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 (https://drive.google.com/file/d/1Y88tgqpQ1Pjko_7rntcPowOJs_QNOrJ-/view)](https://drive.google.com/file/d/1Y88tgqpQ1Pjko_7rntcPowOJs_QNOrJ-/view).

- 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 [14]:

```
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[14]:

```
'2.4.1'
```

Load Image Data

In [15]:

```
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
NUM_EPOCHS    = 100
LEARNING_RATE = 0.0001 # start off with high rate first 0.001 and experiment with re
```

Generate Training and Validation Batches

In [16]:

```
train_datagen = ImageDataGenerator(rescale=1./255,rotation_range=50,featurewise_cent
                                   featurewise_std_normalization = True,width_shift_
                                   height_shift_range=0.2,shear_range=0.25,zoom_rang
                                   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_SIZ
                                                  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_SIZ
                                                  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.
```

[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 [17]:

```
from tensorflow.keras.layers import Flatten, Dense, Dropout
from tensorflow.keras.models import Sequential

resNet50V2 = tf.keras.applications.resnet_v2.ResNet50V2(include_top=False, weights='
resNet50V2.trainable = False


model = Sequential()
model.add(resNet50V2)
model.add(tf.keras.layers.AveragePooling2D(pool_size=7))
model.add(Flatten())
model.add(Dropout(0.5))
model.add(Dense(256, activation='relu', name = 'feature_dense'))
model.add(Dropout(0.4))
model.add(Dense(4, activation='softmax', kernel_initializer='he_normal'))

model.build(input_shape=(224, 224, 3))
model.summary()

model.compile(optimizer='adam', loss=tf.keras.losses.CategoricalCrossentropy(from_lc
```

```
Model: "sequential_6"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
resnet50v2 (Functional)      (None, 7, 7, 2048)        23564800
_________________________________________________________________
average_pooling2d_6 (Average (None, 1, 1, 2048)        0
_________________________________________________________________
flatten_6 (Flatten)          (None, 2048)              0
_________________________________________________________________
dropout_12 (Dropout)         (None, 2048)              0
_________________________________________________________________
feature_dense (Dense)        (None, 256)               524544
_________________________________________________________________
dropout_13 (Dropout)         (None, 256)               0
_________________________________________________________________
dense_11 (Dense)             (None, 4)                 1028
=================================================================
Total params: 24,090,372
Trainable params: 525,572
Non-trainable params: 23,564,800
_________________________________________________________________
```

[5 points] Train Model

In [31]:

```
#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

history = model.fit(train_batches, epochs = NUM_EPOCHS, validation_data = valid_batc
                              steps_per_epoch = STEP_SIZE_TRAIN,
                              validation_steps = STEP_SIZE_VALID)
```

