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

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

In [4]:
```
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__
```

Out[4]:
```
'2.4.1'
```

Load Image Data

In [45]:
```
DATA_LIST = os.listdir('all/train')
DATASET_PATH  = 'all/train'
TEST_DIR =  'all/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    = 100
LEARNING_RATE = 0.0005 # start off with high rate first 0.001 and experiment with reducing it gradually
```

Generate Training and Validation Batches

In [46]:
```
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="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.
/home/shawn/.local/lib/python3.8/site-packages/keras_preprocessing/image/image_data_generator.py:342: UserWarning: T
his ImageDataGenerator specifies `zca_whitening` which overrides setting of`featurewise_std_normalization`.
  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.

In [57]:
```
#raise NotImplementedError("Build your model based on an architecture of your choice "
#                           "A sample model summary is shown below")

from tensorflow.keras.applications.vgg16 import VGG16

model = tf.keras.Sequential()
vgg16 = VGG16(input_shape=(224,224,3),include_top=False, weights='imagenet')

# Make vgg16 untrainable as per specification
vgg16.trainable=False
model.add(vgg16)
model.add(tf.keras.layers.Flatten())
model.add(tf.keras.layers.Dense(256,name='feature_dense',activation='relu'))
model.add(tf.keras.layers.Dense(4,activation='sigmoid'))
model.summary()
```

```
Model: "sequential_6"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
vgg16 (Functional)           (None, 7, 7, 512)         14714688
_________________________________________________________________
flatten_6 (Flatten)          (None, 25088)             0
_________________________________________________________________
feature_dense (Dense)        (None, 256)               6422784
_________________________________________________________________
dense_6 (Dense)              (None, 4)                 1028
=================================================================
Total params: 21,138,500
Trainable params: 6,423,812
Non-trainable params: 14,714,688
_________________________________________________________________
```

[5 points] Train Model

In [48]:
```
#FIT MODEL
import time
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
opt = tf.keras.optimizers.Adam(learning_rate=LEARNING_RATE)
model.compile(loss="categorical_crossentropy", optimizer=opt, metrics=["accuracy"])
t0 = time.time()
history = model.fit(
    train_batches,
    batch_size=BATCH_SIZE,
    epochs=NUM_EPOCHS,
    validation_data=valid_batches
)
print('Training took' + str(time.time()-t0))
# raise NotImplementedError("Use the model.fit function to train your network")
```

```
22
6
Epoch 1/100
22/22 [==============================] - 12s 542ms/step - loss: 2.8617 - accuracy: 0.2432 - val_loss: 1.4774 - val_a
ccuracy: 0.4259
Epoch 2/100
22/22 [==============================] - 12s 529ms/step - loss: 1.4259 - accuracy: 0.3660 - val_loss: 1.1741 - val_a
```

```
ccuracy: 0.3889
Epoch 3/100
22/22 [==============================] - 12s 532ms/step - loss: 1.0977 - accuracy: 0.5230 - val_loss: 1.0145 - val_a
ccuracy: 0.5556
Epoch 4/100
22/22 [==============================] - 12s 529ms/step - loss: 1.1458 - accuracy: 0.4660 - val_loss: 1.2601 - val_a
ccuracy: 0.5000
Epoch 5/100
22/22 [==============================] - 12s 534ms/step - loss: 1.1690 - accuracy: 0.4993 - val_loss: 1.1315 - val_a
ccuracy: 0.5370
Epoch 6/100
22/22 [==============================] - 12s 541ms/step - loss: 1.0416 - accuracy: 0.5381 - val_loss: 0.9387 - val_a
ccuracy: 0.6111
Epoch 7/100
22/22 [==============================] - 12s 528ms/step - loss: 0.9401 - accuracy: 0.5932 - val_loss: 1.1941 - val_a
ccuracy: 0.5370
Epoch 8/100
22/22 [==============================] - 12s 548ms/step - loss: 0.9865 - accuracy: 0.5450 - val_loss: 0.9219 - val_a
ccuracy: 0.5926
Epoch 9/100
22/22 [==============================] - 12s 526ms/step - loss: 0.8717 - accuracy: 0.6755 - val_loss: 0.9041 - val_a
ccuracy: 0.5926
Epoch 10/100
22/22 [==============================] - 11s 522ms/step - loss: 0.8594 - accuracy: 0.6350 - val_loss: 0.8053 - val_a
ccuracy: 0.6667
Epoch 11/100
22/22 [==============================] - 11s 521ms/step - loss: 0.9001 - accuracy: 0.5644 - val_loss: 0.7870 - val_a
ccuracy: 0.5926
Epoch 12/100
22/22 [==============================] - 12s 525ms/step - loss: 0.8094 - accuracy: 0.6486 - val_loss: 0.7537 - val_a
ccuracy: 0.6852
Epoch 13/100
22/22 [==============================] - 12s 523ms/step - loss: 0.9174 - accuracy: 0.6271 - val_loss: 0.7035 - val_a
ccuracy: 0.6481
Epoch 14/100
22/22 [==============================] - 11s 521ms/step - loss: 0.8301 - accuracy: 0.6799 - val_loss: 0.7553 - val_a
ccuracy: 0.6481
Epoch 15/100
22/22 [==============================] - 11s 522ms/step - loss: 0.7860 - accuracy: 0.6913 - val_loss: 0.7450 - val_a
ccuracy: 0.7037
Epoch 16/100
22/22 [==============================] - 11s 519ms/step - loss: 0.7014 - accuracy: 0.7468 - val_loss: 0.8259 - val_a
ccuracy: 0.6481
Epoch 17/100
22/22 [==============================] - 11s 522ms/step - loss: 0.7981 - accuracy: 0.6711 - val_loss: 0.6919 - val_a
ccuracy: 0.6667
Epoch 18/100
22/22 [==============================] - 11s 522ms/step - loss: 0.7573 - accuracy: 0.7079 - val_loss: 0.6180 - val_a
ccuracy: 0.7963
Epoch 19/100
22/22 [==============================] - 11s 519ms/step - loss: 0.6335 - accuracy: 0.7303 - val_loss: 0.7856 - val_a
ccuracy: 0.6481
Epoch 20/100
22/22 [==============================] - 12s 526ms/step - loss: 0.5745 - accuracy: 0.7627 - val_loss: 0.8318 - val_a
ccuracy: 0.6481
Epoch 21/100
22/22 [==============================] - 12s 534ms/step - loss: 0.6804 - accuracy: 0.6773 - val_loss: 0.7833 - val_a
ccuracy: 0.6852
Epoch 22/100
22/22 [==============================] - 12s 531ms/step - loss: 0.7515 - accuracy: 0.6902 - val_loss: 0.8319 - val_a
ccuracy: 0.6667
Epoch 23/100
22/22 [==============================] - 12s 523ms/step - loss: 0.6006 - accuracy: 0.7690 - val_loss: 0.7025 - val_a
ccuracy: 0.6667
Epoch 24/100
22/22 [==============================] - 12s 523ms/step - loss: 0.6541 - accuracy: 0.7104 - val_loss: 0.7501 - val_a
ccuracy: 0.6111
Epoch 25/100
22/22 [==============================] - 12s 523ms/step - loss: 0.7063 - accuracy: 0.7138 - val_loss: 0.6667 - val_a
ccuracy: 0.6667
Epoch 26/100
22/22 [==============================] - 12s 529ms/step - loss: 0.5885 - accuracy: 0.7486 - val_loss: 0.8790 - val_a
ccuracy: 0.5926
Epoch 27/100
22/22 [==============================] - 12s 525ms/step - loss: 0.6767 - accuracy: 0.7492 - val_loss: 0.7788 - val_a
ccuracy: 0.5926
Epoch 28/100
22/22 [==============================] - 11s 521ms/step - loss: 0.6461 - accuracy: 0.7256 - val_loss: 1.2383 - val_a
ccuracy: 0.5000
Epoch 29/100
22/22 [==============================] - 12s 522ms/step - loss: 0.7389 - accuracy: 0.6687 - val_loss: 0.5712 - val_a
ccuracy: 0.7222
Epoch 30/100
22/22 [==============================] - 12s 522ms/step - loss: 0.5538 - accuracy: 0.7672 - val_loss: 0.7505 - val_a
ccuracy: 0.6111
Epoch 31/100
22/22 [==============================] - 12s 520ms/step - loss: 0.5441 - accuracy: 0.7912 - val_loss: 0.7287 - val_a
ccuracy: 0.6667
Epoch 32/100
22/22 [==============================] - 12s 523ms/step - loss: 0.5587 - accuracy: 0.7738 - val_loss: 0.7072 - val_a
ccuracy: 0.6852
Epoch 33/100
22/22 [==============================] - 12s 524ms/step - loss: 0.6134 - accuracy: 0.7261 - val_loss: 0.6366 - val_a
```

