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

Model = Xception

Task 2 [Total points: 30]

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
```

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.

[20 points] Multi-class Classification

```
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__
```

```
    '2.8.0'
```

```
from google.colab import drive

drive.mount('/content/gdrive')
```

```
    Drive already mounted at /content/gdrive; to attempt to forcibly remount, call d
```

Load Image Data

```
DATA_LIST = os.listdir('/content/gdrive/MyDrive/Colab Notebooks/Covid_Data_GradientCre
DATASET_PATH  = '/content/gdrive/MyDrive/Colab Notebooks/Covid_Data_GradientCrescent/a
TEST_DIR =  '/content/gdrive/MyDrive/Colab Notebooks/Covid_Data_GradientCrescent/all/t
IMAGE_SIZE    = (224, 224)
NUM_CLASSES   = len(DATA_LIST)
BATCH_SIZE    = 10  # try reducing batch size or freeze more layers if your GPU runs o
NUM_EPOCHS    = 15
LEARNING_RATE = 0.0001 # start off with high rate first 0.001 and experiment with redu
```

Generate Training and Validation Batches

```
train_datagen = ImageDataGenerator(rescale=1./255,rotation_range=50,featurewise_center
                                   featurewise_std_normalization = True,width_shift_ra
                                   height_shift_range=0.2,shear_range=0.25,zoom_range=
                                   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="categorical")

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="categorical")
```

```
Found 216 images belonging to 4 classes.
Found 54 images belonging to 4 classes.
/usr/local/lib/python3.7/dist-packages/keras_preprocessing/image/image_data_gene
  warnings.warn('This ImageDataGenerator specifies '
```

[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.

```
import numpy as np
from tensorflow.python.keras.layers.pooling import GlobalMaxPool2D
from tensorflow.python.keras.models import Sequential
from keras.layers import Dense, Flatten, GlobalAveragePooling2D, Dropout
from tensorflow.keras.applications.xception import Xception
from keras.models import Model
from keras.preprocessing import image
from keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.optimizers import Adam
import keras.backend as K
import matplotlib.pyplot as plt
from matplotlib.pyplot import imshow

base = Xception(include_top=False,
                weights='imagenet',
                input_shape=(224,224,3))
x = base.output
x = GlobalAveragePooling2D()(x)
x = Dense(50, activation = 'relu', name = 'dense_feature')(x)
x = Dropout(0.2)(x)
head = Dense(4, activation='softmax')(x)
model = Model(inputs=base.input, outputs=head)
```

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

```
/usr/local/lib/python3.7/dist-packages/keras/optimizer_v2/adam.py:105: UserWarnin
  super(Adam, self).__init__(name, **kwargs)
```

[5 points] Train Model

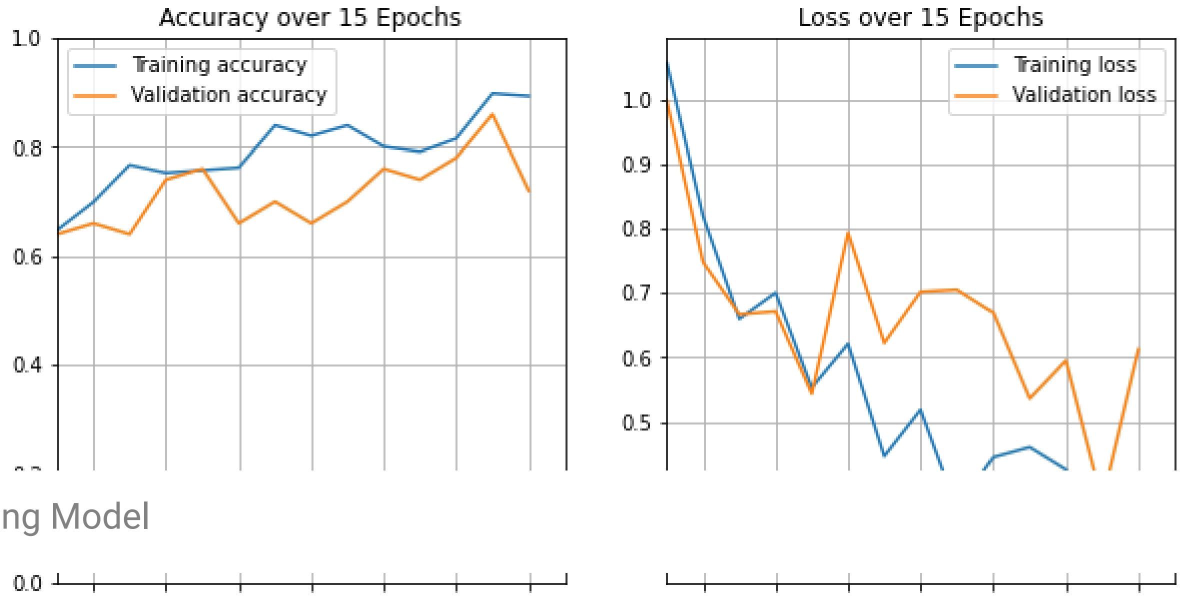
```
#FIT MODEL
print(len(train_batches))
print(len(valid_batches))