```
22
6
Epoch 1/100
21/21 [==============================] - 6s 258ms/step - loss: 0.4858
- accuracy: 0.7427 - val_loss: 0.5517 - val_accuracy: 0.6400
Epoch 2/100
21/21 [==============================] - 5s 248ms/step - loss: 0.4466
- accuracy: 0.7961 - val_loss: 0.5039 - val_accuracy: 0.7200
Epoch 3/100
21/21 [==============================] - 5s 252ms/step - loss: 0.5720
- accuracy: 0.7573 - val_loss: 0.5082 - val_accuracy: 0.7400
Epoch 4/100
21/21 [==============================] - 5s 250ms/step - loss: 0.4393
- accuracy: 0.8301 - val_loss: 0.5916 - val_accuracy: 0.7400
Epoch 5/100
21/21 [==============================] - 5s 253ms/step - loss: 0.4962
- accuracy: 0.7913 - val_loss: 0.5045 - val_accuracy: 0.7200
Epoch 6/100
21/21 [==============================] - 5s 253ms/step - loss: 0.4476
- accuracy: 0.8252 - val_loss: 0.5042 - val_accuracy: 0.8000
Epoch 7/100
21/21 [==============================] - 5s 255ms/step - loss: 0.5342
- accuracy: 0.7427 - val_loss: 0.4907 - val_accuracy: 0.7800
Epoch 8/100
21/21 [==============================] - 5s 254ms/step - loss: 0.5506
- accuracy: 0.7767 - val_loss: 0.5670 - val_accuracy: 0.7200
Epoch 9/100
21/21 [==============================] - 5s 250ms/step - loss: 0.5050
- accuracy: 0.7816 - val_loss: 0.5519 - val_accuracy: 0.6600
Epoch 10/100
21/21 [==============================] - 5s 255ms/step - loss: 0.4957
- accuracy: 0.8000 - val_loss: 0.5600 - val_accuracy: 0.7400
Epoch 11/100
21/21 [==============================] - 5s 246ms/step - loss: 0.4233
- accuracy: 0.8252 - val_loss: 0.5141 - val_accuracy: 0.7200
Epoch 12/100
21/21 [==============================] - 5s 258ms/step - loss: 0.4559
- accuracy: 0.7619 - val_loss: 0.5097 - val_accuracy: 0.7000
Epoch 13/100
21/21 [==============================] - 5s 255ms/step - loss: 0.4363
- accuracy: 0.8058 - val_loss: 0.4676 - val_accuracy: 0.7200
Epoch 14/100
21/21 [==============================] - 5s 256ms/step - loss: 0.4471
- accuracy: 0.8350 - val_loss: 0.5750 - val_accuracy: 0.7200
Epoch 15/100
21/21 [==============================] - 5s 251ms/step - loss: 0.4703
- accuracy: 0.7913 - val_loss: 0.6417 - val_accuracy: 0.6800
```

```
Epoch 16/100
21/21 [==============================] - 5s 253ms/step - loss: 0.4650
- accuracy: 0.8143 - val_loss: 0.5855 - val_accuracy: 0.7000
Epoch 17/100
21/21 [==============================] - 5s 252ms/step - loss: 0.4075
- accuracy: 0.8301 - val_loss: 0.5211 - val_accuracy: 0.7200
Epoch 18/100
21/21 [==============================] - 5s 246ms/step - loss: 0.4691
- accuracy: 0.7864 - val_loss: 0.7428 - val_accuracy: 0.6600
Epoch 19/100
21/21 [==============================] - 5s 254ms/step - loss: 0.3773
- accuracy: 0.7961 - val_loss: 0.5769 - val_accuracy: 0.6400
Epoch 20/100
21/21 [==============================] - 5s 248ms/step - loss: 0.4622
- accuracy: 0.7961 - val_loss: 0.5920 - val_accuracy: 0.7200
Epoch 21/100
21/21 [==============================] - 5s 250ms/step - loss: 0.4556
- accuracy: 0.8155 - val_loss: 0.5438 - val_accuracy: 0.7600
Epoch 22/100
21/21 [==============================] - 5s 250ms/step - loss: 0.4233
- accuracy: 0.8010 - val_loss: 0.5106 - val_accuracy: 0.7600
Epoch 23/100
21/21 [==============================] - 6s 263ms/step - loss: 0.4598
- accuracy: 0.7816 - val_loss: 0.6831 - val_accuracy: 0.7200
Epoch 24/100
21/21 [==============================] - 6s 268ms/step - loss: 0.5413
- accuracy: 0.7621 - val_loss: 0.5064 - val_accuracy: 0.7800
Epoch 25/100
21/21 [==============================] - 5s 254ms/step - loss: 0.3985
- accuracy: 0.8010 - val_loss: 0.5345 - val_accuracy: 0.7200
Epoch 26/100
21/21 [==============================] - 5s 251ms/step - loss: 0.4273
- accuracy: 0.7961 - val_loss: 0.4714 - val_accuracy: 0.7200
Epoch 27/100
21/21 [==============================] - 5s 249ms/step - loss: 0.4684
- accuracy: 0.8155 - val_loss: 0.6050 - val_accuracy: 0.7200
Epoch 28/100
21/21 [==============================] - 5s 248ms/step - loss: 0.4175
- accuracy: 0.8301 - val_loss: 0.4335 - val_accuracy: 0.7800
Epoch 29/100
21/21 [==============================] - 5s 252ms/step - loss: 0.4470
- accuracy: 0.8107 - val_loss: 0.4932 - val_accuracy: 0.7400
Epoch 30/100
21/21 [==============================] - 5s 248ms/step - loss: 0.4405
- accuracy: 0.8301 - val_loss: 0.5145 - val_accuracy: 0.7400
Epoch 31/100
21/21 [==============================] - 5s 253ms/step - loss: 0.3929
- accuracy: 0.8107 - val_loss: 0.5792 - val_accuracy: 0.6800
Epoch 32/100
21/21 [==============================] - 5s 249ms/step - loss: 0.3828
- accuracy: 0.8398 - val_loss: 0.4645 - val_accuracy: 0.7800
Epoch 33/100
21/21 [==============================] - 5s 248ms/step - loss: 0.4622
- accuracy: 0.7961 - val_loss: 0.4662 - val_accuracy: 0.7800
Epoch 34/100
21/21 [==============================] - 5s 251ms/step - loss: 0.4884
- accuracy: 0.8155 - val_loss: 0.4877 - val_accuracy: 0.7400
Epoch 35/100
21/21 [==============================] - 5s 243ms/step - loss: 0.5218
- accuracy: 0.7864 - val_loss: 0.5458 - val_accuracy: 0.7600
Epoch 36/100
```