```
ccuracy: 0.6852
Epoch 34/100
22/22 [==============================] - 12s 523ms/step - loss: 0.6809 - accuracy: 0.7119 - val_loss: 0.8339 - val_a
ccuracy: 0.5741
Epoch 35/100
22/22 [==============================] - 12s 537ms/step - loss: 0.5442 - accuracy: 0.7773 - val_loss: 0.6955 - val_a
ccuracy: 0.6481
Epoch 36/100
22/22 [==============================] - 12s 524ms/step - loss: 0.6540 - accuracy: 0.7027 - val_loss: 0.6905 - val_a
ccuracy: 0.7222
Epoch 37/100
22/22 [==============================] - 11s 520ms/step - loss: 0.5027 - accuracy: 0.8411 - val_loss: 0.6638 - val_a
ccuracy: 0.7407
Epoch 38/100
22/22 [==============================] - 12s 523ms/step - loss: 0.5456 - accuracy: 0.7531 - val_loss: 0.6703 - val_a
ccuracy: 0.7222
Epoch 39/100
22/22 [==============================] - 12s 525ms/step - loss: 0.6372 - accuracy: 0.7223 - val_loss: 0.7589 - val_a
ccuracy: 0.5556
Epoch 40/100
22/22 [==============================] - 12s 531ms/step - loss: 0.5412 - accuracy: 0.7489 - val_loss: 0.6638 - val_a
ccuracy: 0.6667
Epoch 41/100
22/22 [==============================] - 12s 528ms/step - loss: 0.5146 - accuracy: 0.7534 - val_loss: 0.7654 - val_a
ccuracy: 0.6852
Epoch 42/100
22/22 [==============================] - 12s 525ms/step - loss: 0.5298 - accuracy: 0.7688 - val_loss: 0.7819 - val_a
ccuracy: 0.6852
Epoch 43/100
22/22 [==============================] - 11s 523ms/step - loss: 0.5521 - accuracy: 0.7700 - val_loss: 1.0254 - val_a
ccuracy: 0.5556
Epoch 44/100
22/22 [==============================] - 12s 539ms/step - loss: 0.5884 - accuracy: 0.7662 - val_loss: 0.6285 - val_a
ccuracy: 0.7222
Epoch 45/100
22/22 [==============================] - 12s 542ms/step - loss: 0.5819 - accuracy: 0.7807 - val_loss: 0.6038 - val_a
ccuracy: 0.7222
Epoch 46/100
22/22 [==============================] - 12s 550ms/step - loss: 0.5308 - accuracy: 0.7603 - val_loss: 0.5511 - val_a
ccuracy: 0.7222
Epoch 47/100
22/22 [==============================] - 12s 533ms/step - loss: 0.4277 - accuracy: 0.8143 - val_loss: 0.6982 - val_a
ccuracy: 0.7037
Epoch 48/100
22/22 [==============================] - 12s 527ms/step - loss: 0.6126 - accuracy: 0.7423 - val_loss: 0.5256 - val_a
ccuracy: 0.7593
Epoch 49/100
22/22 [==============================] - 12s 530ms/step - loss: 0.4756 - accuracy: 0.7853 - val_loss: 0.5283 - val_a
ccuracy: 0.7037
Epoch 50/100
22/22 [==============================] - 12s 525ms/step - loss: 0.5112 - accuracy: 0.8086 - val_loss: 0.5930 - val_a
ccuracy: 0.7037
Epoch 51/100
22/22 [==============================] - 12s 527ms/step - loss: 0.5495 - accuracy: 0.8357 - val_loss: 0.7008 - val_a
ccuracy: 0.6852
Epoch 52/100
22/22 [==============================] - 12s 526ms/step - loss: 0.5816 - accuracy: 0.7416 - val_loss: 0.7676 - val_a
ccuracy: 0.5926
Epoch 53/100
22/22 [==============================] - 12s 530ms/step - loss: 0.4458 - accuracy: 0.8176 - val_loss: 0.6005 - val_a
ccuracy: 0.7037
Epoch 54/100
22/22 [==============================] - 12s 533ms/step - loss: 0.4544 - accuracy: 0.7803 - val_loss: 0.6269 - val_a
ccuracy: 0.7037
Epoch 55/100
22/22 [==============================] - 12s 527ms/step - loss: 0.4987 - accuracy: 0.8043 - val_loss: 0.6694 - val_a
ccuracy: 0.7037
Epoch 56/100
22/22 [==============================] - 12s 527ms/step - loss: 0.4848 - accuracy: 0.8093 - val_loss: 0.6021 - val_a
ccuracy: 0.6852
Epoch 57/100
22/22 [==============================] - 12s 529ms/step - loss: 0.5017 - accuracy: 0.7551 - val_loss: 0.8078 - val_a
ccuracy: 0.6296
Epoch 58/100
22/22 [==============================] - 12s 532ms/step - loss: 0.4952 - accuracy: 0.7561 - val_loss: 0.9053 - val_a
ccuracy: 0.5926
Epoch 59/100
22/22 [==============================] - 12s 536ms/step - loss: 0.6030 - accuracy: 0.7159 - val_loss: 0.6707 - val_a
ccuracy: 0.6667
Epoch 60/100
22/22 [==============================] - 12s 530ms/step - loss: 0.5649 - accuracy: 0.7239 - val_loss: 0.6047 - val_a
ccuracy: 0.7037
Epoch 61/100
22/22 [==============================] - 12s 530ms/step - loss: 0.5348 - accuracy: 0.7660 - val_loss: 0.7349 - val_a
ccuracy: 0.6296
Epoch 62/100
22/22 [==============================] - 12s 532ms/step - loss: 0.4052 - accuracy: 0.8194 - val_loss: 0.7907 - val_a
ccuracy: 0.6481
Epoch 63/100
22/22 [==============================] - 12s 534ms/step - loss: 0.5388 - accuracy: 0.7523 - val_loss: 0.7878 - val_a
ccuracy: 0.6296
Epoch 64/100
22/22 [==============================] - 12s 532ms/step - loss: 0.4551 - accuracy: 0.7984 - val_loss: 0.6522 - val_a
```