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

result=model.fit(train_batches,
                        steps_per_epoch =STEP_SIZE_TRAIN,
                        validation_data = valid_batches,
                        validation_steps = STEP_SIZE_VALID,
                        epochs=NUM_EPOCHS)

#Save model checkpoints, relabel graph
```

```
/usr/local/lib/python3.7/dist-packages/keras_preprocessing/image/image_data_gene
  warnings.warn('This ImageDataGenerator specifies '
/usr/local/lib/python3.7/dist-packages/keras_preprocessing/image/image_data_gene
  warnings.warn('This ImageDataGenerator specifies '
22
6
Epoch 1/15
21/21 [==============================] - 189s 9s/step - loss: 1.3189 - accuracy:
Epoch 2/15
21/21 [==============================] - 187s 9s/step - loss: 1.0570 - accuracy:
Epoch 3/15
21/21 [==============================] - 181s 9s/step - loss: 0.8193 - accuracy:
Epoch 4/15
21/21 [==============================] - 181s 9s/step - loss: 0.6599 - accuracy:
Epoch 5/15
21/21 [==============================] - 181s 9s/step - loss: 0.7004 - accuracy:
Epoch 6/15
21/21 [==============================] - 181s 9s/step - loss: 0.5532 - accuracy:
Epoch 7/15
21/21 [==============================] - 182s 9s/step - loss: 0.6213 - accuracy:
Epoch 8/15
21/21 [==============================] - 182s 9s/step - loss: 0.4468 - accuracy:
Epoch 9/15
21/21 [==============================] - 183s 9s/step - loss: 0.5188 - accuracy:
Epoch 10/15
21/21 [==============================] - 182s 9s/step - loss: 0.3656 - accuracy:
```

```
Epoch 11/15
21/21 [==============================] - 186s 9s/step - loss: 0.4452 - accuracy:
Epoch 12/15
21/21 [==============================] - 183s 9s/step - loss: 0.4608 - accuracy:
Epoch 13/15
21/21 [==============================] - 185s 9s/step - loss: 0.4260 - accuracy:
Epoch 14/15
21/21 [==============================] - 183s 9s/step - loss: 0.3202 - accuracy:
Epoch 15/15
21/21 [==============================] - 182s 9s/step - loss: 0.2895 - accuracy:
```

[5 points] Plot Accuracy and Loss During Training

```
import matplotlib.pyplot as plt

def p(result, epochs):
    acc = result.history['accuracy']
    loss = result.history['loss']
    val_acc = result.history['val_accuracy']
    val_loss = result.history['val_loss']
    title1 = 'Accuracy over ' + str(epochs) + ' Epochs'
    title2 = 'Loss over ' + str(epochs) + ' Epochs'
    plt.figure(figsize=(10, 5))

    plt.subplot(121)
    plt.grid(True)
    plt.ylim(0, 1)
    plt.xlim(1, epochs)
    plt.plot(range(1,epochs), acc[1:], label='Training accuracy')
    plt.plot(range(1,epochs), val_acc[1:], label='Validation accuracy')
    plt.title(title1)
    plt.legend()

    plt.subplot(122)
    plt.grid(True)
    plt.xlim(1, epochs)
    plt.plot(range(1,epochs), loss[1:], label='Training loss')
    plt.plot(range(1,epochs), val_loss[1:], label='Validation loss')
    plt.title(title2)
    plt.legend()
    plt.show()

p(result, NUM_EPOCHS)
```



Testing Model

```
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,cl
eval_generator.reset()
print(len(eval_generator))
x = model.evaluate_generator(eval_generator,steps = np.ceil(len(eval_generator)),
                           use_multiprocessing = False,verbose = 1,workers=1)
print('Test loss:' , x[0])
print('Test accuracy:',x[1])
```

```
Found 36 images belonging to 4 classes.
36
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:8: UserWarning: `Mo