```
21/21 [==============================] - 5s 250ms/step - loss: 0.3817
- accuracy: 0.8592 - val_loss: 0.4216 - val_accuracy: 0.8000
Epoch 37/100
21/21 [==============================] - 5s 249ms/step - loss: 0.5139
- accuracy: 0.7670 - val_loss: 0.5301 - val_accuracy: 0.7400
Epoch 38/100
21/21 [==============================] - 5s 247ms/step - loss: 0.4351
- accuracy: 0.8107 - val_loss: 0.5353 - val_accuracy: 0.7600
Epoch 39/100
21/21 [==============================] - 5s 252ms/step - loss: 0.4694
- accuracy: 0.7864 - val_loss: 0.5985 - val_accuracy: 0.7200
Epoch 40/100
21/21 [==============================] - 5s 252ms/step - loss: 0.4139
- accuracy: 0.7961 - val_loss: 0.6548 - val_accuracy: 0.6800
Epoch 41/100
21/21 [==============================] - 5s 254ms/step - loss: 0.4444
- accuracy: 0.8495 - val_loss: 0.4982 - val_accuracy: 0.7600
Epoch 42/100
21/21 [==============================] - 5s 243ms/step - loss: 0.4673
- accuracy: 0.8058 - val_loss: 0.6582 - val_accuracy: 0.6200
Epoch 43/100
21/21 [==============================] - 5s 249ms/step - loss: 0.4648
- accuracy: 0.8447 - val_loss: 0.6049 - val_accuracy: 0.7200
Epoch 44/100
21/21 [==============================] - 5s 250ms/step - loss: 0.3984
- accuracy: 0.8350 - val_loss: 0.5524 - val_accuracy: 0.7200
Epoch 45/100
21/21 [==============================] - 5s 256ms/step - loss: 0.4385
- accuracy: 0.8350 - val_loss: 0.4835 - val_accuracy: 0.6600
Epoch 46/100
21/21 [==============================] - 5s 253ms/step - loss: 0.4895
- accuracy: 0.7913 - val_loss: 0.5412 - val_accuracy: 0.6800
Epoch 47/100
21/21 [==============================] - 5s 242ms/step - loss: 0.4175
- accuracy: 0.8010 - val_loss: 0.6342 - val_accuracy: 0.6800
Epoch 48/100
21/21 [==============================] - 5s 249ms/step - loss: 0.4678
- accuracy: 0.8155 - val_loss: 0.4734 - val_accuracy: 0.8200
Epoch 49/100
21/21 [==============================] - 5s 248ms/step - loss: 0.4053
- accuracy: 0.8107 - val_loss: 0.6032 - val_accuracy: 0.7000
Epoch 50/100
21/21 [==============================] - 5s 246ms/step - loss: 0.4348
- accuracy: 0.8252 - val_loss: 0.5651 - val_accuracy: 0.7600
Epoch 51/100
21/21 [==============================] - 5s 248ms/step - loss: 0.4000
- accuracy: 0.8301 - val_loss: 0.5911 - val_accuracy: 0.7200
Epoch 52/100
21/21 [==============================] - 5s 250ms/step - loss: 0.3967
- accuracy: 0.7864 - val_loss: 0.5400 - val_accuracy: 0.6600
Epoch 53/100
21/21 [==============================] - 5s 253ms/step - loss: 0.3358
- accuracy: 0.8544 - val_loss: 0.6214 - val_accuracy: 0.7000
Epoch 54/100
21/21 [==============================] - 5s 247ms/step - loss: 0.4491
- accuracy: 0.8058 - val_loss: 0.7071 - val_accuracy: 0.6800
Epoch 55/100
21/21 [==============================] - 5s 243ms/step - loss: 0.3328
- accuracy: 0.8350 - val_loss: 0.6301 - val_accuracy: 0.7600
Epoch 56/100
21/21 [==============================] - 5s 255ms/step - loss: 0.4887
```