```
ccuracy: 0.7037
Epoch 65/100
22/22 [==============================] - 12s 531ms/step - loss: 0.5509 - accuracy: 0.7420 - val_loss: 0.6622 - val_a
ccuracy: 0.6667
Epoch 66/100
22/22 [==============================] - 12s 527ms/step - loss: 0.5642 - accuracy: 0.7726 - val_loss: 0.7752 - val_a
ccuracy: 0.6111
Epoch 67/100
22/22 [==============================] - 12s 527ms/step - loss: 0.5178 - accuracy: 0.8071 - val_loss: 0.9146 - val_a
ccuracy: 0.5741
Epoch 68/100
22/22 [==============================] - 12s 527ms/step - loss: 0.5572 - accuracy: 0.7742 - val_loss: 0.7903 - val_a
ccuracy: 0.6667
Epoch 69/100
22/22 [==============================] - 12s 528ms/step - loss: 0.6158 - accuracy: 0.7383 - val_loss: 0.8410 - val_a
ccuracy: 0.6481
Epoch 70/100
22/22 [==============================] - 12s 527ms/step - loss: 0.5222 - accuracy: 0.8145 - val_loss: 0.6673 - val_a
ccuracy: 0.7037
Epoch 71/100
22/22 [==============================] - 12s 531ms/step - loss: 0.5504 - accuracy: 0.7587 - val_loss: 0.8170 - val_a
ccuracy: 0.6296
Epoch 72/100
22/22 [==============================] - 12s 531ms/step - loss: 0.5653 - accuracy: 0.7383 - val_loss: 0.8112 - val_a
ccuracy: 0.6481
Epoch 73/100
22/22 [==============================] - 12s 529ms/step - loss: 0.4551 - accuracy: 0.7770 - val_loss: 0.6097 - val_a
ccuracy: 0.7037
Epoch 74/100
22/22 [==============================] - 12s 531ms/step - loss: 0.5381 - accuracy: 0.7562 - val_loss: 0.6489 - val_a
ccuracy: 0.6667
Epoch 75/100
22/22 [==============================] - 12s 538ms/step - loss: 0.5498 - accuracy: 0.7673 - val_loss: 0.7481 - val_a
ccuracy: 0.7037
Epoch 76/100
22/22 [==============================] - 12s 533ms/step - loss: 0.4582 - accuracy: 0.7886 - val_loss: 0.7066 - val_a
ccuracy: 0.6481
Epoch 77/100
22/22 [==============================] - 12s 532ms/step - loss: 0.5235 - accuracy: 0.7886 - val_loss: 0.7084 - val_a
ccuracy: 0.6481
Epoch 78/100
22/22 [==============================] - 12s 532ms/step - loss: 0.4803 - accuracy: 0.7721 - val_loss: 0.7926 - val_a
ccuracy: 0.6481
Epoch 79/100
22/22 [==============================] - 12s 536ms/step - loss: 0.4805 - accuracy: 0.8106 - val_loss: 0.9831 - val_a
ccuracy: 0.5741
Epoch 80/100
22/22 [==============================] - 12s 543ms/step - loss: 0.5195 - accuracy: 0.8298 - val_loss: 0.6682 - val_a
ccuracy: 0.7037
Epoch 81/100
22/22 [==============================] - 12s 530ms/step - loss: 0.4525 - accuracy: 0.7980 - val_loss: 0.6648 - val_a
ccuracy: 0.7407
Epoch 82/100
22/22 [==============================] - 12s 538ms/step - loss: 0.4667 - accuracy: 0.7896 - val_loss: 1.0253 - val_a
ccuracy: 0.5926
Epoch 83/100
22/22 [==============================] - 12s 530ms/step - loss: 0.5058 - accuracy: 0.7872 - val_loss: 0.5967 - val_a
ccuracy: 0.7037
Epoch 84/100
22/22 [==============================] - 12s 542ms/step - loss: 0.5378 - accuracy: 0.7353 - val_loss: 0.6560 - val_a
ccuracy: 0.6667
Epoch 85/100
22/22 [==============================] - 12s 547ms/step - loss: 0.4387 - accuracy: 0.8233 - val_loss: 0.6259 - val_a
ccuracy: 0.7222
Epoch 86/100
22/22 [==============================] - 12s 548ms/step - loss: 0.4435 - accuracy: 0.8067 - val_loss: 0.6703 - val_a
ccuracy: 0.6667
Epoch 87/100
22/22 [==============================] - 12s 528ms/step - loss: 0.5309 - accuracy: 0.7509 - val_loss: 0.5874 - val_a
ccuracy: 0.7222
Epoch 88/100
22/22 [==============================] - 12s 543ms/step - loss: 0.3491 - accuracy: 0.8698 - val_loss: 0.8725 - val_a
ccuracy: 0.6667
Epoch 89/100
22/22 [==============================] - 12s 523ms/step - loss: 0.5168 - accuracy: 0.7485 - val_loss: 0.8530 - val_a
ccuracy: 0.6296
Epoch 90/100
22/22 [==============================] - 12s 530ms/step - loss: 0.5279 - accuracy: 0.7553 - val_loss: 0.6004 - val_a
ccuracy: 0.7037
Epoch 91/100
22/22 [==============================] - 12s 553ms/step - loss: 0.4331 - accuracy: 0.8270 - val_loss: 0.8196 - val_a
ccuracy: 0.6111
Epoch 92/100
22/22 [==============================] - 12s 531ms/step - loss: 0.4250 - accuracy: 0.8308 - val_loss: 0.6763 - val_a
ccuracy: 0.6111
Epoch 93/100
22/22 [==============================] - 11s 522ms/step - loss: 0.3348 - accuracy: 0.8964 - val_loss: 0.6926 - val_a
ccuracy: 0.6852
Epoch 94/100
22/22 [==============================] - 11s 523ms/step - loss: 0.4774 - accuracy: 0.7832 - val_loss: 0.9018 - val_a
ccuracy: 0.6852
Epoch 95/100
22/22 [==============================] - 12s 536ms/step - loss: 0.4291 - accuracy: 0.8146 - val_loss: 0.8662 - val_a
```

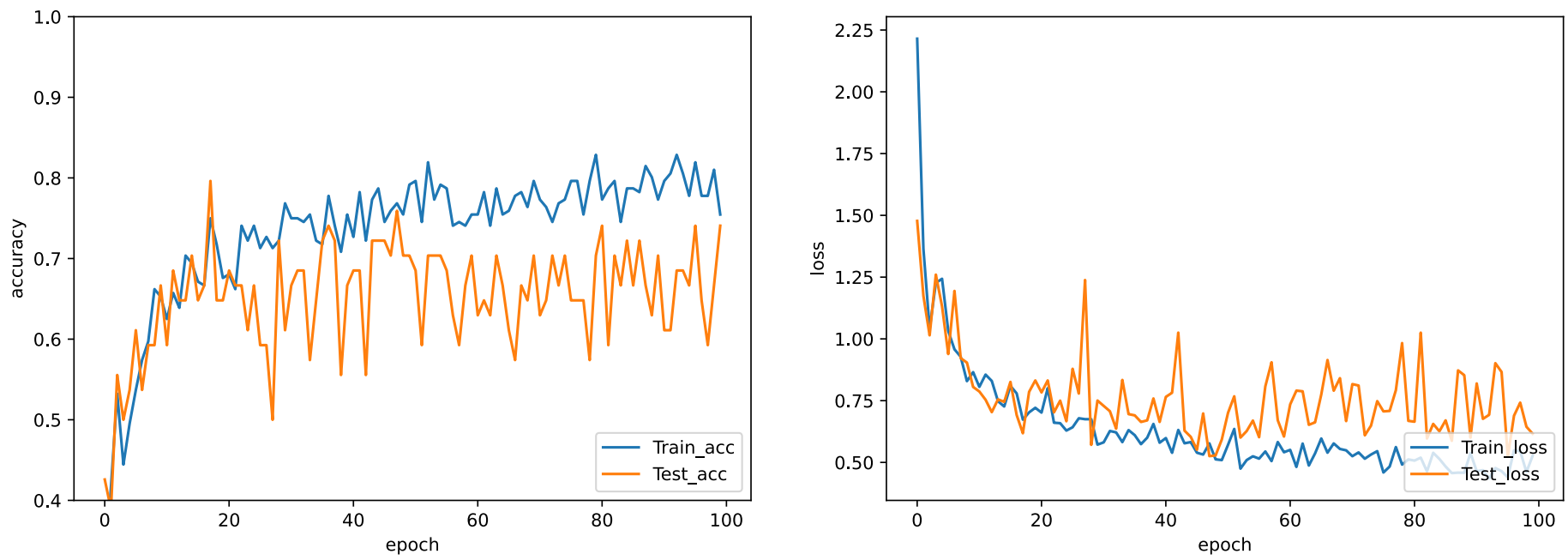
```
ccuracy: 0.6667
Epoch 96/100
22/22 [==============================] - 12s 530ms/step - loss: 0.4905 - accuracy: 0.7949 - val_loss: 0.5233 - val_a
ccuracy: 0.7407
Epoch 97/100
22/22 [==============================] - 12s 532ms/step - loss: 0.5153 - accuracy: 0.7860 - val_loss: 0.6891 - val_a
ccuracy: 0.6481
Epoch 98/100
22/22 [==============================] - 12s 536ms/step - loss: 0.4728 - accuracy: 0.8195 - val_loss: 0.7421 - val_a
ccuracy: 0.5926
Epoch 99/100
22/22 [==============================] - 12s 535ms/step - loss: 0.5287 - accuracy: 0.7662 - val_loss: 0.6445 - val_a
ccuracy: 0.6667
Epoch 100/100
22/22 [==============================] - 12s 523ms/step - loss: 0.4706 - accuracy: 0.7794 - val_loss: 0.6161 - val_a
ccuracy: 0.7407
```

[5 points] Plot Accuracy and Loss During Training

In [49]:

```
import matplotlib.pyplot as plt
plt.figure(figsize=(15,5))
plt.subplot(1,2,1)
plt.plot(history.history['accuracy'],label='Train_acc')
plt.plot(history.history['val_accuracy'],label='Test_acc')
plt.xlabel('epoch')
plt.ylabel('accuracy')
plt.ylim([0.4,1])
plt.legend(loc='lower right')

plt.subplot(1,2,2)
plt.plot(history.history['loss'],label='Train_loss')
plt.plot(history.history['val_loss'],label='Test_loss')
plt.xlabel('epoch')
plt.ylabel('loss')
plt.legend(loc='lower right')
```

Out[49]: <matplotlib.legend.Legend at 0x7f56ef707730>



Testing Model

In [51]:

```
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="categorical")
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])
```

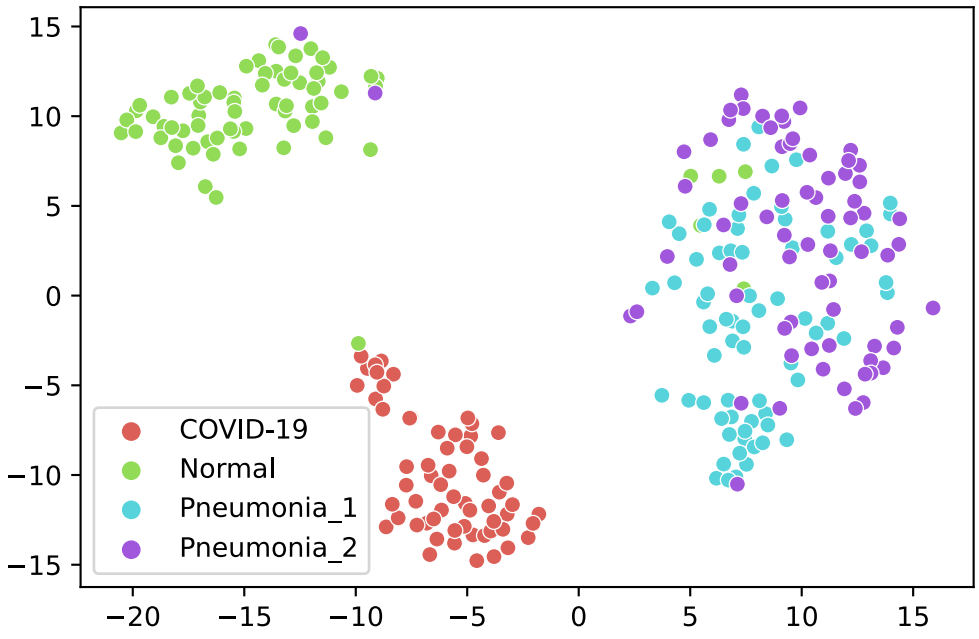
```
Found 36 images belonging to 4 classes.
36
36/36 [==============================] - 2s 46ms/step - loss: 0.7771 - accuracy: 0.6944
Test loss: 0.7770802974700928
Test accuracy: 0.6944444179534912
```

[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.

In [56]:
```
from sklearn.manifold import TSNE
import seaborn as sns
intermediate_layer_model = tf.keras.models.Model(inputs=model.input,
                                         outputs=model.get_layer('feature_dense').output)

tsne_eval_generator = test_datagen.flow_from_directory(DATASET_PATH,target_size=IMAGE_SIZE,
                                                       batch_size=1,shuffle=False,seed=42,class_mode="categorical")

intermediate_output = intermediate_layer_model.predict(tsne_eval_generator)
tsne = TSNE(n_components=2)
tsne_data = tsne.fit_transform(intermediate_output)
labels = map(lambda x: "COVID-19" if x == 0 else ("Normal" if x==1 else ("Pneumonia_1" if x==2 else "Pneumonia_2"))

scat=sns.scatterplot(tsne_data[:,0],tsne_data[:,1], hue = labels, palette=sns.color_palette("hls", 4))

plt.show()

# 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 four classes.")
```

```
Found 270 images belonging to 4 classes.
/home/shawn/.local/lib/python3.8/site-packages/seaborn/_decorators.py:36: FutureWarning: Pass the following variable
s as keyword args: x, y. From version 0.12, the only valid positional argument will be `data`, and passing other arg
uments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```



