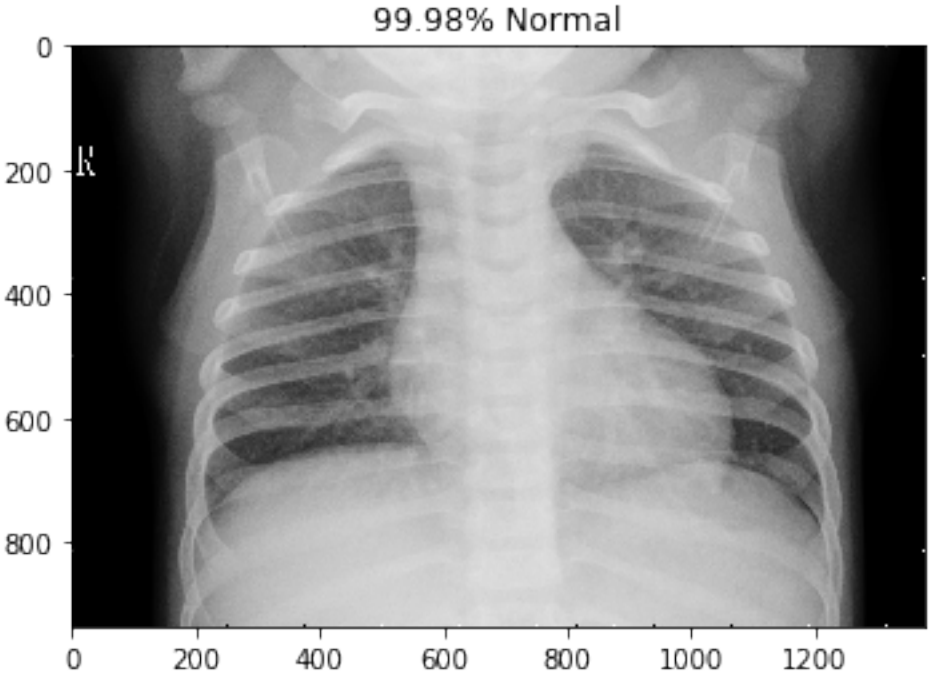
normal\NORMAL2-IM-1400-0001.jpeg



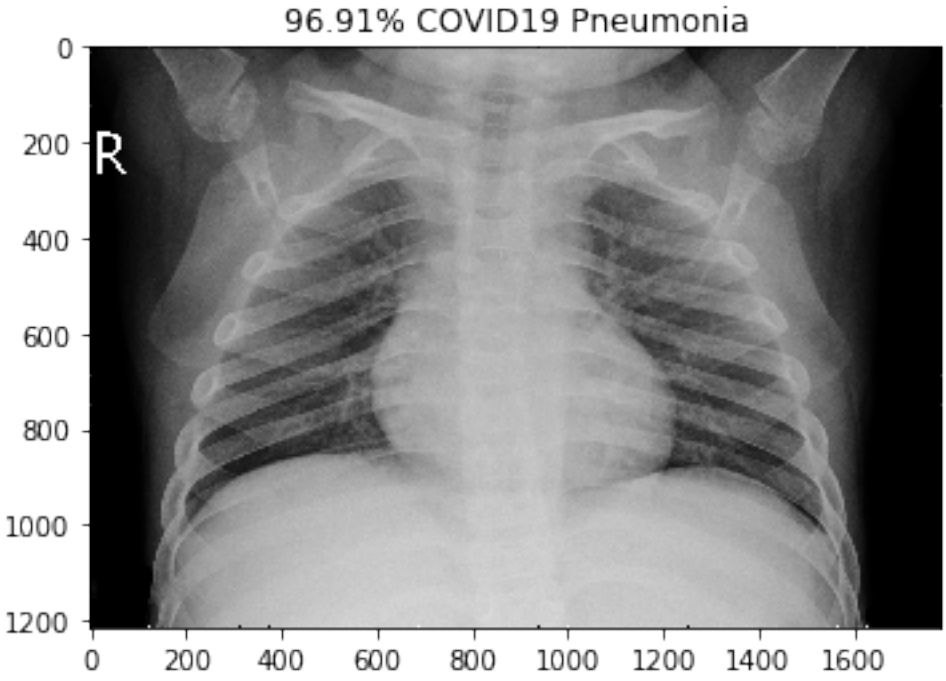
normal\NORMAL2-IM-1401-0001.jpeg



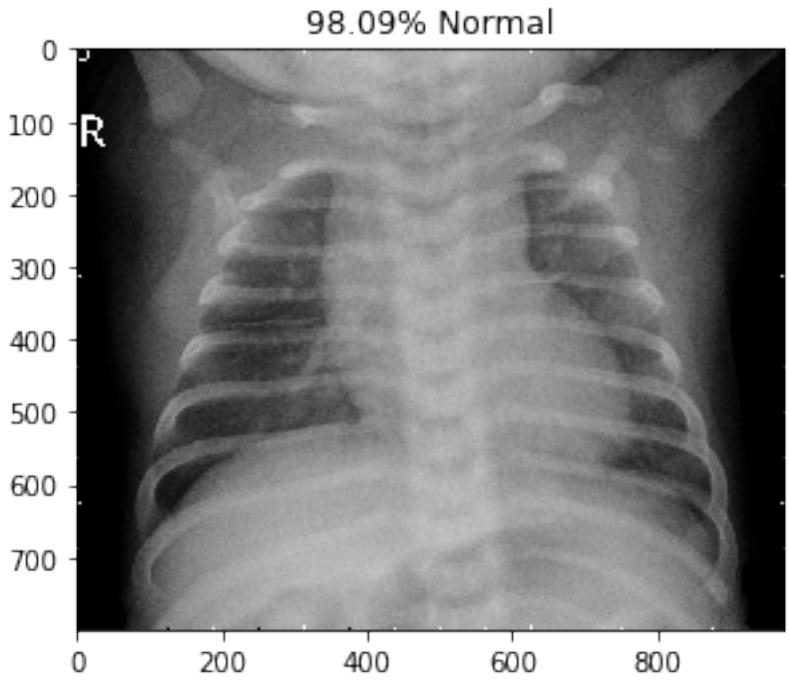
normal\NORMAL2-IM-1406-0001.jpeg



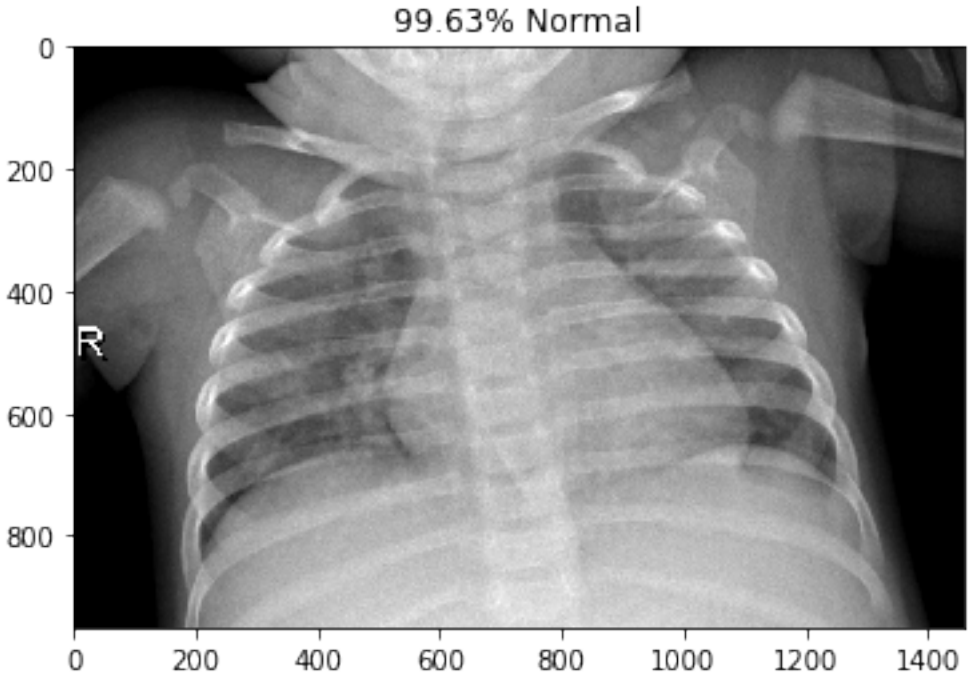
normal\NORMAL2-IM-1412-0001.jpeg



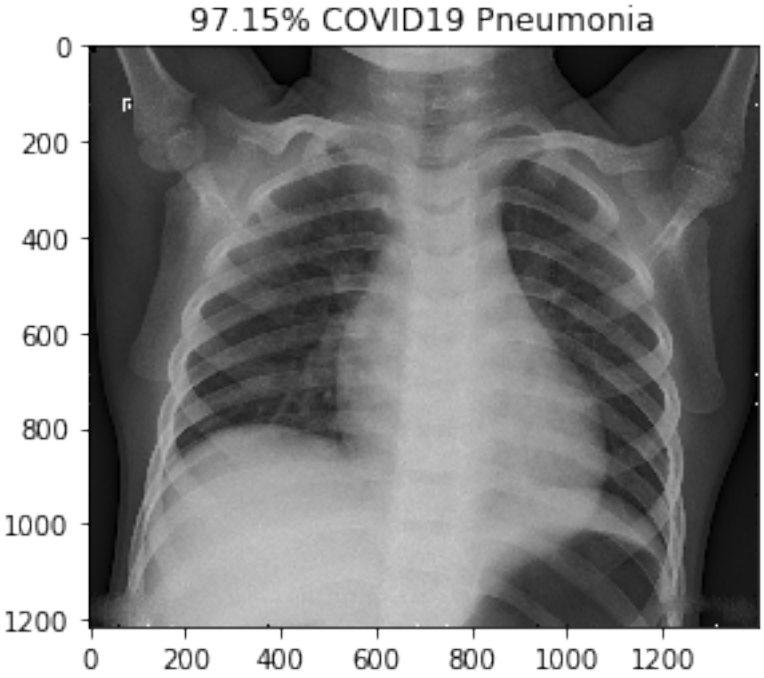
normal\NORMAL2-IM-1419-0001.jpeg



normal\NORMAL2-IM-1422-0001.jpeg



```
normal\NORMAL2-IM-1423-0001.jpeg
```



2.4 [10 points] TSNE Plot

t-Distributed Stochastic Neighbor Embedding (t-SNE) is a widely used technique for dimensionality reduction that is particularly well suited for the visualization of high-dimensional datasets. After training is complete, extract features from a specific deep layer of your choice, use t-SNE to reduce the dimensionality of your extracted features to 2 dimensions and plot the resulting 2D features.

[13]:
```
from sklearn.manifold import TSNE

intermediate_layer_model = models.Model(inputs=model.input,
                                        outputs=model.get_layer('dense_feature').
 ↪output)
tsne_data_generator = test_datagen.
 ↪flow_from_directory(DATASET_PATH,target_size=IMAGE_SIZE,
                                                    ␣
 ↪batch_size=1,shuffle=True,seed=42,class_mode="binary")

raise NotImplementedError("Extract features from the tsne_data_generator and fit␣
 ↪a t-SNE model for the features,"
                          "and plot the resulting 2D features of the two classes.
 ↪")
```

```
Found 130 images belonging to 2 classes.
130
```