```
- accuracy: 0.8107 - val_loss: 0.5833 - val_accuracy: 0.7200
Epoch 57/100
21/21 [==============================] - 5s 259ms/step - loss: 0.4797
- accuracy: 0.7810 - val_loss: 0.5436 - val_accuracy: 0.7000

Epoch 58/100
21/21 [==============================] - 5s 260ms/step - loss: 0.3860
- accuracy: 0.8350 - val_loss: 0.4737 - val_accuracy: 0.8000
Epoch 59/100
21/21 [==============================] - 5s 249ms/step - loss: 0.3946
- accuracy: 0.8447 - val_loss: 0.4505 - val_accuracy: 0.7600
Epoch 60/100
21/21 [==============================] - 5s 248ms/step - loss: 0.3603
- accuracy: 0.8252 - val_loss: 0.4626 - val_accuracy: 0.8000
Epoch 61/100
21/21 [==============================] - 5s 261ms/step - loss: 0.3956
- accuracy: 0.8058 - val_loss: 0.4387 - val_accuracy: 0.7600
Epoch 62/100
21/21 [==============================] - 5s 256ms/step - loss: 0.3906
- accuracy: 0.8107 - val_loss: 0.5827 - val_accuracy: 0.8000
Epoch 63/100
21/21 [==============================] - 5s 245ms/step - loss: 0.4298
- accuracy: 0.8058 - val_loss: 0.5866 - val_accuracy: 0.6800
Epoch 64/100
21/21 [==============================] - 5s 245ms/step - loss: 0.4284
- accuracy: 0.8252 - val_loss: 0.6281 - val_accuracy: 0.6400
Epoch 65/100
21/21 [==============================] - 5s 251ms/step - loss: 0.3859
- accuracy: 0.8301 - val_loss: 0.7395 - val_accuracy: 0.6800
Epoch 66/100
21/21 [==============================] - 5s 254ms/step - loss: 0.4529
- accuracy: 0.8301 - val_loss: 0.5606 - val_accuracy: 0.6800
Epoch 67/100
21/21 [==============================] - 5s 250ms/step - loss: 0.4158
- accuracy: 0.8301 - val_loss: 0.5817 - val_accuracy: 0.7000
Epoch 68/100
21/21 [==============================] - 5s 264ms/step - loss: 0.3658
- accuracy: 0.8252 - val_loss: 0.5212 - val_accuracy: 0.7400
Epoch 69/100
21/21 [==============================] - 5s 252ms/step - loss: 0.4595
- accuracy: 0.8252 - val_loss: 0.5393 - val_accuracy: 0.8000
Epoch 70/100
21/21 [==============================] - 5s 254ms/step - loss: 0.4900
- accuracy: 0.7864 - val_loss: 0.6473 - val_accuracy: 0.6600
Epoch 71/100
21/21 [==============================] - 5s 253ms/step - loss: 0.4670
- accuracy: 0.8190 - val_loss: 0.6775 - val_accuracy: 0.7200
Epoch 72/100
21/21 [==============================] - 5s 252ms/step - loss: 0.4161
- accuracy: 0.8058 - val_loss: 0.4968 - val_accuracy: 0.7000
Epoch 73/100
21/21 [==============================] - 5s 248ms/step - loss: 0.4285
- accuracy: 0.8252 - val_loss: 0.4886 - val_accuracy: 0.8000
Epoch 74/100
21/21 [==============================] - 5s 255ms/step - loss: 0.4568
- accuracy: 0.8155 - val_loss: 0.5531 - val_accuracy: 0.7600
Epoch 75/100
21/21 [==============================] - 5s 247ms/step - loss: 0.4584
- accuracy: 0.8010 - val_loss: 0.5055 - val_accuracy: 0.6800
Epoch 76/100
21/21 [==============================] - 5s 251ms/step - loss: 0.3693
```