Second model

I will be using ResNet101 as my second model

In [73]:
```
base_model = tf.keras.applications.ResNet101(
    input_shape=(224,224,3),
    include_top=False,
    weights='imagenet'
)
base_model.trainable=False
n_model = model = tf.keras.Sequential()

# Make vgg16 untrainable as per specification
n_model.add(base_model)
n_model.add(tf.keras.layers.Flatten())
n_model.add(tf.keras.layers.Dense(256,name='feature_dense',activation='relu'))
n_model.add(tf.keras.layers.Dense(4,activation='sigmoid'))
n_model.summary()
```

```
Model: "sequential_15"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
resnet101 (Functional)       (None, 7, 7, 2048)        42658176
_________________________________________________________________
flatten_15 (Flatten)         (None, 100352)            0
_________________________________________________________________
feature_dense (Dense)        (None, 256)               25690368
_________________________________________________________________
dense_15 (Dense)             (None, 4)                 1028
=================================================================
Total params: 68,349,572
Trainable params: 25,691,396
Non-trainable params: 42,658,176
_________________________________________________________________
```

In [74]:

```
import time
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
opt = tf.keras.optimizers.Adam(learning_rate=0.001)
model.compile(loss="categorical_crossentropy", optimizer=opt, metrics=["accuracy"])
t0 = time.time()
history = n_model.fit(
    train_batches,
    batch_size=BATCH_SIZE,
    epochs=NUM_EPOCHS,
    validation_data=valid_batches
)
print('Training took' + str(time.time()-t0))
```

```
22
6
Epoch 1/100
22/22 [==============================] - 14s 507ms/step - loss: 8.5160 - accuracy: 0.2591 - val_loss: 1.6539 - val_a
ccuracy: 0.3333
Epoch 2/100
22/22 [==============================] - 10s 456ms/step - loss: 2.1214 - accuracy: 0.3533 - val_loss: 1.9378 - val_a
ccuracy: 0.2593
Epoch 3/100
22/22 [==============================] - 10s 458ms/step - loss: 1.8054 - accuracy: 0.3940 - val_loss: 1.7311 - val_a
ccuracy: 0.3519
Epoch 4/100
22/22 [==============================] - 10s 458ms/step - loss: 2.0533 - accuracy: 0.3889 - val_loss: 1.7701 - val_a
ccuracy: 0.3148
Epoch 5/100
22/22 [==============================] - 10s 470ms/step - loss: 1.7996 - accuracy: 0.3937 - val_loss: 3.3099 - val_a
ccuracy: 0.2963
Epoch 6/100
22/22 [==============================] - 10s 469ms/step - loss: 2.2712 - accuracy: 0.3983 - val_loss: 1.3860 - val_a
ccuracy: 0.3519
Epoch 7/100
22/22 [==============================] - 10s 467ms/step - loss: 1.7378 - accuracy: 0.3830 - val_loss: 1.4656 - val_a
ccuracy: 0.4815
Epoch 8/100
22/22 [==============================] - 10s 472ms/step - loss: 1.6495 - accuracy: 0.3785 - val_loss: 2.4397 - val_a
ccuracy: 0.3889
Epoch 9/100
22/22 [==============================] - 10s 472ms/step - loss: 1.8613 - accuracy: 0.4280 - val_loss: 1.8781 - val_a
ccuracy: 0.4630
Epoch 10/100
22/22 [==============================] - 10s 468ms/step - loss: 2.0786 - accuracy: 0.4135 - val_loss: 2.2863 - val_a
ccuracy: 0.5185
Epoch 11/100
22/22 [==============================] - 10s 465ms/step - loss: 2.2017 - accuracy: 0.4158 - val_loss: 1.4083 - val_a
ccuracy: 0.2593
Epoch 12/100
22/22 [==============================] - 10s 464ms/step - loss: 1.5242 - accuracy: 0.4674 - val_loss: 1.4696 - val_a
ccuracy: 0.4444
Epoch 13/100
22/22 [==============================] - 10s 469ms/step - loss: 2.0111 - accuracy: 0.3211 - val_loss: 1.6805 - val_a
ccuracy: 0.3148
Epoch 14/100
22/22 [==============================] - 10s 462ms/step - loss: 1.3236 - accuracy: 0.4507 - val_loss: 1.4076 - val_a
ccuracy: 0.4074
Epoch 15/100
22/22 [==============================] - 10s 468ms/step - loss: 1.6142 - accuracy: 0.3463 - val_loss: 1.2030 - val_a
ccuracy: 0.3704
Epoch 16/100
22/22 [==============================] - 10s 459ms/step - loss: 1.3452 - accuracy: 0.4542 - val_loss: 1.2067 - val_a
ccuracy: 0.4074
Epoch 17/100
22/22 [==============================] - 10s 458ms/step - loss: 1.3006 - accuracy: 0.3775 - val_loss: 1.5283 - val_a
ccuracy: 0.3889
Epoch 18/100
22/22 [==============================] - 10s 459ms/step - loss: 1.1496 - accuracy: 0.5252 - val_loss: 1.3914 - val_a
ccuracy: 0.4444
Epoch 19/100
22/22 [==============================] - 10s 457ms/step - loss: 1.2184 - accuracy: 0.4540 - val_loss: 1.1161 - val_a
ccuracy: 0.4815
Epoch 20/100
22/22 [==============================] - 10s 457ms/step - loss: 1.3479 - accuracy: 0.4243 - val_loss: 1.3626 - val_a
ccuracy: 0.4074
Epoch 21/100
22/22 [==============================] - 10s 456ms/step - loss: 1.1424 - accuracy: 0.5120 - val_loss: 1.1030 - val_a
ccuracy: 0.5185
Epoch 22/100
22/22 [==============================] - 10s 458ms/step - loss: 1.2306 - accuracy: 0.4212 - val_loss: 1.2503 - val_a
ccuracy: 0.4815
Epoch 23/100
22/22 [==============================] - 10s 465ms/step - loss: 1.2676 - accuracy: 0.4713 - val_loss: 1.1430 - val_a
ccuracy: 0.4259
Epoch 24/100
22/22 [==============================] - 10s 463ms/step - loss: 1.2759 - accuracy: 0.4081 - val_loss: 1.3309 - val_a
ccuracy: 0.5000
```

```
Epoch 25/100
22/22 [==============================] - 11s 485ms/step - loss: 1.1829 - accuracy: 0.4320 - val_loss: 1.1658 - val_a
ccuracy: 0.5185
Epoch 26/100
22/22 [==============================] - 10s 472ms/step - loss: 1.2705 - accuracy: 0.4558 - val_loss: 1.3093 - val_a
ccuracy: 0.4259
Epoch 27/100
22/22 [==============================] - 11s 478ms/step - loss: 1.3346 - accuracy: 0.3707 - val_loss: 1.3205 - val_a
ccuracy: 0.4630
Epoch 28/100
22/22 [==============================] - 10s 471ms/step - loss: 1.2803 - accuracy: 0.4864 - val_loss: 1.2476 - val_a
ccuracy: 0.3889
Epoch 29/100
22/22 [==============================] - 10s 450ms/step - loss: 1.2673 - accuracy: 0.4311 - val_loss: 1.2925 - val_a
ccuracy: 0.3333
Epoch 30/100
22/22 [==============================] - 10s 448ms/step - loss: 1.3315 - accuracy: 0.4680 - val_loss: 1.2276 - val_a
ccuracy: 0.4630
Epoch 31/100
22/22 [==============================] - 10s 464ms/step - loss: 1.2284 - accuracy: 0.4116 - val_loss: 1.0486 - val_a
ccuracy: 0.5741
Epoch 32/100
22/22 [==============================] - 10s 468ms/step - loss: 1.3133 - accuracy: 0.3917 - val_loss: 1.1347 - val_a
ccuracy: 0.5370
Epoch 33/100
22/22 [==============================] - 10s 474ms/step - loss: 1.0932 - accuracy: 0.5010 - val_loss: 1.3574 - val_a
ccuracy: 0.4259
Epoch 34/100
22/22 [==============================] - 10s 469ms/step - loss: 1.1646 - accuracy: 0.4303 - val_loss: 1.1035 - val_a
ccuracy: 0.4444
Epoch 35/100
22/22 [==============================] - 10s 468ms/step - loss: 1.0742 - accuracy: 0.5150 - val_loss: 1.2258 - val_a
ccuracy: 0.3889
Epoch 36/100
22/22 [==============================] - 10s 450ms/step - loss: 1.2027 - accuracy: 0.4517 - val_loss: 1.0483 - val_a
ccuracy: 0.5185
Epoch 37/100
22/22 [==============================] - 10s 456ms/step - loss: 1.3365 - accuracy: 0.4139 - val_loss: 1.1309 - val_a
ccuracy: 0.4630
Epoch 38/100
22/22 [==============================] - 10s 451ms/step - loss: 1.2242 - accuracy: 0.4661 - val_loss: 1.1060 - val_a
ccuracy: 0.5185
Epoch 39/100
22/22 [==============================] - 10s 447ms/step - loss: 1.1665 - accuracy: 0.4616 - val_loss: 1.0520 - val_a
ccuracy: 0.4815
Epoch 40/100
22/22 [==============================] - 10s 445ms/step - loss: 1.1093 - accuracy: 0.4951 - val_loss: 1.1080 - val_a
ccuracy: 0.4815
Epoch 41/100
22/22 [==============================] - 10s 448ms/step - loss: 1.2128 - accuracy: 0.4043 - val_loss: 1.0722 - val_a
ccuracy: 0.4815
Epoch 42/100
22/22 [==============================] - 10s 449ms/step - loss: 1.1621 - accuracy: 0.4396 - val_loss: 1.0801 - val_a
ccuracy: 0.4259
Epoch 43/100
22/22 [==============================] - 10s 453ms/step - loss: 1.1418 - accuracy: 0.4970 - val_loss: 1.0052 - val_a
ccuracy: 0.5556
Epoch 44/100
22/22 [==============================] - 10s 453ms/step - loss: 1.1088 - accuracy: 0.4866 - val_loss: 1.0318 - val_a
ccuracy: 0.5000
Epoch 45/100
22/22 [==============================] - 10s 447ms/step - loss: 1.1672 - accuracy: 0.4828 - val_loss: 1.0418 - val_a
ccuracy: 0.5556
Epoch 46/100
22/22 [==============================] - 10s 447ms/step - loss: 1.1214 - accuracy: 0.4920 - val_loss: 1.0247 - val_a
ccuracy: 0.5370
Epoch 47/100
22/22 [==============================] - 10s 454ms/step - loss: 1.1248 - accuracy: 0.5045 - val_loss: 1.1069 - val_a
ccuracy: 0.4630
Epoch 48/100
22/22 [==============================] - 11s 497ms/step - loss: 1.1343 - accuracy: 0.4930 - val_loss: 0.9986 - val_a
ccuracy: 0.4815
Epoch 49/100
22/22 [==============================] - 10s 475ms/step - loss: 1.1429 - accuracy: 0.4902 - val_loss: 1.1517 - val_a
ccuracy: 0.3889
Epoch 50/100
22/22 [==============================] - 10s 467ms/step - loss: 1.1735 - accuracy: 0.4444 - val_loss: 1.0163 - val_a
ccuracy: 0.4815
Epoch 51/100
22/22 [==============================] - 11s 481ms/step - loss: 1.0649 - accuracy: 0.5799 - val_loss: 1.2082 - val_a
ccuracy: 0.4630
Epoch 52/100
22/22 [==============================] - 11s 479ms/step - loss: 1.1811 - accuracy: 0.4827 - val_loss: 1.3090 - val_a
ccuracy: 0.4444
Epoch 53/100
22/22 [==============================] - 11s 482ms/step - loss: 1.1385 - accuracy: 0.5471 - val_loss: 0.9381 - val_a
ccuracy: 0.5185
Epoch 54/100
22/22 [==============================] - 10s 468ms/step - loss: 1.0307 - accuracy: 0.5476 - val_loss: 1.0461 - val_a
ccuracy: 0.5185
Epoch 55/100
22/22 [==============================] - 10s 462ms/step - loss: 1.1791 - accuracy: 0.4903 - val_loss: 1.0699 - val_a
ccuracy: 0.4074
```