36/36 [==============================] - 8s 233ms/step - loss: 0.6586 - accuracy
Test loss: 0.6585630774497986
Test accuracy: 0.7222222089767456
```

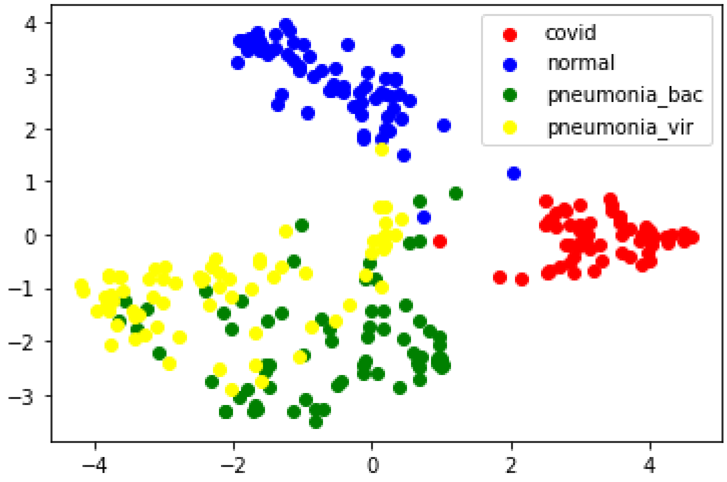
[10 points] TSNE Plot

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.

```
from sklearn.manifold import TSNE
from keras import models
```

```
intermediate_layer_model = models.Model(inputs=model.input,
                                        outputs=model.get_layer('dense_feature').outpu

tsne_eval_generator = test_datagen.flow_from_directory(DATASET_PATH,target_size=IMAGE_
                                                  batch_size=1,shuffle=False,seed=42,c

labels = tsne_eval_generator.classes

X = TSNE(learning_rate = .01).fit_transform(intermediate_layer_model.predict_generato

print(tsne_eval_generator.class_indices)
c = []
for key in tsne_eval_generator.class_indices:
  c.append(key)


group = X[np.where(labels == 0)]
plt.scatter(group[:, 0], group[:, 1], label = c[0], color='red')
group = X[np.where(labels == 1)]
plt.scatter(group[:, 0], group[:, 1], label = c[1], color='blue')
group = X[np.where(labels == 2)]
plt.scatter(group[:, 0], group[:, 1], label = c[2], color='green')
group = X[np.where(labels == 3)]
plt.scatter(group[:, 0], group[:, 1], label = c[3], color='yellow')
plt.legend()
```

```
Found 270 images belonging to 4 classes.
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:13: UserWarning: `Mo
  del sys.path[0]
270/270 [==============================] - 63s 229ms/step
/usr/local/lib/python3.7/dist-packages/sklearn/manifold/_t_sne.py:783: FutureWarn
  FutureWarning,
{'covid': 0, 'normal': 1, 'pneumonia_bac': 2, 'pneumonia_vir': 3}
<matplotlib.legend.Legend at 0x7fd5c7ecdb50>
```



```
model.summary()
```

```
    Model: "model_5"
    ________________________________________________________________________________
     Layer (type)                   Output Shape         Param #     Connected to
    ================================================================================
     input_4 (InputLayer)           [(None, 224, 224, 3  0           []
                                    )]

     block1_conv1 (Conv2D)          (None, 111, 111, 32  864         ['input_4[0]

     block1_conv1_bn (BatchNormaliz  (None, 111, 111, 32  128        ['block1_con
     ation)                         )

     block1_conv1_act (Activation)  (None, 111, 111, 32  0           ['block1_con
                                    )

     block1_conv2 (Conv2D)          (None, 109, 109, 64  18432       ['block1_con
                                    )

     block1_conv2_bn (BatchNormaliz  (None, 109, 109, 64  256        ['block1_con
     ation)                         )

     block1_conv2_act (Activation)  (None, 109, 109, 64  0           ['block1_con
                                    )

     block2_sepconv1 (SeparableConv  (None, 109, 109, 12  8768       ['block1_con
     2D)                            8)

     block2_sepconv1_bn (BatchNorma  (None, 109, 109, 12  512        ['block2_sep
     lization)                      8)

     block2_sepconv2_act (Activatio  (None, 109, 109, 12  0          ['block2_sep
     n)                             8)

     block2_sepconv2 (SeparableConv  (None, 109, 109, 12  17536      ['block2_sep
     2D)                            8)

     block2_sepconv2_bn (BatchNorma  (None, 109, 109, 12  512        ['block2_sep
     lization)                      8)

     conv2d_12 (Conv2D)             (None, 55, 55, 128)  8192        ['block1_con

     block2_pool (MaxPooling2D)     (None, 55, 55, 128)  0           ['block2_sep

     batch_normalization_12 (BatchN  (None, 55, 55, 128)  512        ['conv2d_12[
     ormalization)

     add_36 (Add)                   (None, 55, 55, 128)  0           ['block2_poo
                                                                      'batch_norm

     block3_sepconv1_act (Activatio  (None, 55, 55, 128)  0          ['add_36[0][
     n)
```

```
block3_sepconv1 (SeparableConv  (None, 55, 55, 256)  33920      ['block3_sep
2D)
```