```
- accuracy: 0.8592 - val_loss: 0.7053 - val_accuracy: 0.7200
Epoch 77/100
21/21 [==============================] - 5s 249ms/step - loss: 0.5354
- accuracy: 0.7864 - val_loss: 0.6983 - val_accuracy: 0.6600
Epoch 78/100
21/21 [==============================] - 5s 251ms/step - loss: 0.4301
- accuracy: 0.8155 - val_loss: 0.4767 - val_accuracy: 0.7000
Epoch 79/100
21/21 [==============================] - 5s 250ms/step - loss: 0.4451
- accuracy: 0.8350 - val_loss: 0.4509 - val_accuracy: 0.8400
Epoch 80/100
21/21 [==============================] - 5s 256ms/step - loss: 0.4183
- accuracy: 0.8155 - val_loss: 0.5497 - val_accuracy: 0.6800
Epoch 81/100
21/21 [==============================] - 5s 257ms/step - loss: 0.5009
- accuracy: 0.8155 - val_loss: 0.5002 - val_accuracy: 0.7800
Epoch 82/100
21/21 [==============================] - 5s 250ms/step - loss: 0.5094
- accuracy: 0.8204 - val_loss: 0.6596 - val_accuracy: 0.6400
Epoch 83/100
21/21 [==============================] - 5s 248ms/step - loss: 0.5317
- accuracy: 0.7573 - val_loss: 0.4365 - val_accuracy: 0.7600
Epoch 84/100
21/21 [==============================] - 5s 247ms/step - loss: 0.4788
- accuracy: 0.8107 - val_loss: 0.4618 - val_accuracy: 0.7600
Epoch 85/100
21/21 [==============================] - 5s 254ms/step - loss: 0.4511
- accuracy: 0.8010 - val_loss: 0.6157 - val_accuracy: 0.7200
Epoch 86/100
21/21 [==============================] - 5s 251ms/step - loss: 0.3465
- accuracy: 0.8301 - val_loss: 0.6056 - val_accuracy: 0.6600
Epoch 87/100
21/21 [==============================] - 5s 243ms/step - loss: 0.4365
- accuracy: 0.8350 - val_loss: 0.7311 - val_accuracy: 0.6000
Epoch 88/100
21/21 [==============================] - 5s 249ms/step - loss: 0.4486
- accuracy: 0.8301 - val_loss: 0.5756 - val_accuracy: 0.7000
Epoch 89/100
21/21 [==============================] - 5s 254ms/step - loss: 0.5072
- accuracy: 0.8107 - val_loss: 0.6265 - val_accuracy: 0.6400
Epoch 90/100
21/21 [==============================] - 5s 250ms/step - loss: 0.4543
- accuracy: 0.8107 - val_loss: 0.5097 - val_accuracy: 0.8000
Epoch 91/100
21/21 [==============================] - 5s 253ms/step - loss: 0.4971
- accuracy: 0.7816 - val_loss: 0.5044 - val_accuracy: 0.7200
Epoch 92/100
21/21 [==============================] - 5s 239ms/step - loss: 0.4150
- accuracy: 0.8107 - val_loss: 0.6215 - val_accuracy: 0.7000
Epoch 93/100
21/21 [==============================] - 5s 251ms/step - loss: 0.4535
- accuracy: 0.7816 - val_loss: 0.6169 - val_accuracy: 0.7600
Epoch 94/100
21/21 [==============================] - 5s 239ms/step - loss: 0.4599
- accuracy: 0.8155 - val_loss: 0.6102 - val_accuracy: 0.6800
Epoch 95/100
21/21 [==============================] - 5s 250ms/step - loss: 0.4037
- accuracy: 0.8155 - val_loss: 0.5319 - val_accuracy: 0.7400
Epoch 96/100
21/21 [==============================] - 5s 251ms/step - loss: 0.4334
- accuracy: 0.7961 - val_loss: 0.5831 - val_accuracy: 0.6400
```

```
Epoch 97/100
21/21 [==============================] - 5s 245ms/step - loss: 0.3511
- accuracy: 0.8592 - val_loss: 0.4566 - val_accuracy: 0.7000
Epoch 98/100
21/21 [==============================] - 5s 255ms/step - loss: 0.4392
- accuracy: 0.8107 - val_loss: 0.4516 - val_accuracy: 0.7600
Epoch 99/100
21/21 [==============================] - 5s 250ms/step - loss: 0.4237
- accuracy: 0.8252 - val_loss: 0.5459 - val_accuracy: 0.7400
Epoch 100/100
21/21 [==============================] - 5s 253ms/step - loss: 0.4809
- accuracy: 0.7864 - val_loss: 0.7709 - val_accuracy: 0.6200
```

[5 points] Plot Accuracy and Loss During Training

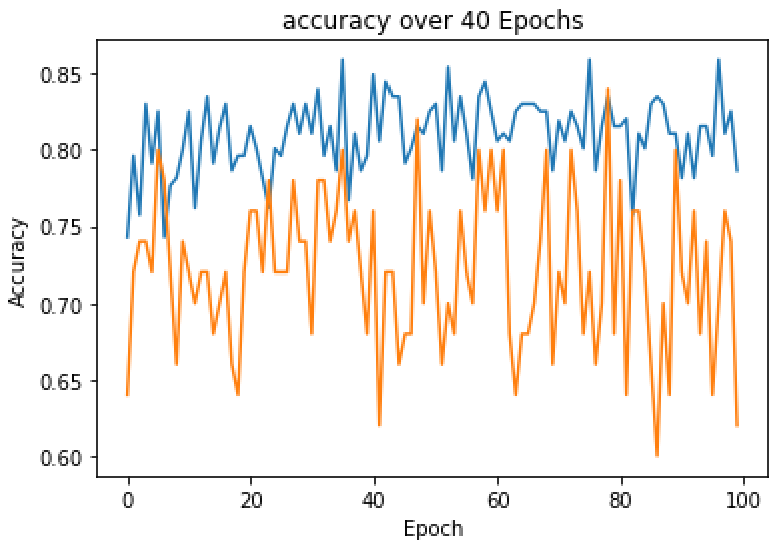
In [32]:

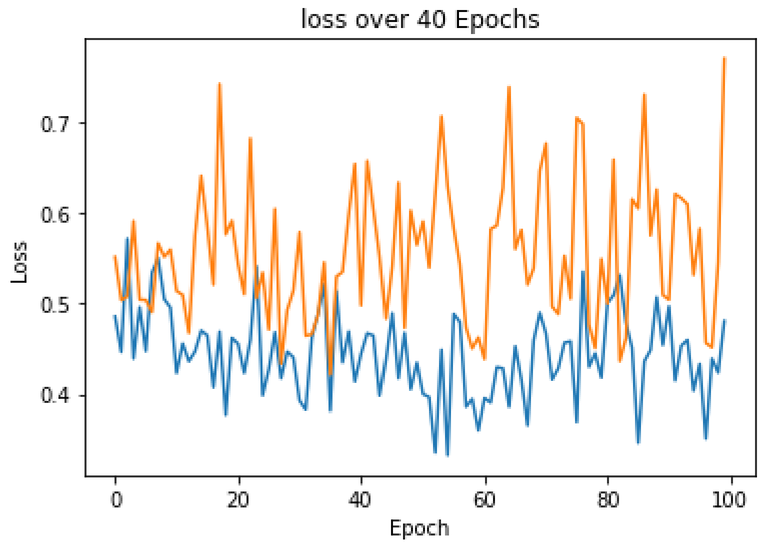
```
import matplotlib.pyplot as plt

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


#Accuracy
plt.plot(history.history['accuracy'], label = 'accuracy')
plt.plot(history.history['val_accuracy'], label = 'validation accuracy')
plt.title('accuracy over 40 Epochs')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.show()

#Loss
plt.plot(history.history['loss'], label = "loss")
plt.plot(history.history['val_loss'], label = "validation loss")
plt.title('loss over 40 Epochs')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.show()
```





Testing Model

In [33]:

```
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,
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 [==============================] - 1s 30ms/step - loss: 0.7694 -
accuracy: 0.6944
Test loss: 0.7693760991096497
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 [27]:

```
from sklearn.manifold import TSNE

intermediate_layer_model = tf.keras.models.Model(inputs=model.input,
                                                  outputs=model.get_layer('feature_dense').out

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

#raise NotImplementedError("Extract features from the tsne_data_generator and fit a
                           #"and plot the resulting 2D features of the four classes.'

pred = intermediate_layer_model.predict_generator(tsne_eval_generator,270,verbose=1)
print(pred.shape)
features = TSNE(n_components=2).fit_transform(pred)
print(features.shape)

x1,x2,x3,x4,y1,y2,y3,y4 = [],[],[],[],[],[],[],[]
cls = tsne_eval_generator.classes
for i in range(len(features)):
    if cls[i] == 0: #covid19
        x1.append(features[i, 0])
        y1.append(features[i, 1])
    elif cls[i] == 1: #norm
        x2.append(features[i, 0])
        y2.append(features[i, 1])
    elif cls[i] == 2: #bac
        x3.append(features[i, 0])
        y3.append(features[i, 1])
    else: #vir
        x4.append(features[i, 0])
        y4.append(features[i, 1])

plt.figure()
plt.plot(x1, y1, 'ro', label="COVID-19")
plt.plot(x2, y2, 'bo', label="Normal")
plt.plot(x3, y3, 'yo', label="Pneumonia_ba")
plt.plot(x4, y4, 'go', label="Pneumonia_vir")
plt.legend(loc='upper right')
```

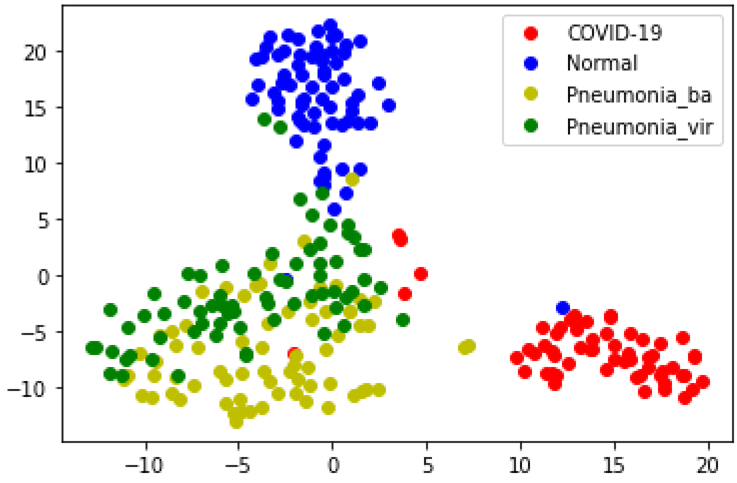
```
Found 270 images belonging to 4 classes.

/share/pkg.7/tensorflow/2.4.1/install/lib/SCC/../python3.8/site-packag
es/tensorflow/python/keras/engine/training.py:1905: UserWarning: `Mode
l.predict_generator` is deprecated and will be removed in a future ver
sion. Please use `Model.predict`, which supports generators.
  warnings.warn('`Model.predict_generator` is deprecated and '

270/270 [==============================] - 11s 37ms/step
(270, 256)
(270, 2)
```

Out[27]:

```
<matplotlib.legend.Legend at 0x2b64aa036670>
```


COVID-19
Normal
Pneumonia_ba
Pneumonia_vir
20
15
10
5
0
−5
−10
−10
−5
0
5
10
15
20