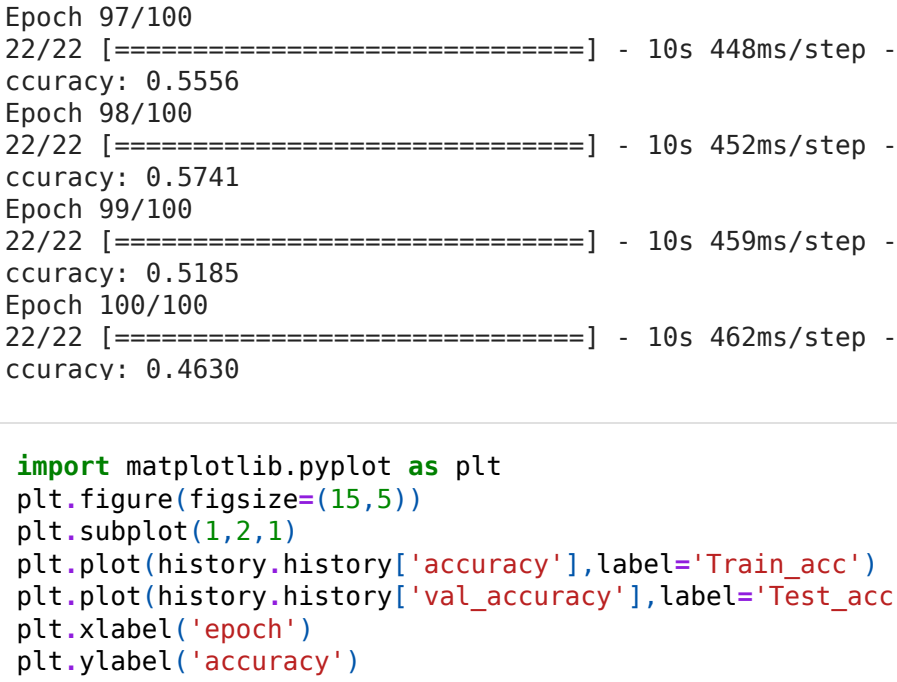
```
Epoch 56/100
22/22 [==============================] - 10s 467ms/step - loss: 1.0392 - accuracy: 0.5159 - val_loss: 1.1287 - val_a
ccuracy: 0.5185
Epoch 57/100
22/22 [==============================] - 10s 475ms/step - loss: 1.1394 - accuracy: 0.4516 - val_loss: 1.0713 - val_a
ccuracy: 0.4259
Epoch 58/100
22/22 [==============================] - 11s 485ms/step - loss: 1.1300 - accuracy: 0.4643 - val_loss: 1.1131 - val_a
ccuracy: 0.4630
Epoch 59/100
22/22 [==============================] - 10s 474ms/step - loss: 1.0173 - accuracy: 0.5614 - val_loss: 1.0020 - val_a
ccuracy: 0.4815
Epoch 60/100
22/22 [==============================] - 10s 459ms/step - loss: 1.1208 - accuracy: 0.5224 - val_loss: 0.9983 - val_a
ccuracy: 0.5556
Epoch 61/100
22/22 [==============================] - 11s 486ms/step - loss: 1.0554 - accuracy: 0.4922 - val_loss: 1.0274 - val_a
ccuracy: 0.4815
Epoch 62/100
22/22 [==============================] - 11s 478ms/step - loss: 1.1125 - accuracy: 0.5025 - val_loss: 0.9547 - val_a
ccuracy: 0.5185
Epoch 63/100
22/22 [==============================] - 10s 452ms/step - loss: 1.1782 - accuracy: 0.4693 - val_loss: 1.0710 - val_a
ccuracy: 0.4630
Epoch 64/100
22/22 [==============================] - 10s 458ms/step - loss: 1.0442 - accuracy: 0.5493 - val_loss: 1.0989 - val_a
ccuracy: 0.5000
Epoch 65/100
22/22 [==============================] - 10s 452ms/step - loss: 1.1208 - accuracy: 0.5166 - val_loss: 1.0244 - val_a
ccuracy: 0.4815
Epoch 66/100
22/22 [==============================] - 10s 448ms/step - loss: 1.0802 - accuracy: 0.5550 - val_loss: 1.1369 - val_a
ccuracy: 0.4630
Epoch 67/100
22/22 [==============================] - 10s 452ms/step - loss: 1.0410 - accuracy: 0.5021 - val_loss: 0.9946 - val_a
ccuracy: 0.5556
Epoch 68/100
22/22 [==============================] - 10s 455ms/step - loss: 0.9903 - accuracy: 0.5488 - val_loss: 1.0606 - val_a
ccuracy: 0.5000
Epoch 69/100
22/22 [==============================] - 10s 450ms/step - loss: 1.1437 - accuracy: 0.4657 - val_loss: 1.1046 - val_a
ccuracy: 0.5000
Epoch 70/100
22/22 [==============================] - 10s 455ms/step - loss: 1.0964 - accuracy: 0.5139 - val_loss: 1.1506 - val_a
ccuracy: 0.3704
Epoch 71/100
22/22 [==============================] - 10s 450ms/step - loss: 1.1422 - accuracy: 0.4703 - val_loss: 1.0154 - val_a
ccuracy: 0.5185
Epoch 72/100
22/22 [==============================] - 10s 448ms/step - loss: 1.0755 - accuracy: 0.5028 - val_loss: 1.0744 - val_a
ccuracy: 0.5370
Epoch 73/100
22/22 [==============================] - 10s 450ms/step - loss: 1.0748 - accuracy: 0.5571 - val_loss: 1.2563 - val_a
ccuracy: 0.3333
Epoch 74/100
22/22 [==============================] - 10s 469ms/step - loss: 1.1556 - accuracy: 0.4503 - val_loss: 1.0324 - val_a
ccuracy: 0.4630
Epoch 75/100
22/22 [==============================] - 10s 452ms/step - loss: 1.0393 - accuracy: 0.5014 - val_loss: 1.2241 - val_a
ccuracy: 0.3889
Epoch 76/100
22/22 [==============================] - 10s 453ms/step - loss: 1.1550 - accuracy: 0.4545 - val_loss: 0.9621 - val_a
ccuracy: 0.5926
Epoch 77/100
22/22 [==============================] - 10s 451ms/step - loss: 0.9899 - accuracy: 0.5751 - val_loss: 1.1104 - val_a
ccuracy: 0.4815
Epoch 78/100
22/22 [==============================] - 10s 452ms/step - loss: 1.0923 - accuracy: 0.4667 - val_loss: 1.0068 - val_a
ccuracy: 0.5185
Epoch 79/100
22/22 [==============================] - 10s 453ms/step - loss: 1.0327 - accuracy: 0.5435 - val_loss: 1.0395 - val_a
ccuracy: 0.5185
Epoch 80/100
22/22 [==============================] - 10s 450ms/step - loss: 1.0337 - accuracy: 0.5064 - val_loss: 0.9472 - val_a
ccuracy: 0.5370
Epoch 81/100
22/22 [==============================] - 10s 452ms/step - loss: 1.0964 - accuracy: 0.5223 - val_loss: 1.0315 - val_a
ccuracy: 0.5926
Epoch 82/100
22/22 [==============================] - 10s 451ms/step - loss: 1.1515 - accuracy: 0.4810 - val_loss: 1.0016 - val_a
ccuracy: 0.4815
Epoch 83/100
22/22 [==============================] - 10s 448ms/step - loss: 1.0465 - accuracy: 0.5470 - val_loss: 1.0561 - val_a
ccuracy: 0.5000
Epoch 84/100
22/22 [==============================] - 11s 484ms/step - loss: 0.9945 - accuracy: 0.5565 - val_loss: 0.9721 - val_a
ccuracy: 0.5000
Epoch 85/100
22/22 [==============================] - 10s 460ms/step - loss: 0.9792 - accuracy: 0.5724 - val_loss: 1.0032 - val_a
ccuracy: 0.4815
Epoch 86/100
22/22 [==============================] - 10s 450ms/step - loss: 1.0749 - accuracy: 0.5552 - val_loss: 1.1363 - val_a
ccuracy: 0.5000
```

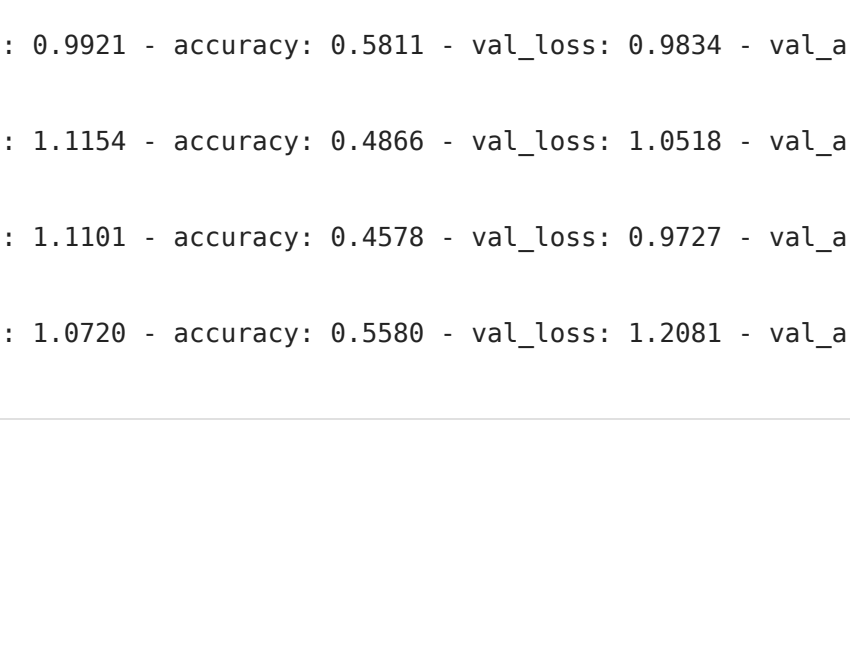
```
Epoch 87/100
22/22 [==============================] - 10s 448ms/step - loss: 1.0411 - accuracy: 0.5676 - val_loss: 1.1426 - val_a
ccuracy: 0.5000
Epoch 88/100
22/22 [==============================] - 10s 454ms/step - loss: 1.0759 - accuracy: 0.5650 - val_loss: 1.2639 - val_a
ccuracy: 0.3889
Epoch 89/100
22/22 [==============================] - 10s 448ms/step - loss: 1.1552 - accuracy: 0.4526 - val_loss: 1.0748 - val_a
ccuracy: 0.5000
Epoch 90/100
22/22 [==============================] - 10s 447ms/step - loss: 1.0683 - accuracy: 0.4320 - val_loss: 0.9396 - val_a
ccuracy: 0.6111
Epoch 91/100
22/22 [==============================] - 10s 458ms/step - loss: 1.1127 - accuracy: 0.5694 - val_loss: 1.0697 - val_a
ccuracy: 0.4630
Epoch 92/100
22/22 [==============================] - 10s 462ms/step - loss: 1.0718 - accuracy: 0.5240 - val_loss: 0.9555 - val_a
ccuracy: 0.5741
Epoch 93/100
22/22 [==============================] - 10s 450ms/step - loss: 1.0164 - accuracy: 0.5304 - val_loss: 1.0308 - val_a
ccuracy: 0.5000
Epoch 94/100
22/22 [==============================] - 10s 450ms/step - loss: 1.1175 - accuracy: 0.4886 - val_loss: 1.1295 - val_a
ccuracy: 0.5000
Epoch 95/100
22/22 [==============================] - 10s 452ms/step - loss: 1.0662 - accuracy: 0.5073 - val_loss: 0.9858 - val_a
ccuracy: 0.5000
Epoch 96/100
22/22 [==============================] - 10s 450ms/step - loss: 1.0934 - accuracy: 0.4986 - val_loss: 1.0530 - val_a
ccuracy: 0.5185
Epoch 97/100
22/22 [==============================] - 10s 448ms/step - loss: 0.9921 - accuracy: 0.5811 - val_loss: 0.9834 - val_a
ccuracy: 0.5556
Epoch 98/100
22/22 [==============================] - 10s 452ms/step - loss: 1.1154 - accuracy: 0.4866 - val_loss: 1.0518 - val_a
ccuracy: 0.5741
Epoch 99/100
22/22 [==============================] - 10s 459ms/step - loss: 1.1101 - accuracy: 0.4578 - val_loss: 0.9727 - val_a
ccuracy: 0.5185
Epoch 100/100
22/22 [==============================] - 10s 462ms/step - loss: 1.0720 - accuracy: 0.5580 - val_loss: 1.2081 - val_a
ccuracy: 0.4630
```

In [76]:

```
import matplotlib.pyplot as plt
plt.figure(figsize=(15,5))
plt.subplot(1,2,1)
plt.plot(history.history['accuracy'],label='Train_acc')
plt.plot(history.history['val_accuracy'],label='Test_acc')
plt.xlabel('epoch')
plt.ylabel('accuracy')
plt.ylim([0.4,0.7])
plt.legend(loc='lower right')

plt.subplot(1,2,2)
plt.plot(history.history['loss'],label='Train_loss')
plt.plot(history.history['val_loss'],label='Test_loss')
plt.xlabel('epoch')
plt.ylabel('loss')
plt.legend(loc='lower right')
```

Out[76]: <matplotlib.legend.Legend at 0x7f56bc1aa730>





Testing

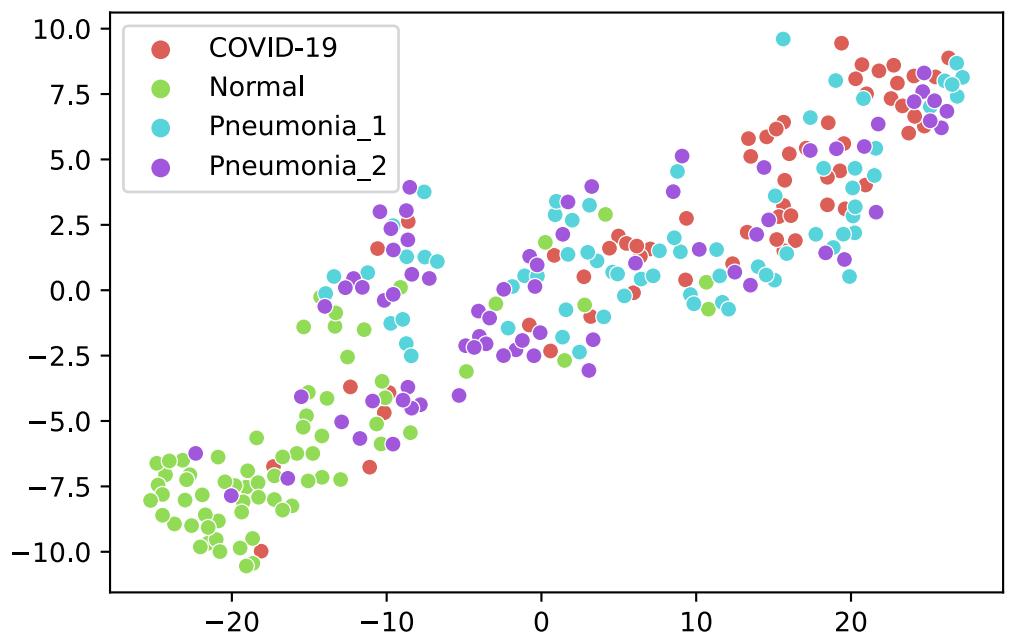
In [81]:
```
test_datagen = ImageDataGenerator(rescale=1. / 255)

eval_generator = test_datagen.flow_from_directory(TEST_DIR,target_size=IMAGE_SIZE,
                                                  batch_size=1,shuffle=True,seed=42,class_mode="categorical")
eval_generator.reset()
print(len(eval_generator))
x = n_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
 3/36 [=>............................] - ETA: 1s - loss: 1.3358 - accuracy: 0.3333    /home/shawn/.local/lib/python
3.8/site-packages/tensorflow/python/keras/engine/training.py:1877: UserWarning: `Model.evaluate_generator` is deprec
ated and will be removed in a future version. Please use `Model.evaluate`, which supports generators.
  warnings.warn('`Model.evaluate_generator` is deprecated and '
36/36 [==============================] - 2s 53ms/step - loss: 1.4799 - accuracy: 0.3056
Test loss: 1.4798626899719238
Test accuracy: 0.3055555522441864
```

TSNE

In [80]:
```
from sklearn.manifold import TSNE
import seaborn as sns
intermediate_layer_model = tf.keras.models.Model(inputs=n_model.input,
                                         outputs=n_model.get_layer('feature_dense').output)

tsne_eval_generator = test_datagen.flow_from_directory(DATASET_PATH,target_size=IMAGE_SIZE,
                                                       batch_size=1,shuffle=False,seed=42,class_mode="categorical")

intermediate_output = intermediate_layer_model.predict(tsne_eval_generator)
tsne = TSNE(n_components=2)
tsne_data = tsne.fit_transform(intermediate_output)
labels = map(lambda x: "COVID-19" if x == 0 else ("Normal" if x==1 else ("Pneumonia_1" if x==2 else "Pneumonia_2")),
             tsne_eval_generator.classes)

scat=sns.scatterplot(tsne_data[:,0],tsne_data[:,1], hue = labels, palette=sns.color_palette("hls", 4))

plt.show()
```

```
Found 270 images belonging to 4 classes.
/home/shawn/.local/lib/python3.8/site-packages/seaborn/_decorators.py:36: FutureWarning: Pass the following variable
s as keyword args: x, y. From version 0.12, the only valid positional argument will be `data`, and passing other arg
uments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```



In [ ]: