COVID-19-Image-Classification-phase1

November 23, 2021

1 Covid19 Image Classification - Phase 1

Abstract: Our Aim is to detect Covid19 from chest X-rays. The covid19 image dataset is small with 251 training and 60 test images belonging to 'NoFinding', 'Covid19' and 'Pneumonia' respectively. This dataset small and is insufficient to generalize. So for the purpoe of our project, in Phase-I we will first use NIH X-ray image data to retrain and finetune pretrained model architecture such as ResNet50V2, MobileNetV2 and VGG16.

1.0.1 Project Code Orginzation

Cookiecutter is a command-line utility that creates projects from cookiecutters (project templates), e.g. Python package projects, LaTeX documents, etc.

Installed and created the project template using Cookiecutter:
Follow instructions from https://ericbassett.tech/cookiecutter-data-science-crash-course/

1.1 Validating and pre-processing NIH X-ray metadata dataset

Following instructions use make tool, run commands from from your terminal from your project folder **Data Extraction:(execute only once)**
Download and extract image data for the project

```
1. Download and unzip the NIH X-ray images in data/raw
    Run: make get_nih_images
```

Data Validation:(execute only once)

```
3. Validate Dataset (rename columns and delete patient record with age greater than 100)
    Run: make validate_nih_images
```

Data Prepartion:(execute only once)

```
4. Prepare Dataset (add path attribute, split dataset into train and validation dataset)
    Run: make prepare_nih_images
```

This proudces the three output files in processed folder: 1. prepared_data_entry_2017.csv (full dataset) 2. prepared_train_data_entry_2017.csv (train_dataset) 3. prepared_valid_data_entry_2017.csv (validation_dataset)

Next, we use prepared_train_data_entry_2017.csv and prepared_valid_data_entry_2017.csv files to retrain CNN model architectures pre-trained using IMAGENET database

[7]:
```
# common imports
import os
import numpy as np
import matplotlib.pylab as plt
import pandas as pd
from glob import glob
from pathlib import Path
from functools import partial

from sklearn.model_selection import train_test_split

os.environ["TF_CPP_MIN_LOG_LEVEL"] = "2"
# prevent VRAM occupied
os.environ['TF_FORCE_GPU_ALLOW_GROWTH'] = 'true'

# TensorFlow 2.0 is required
import tensorflow as tf
from tensorflow import keras
assert tf.__version__ >= "2.0"

import warnings
warnings.filterwarnings('ignore')
```

[2]:
```
#change working directory  - as the images are located in data/raw in the␣
 ↪project folder (Execute this cell Once)
os.chdir("../")
```

[3]:
```
# Import functions for trianing the model
%load_ext autoreload
%autoreload 2
import src.models.train_model as train_model

# load tensorboard extension
%reload_ext tensorboard
```

[4]:
```
# Constants
SEED =42
IMAGE_SIZE = (224,224)
IMAGE_SHAPE = (224,224,3)
BATCH_SIZE = 32
SHUFFLE = True
TARGET_WIDTH= 224
TARGET_HEGIHT =224
NUM_CLASSES = 15 # number of ClassesNUM
NUM_EPOCHS = 10
PRETRAINED_MODELS = ['ResNet50V2', 'MobileNetV2', 'VGG16']
log_folder = 'logs' # logs folder
```

```
# Train and validate function
def train_and_validate_model(model_name,train_generator, valid_generator,␣
↪freeze_layers:bool = True,
                        activation: str = 'softmax', learning_rate: float =0.01,
                        fine_tune_learning_rate: float = 0.
↪001,fine_tune_at_layer:int = 186, num_classes: int = NUM_CLASSES):

    print(model_name)

    my_model = train_model.
↪get_base_model_with_new_toplayer(base_model=model_name,
                                                                freeze_layers =␣
↪freeze_layers,
                                                                num_classes =␣
↪num_classes,
                                                               ␣
↪activation_func=activation,
                                                                learning_rate =␣
↪learning_rate,
                                                                input_shape =␣
↪IMAGE_SHAPE)

    my_model_history = train_model.fit_model(my_model, train_generator,␣
↪valid_generator,num_epochs=NUM_EPOCHS,batch_size=BATCH_SIZE)

    print(f'{model_name} Accuracy and Loss plots')
    train_model.plot_accuracy_and_loss(my_model_history)


    print("\n")
    #fine_tune model_name
    model_ft = train_model.
↪fine_tune_model(my_model,fine_tune_learning_rate,optimizer='Adam',fine_tune_at_layer=fine_t


    print("\n")
    print(f'Fine-Tuned {model_name} Training and Validation: ')
    model_ft_history = train_model.fit_model(model_ft, train_generator,
            valid_generator, num_epochs=NUM_EPOCHS,batch_size=BATCH_SIZE)
    print(f'Fine-Tuned {model_name} Accuracy and Loss plots')
    train_model.plot_accuracy_and_loss(model_ft_history)
    return model_ft
```

[5]:
```
def load_data():
    nih_xrays_train_df = pd.read_csv('data/processed/
 ↪prepared_train_data_entry_2017.csv')
    nih_xrays_valid_df = pd.read_csv('data/processed/
 ↪prepared_valid_data_entry_2017.csv')
    return nih_xrays_train_df,nih_xrays_valid_df
nih_xrays_train_df, nih_xrays_valid_df = load_data()
```

[6]:
```
# Get fourteen unique diagnosis
# It is a function that takes a series of iterables and returns one iterable
# The asterisk "*" is used in Python to define a variable number of arguments.
# The asterisk character has to precede a variable identifier in the parameter␣
 ↪list
from itertools import chain
all_labels = np.unique(list(chain(*nih_xrays_train_df['finding_label'].
 ↪map(lambda x: x.split('|')).tolist())))
# remove the empty label
all_labels = [x for x in all_labels if len(x)>0]
print('All Labels ({}): {}'.format(len(all_labels), all_labels))
```

```
All Labels (15): ['Atelectasis', 'Cardiomegaly', 'Consolidation', 'Edema',
'Effusion', 'Emphysema', 'Fibrosis', 'Hernia', 'Infiltration', 'Mass',
'NoFinding', 'Nodule', 'Pleural_Thickening', 'Pneumonia', 'Pneumothorax']
```

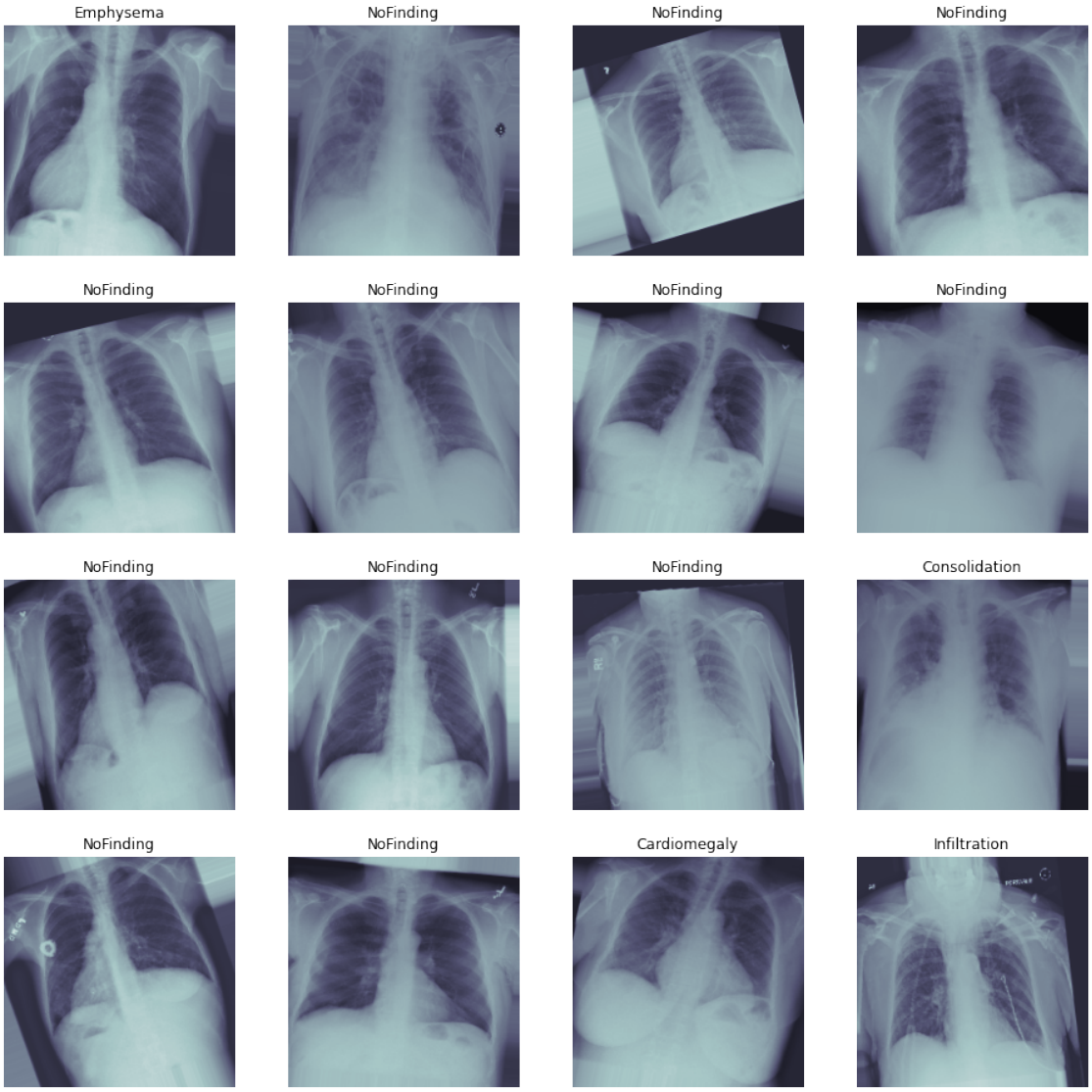
1.2 Preprocess Images

[7]:
```
from keras.applications.resnet_v2 import preprocess_input
from keras.preprocessing.image import ImageDataGenerator

train_generator = train_model.
 ↪get_image_data_generator(nih_xrays_train_df,batch_size=BATCH_SIZE,image_size=IMAGE_SIZE,lab
valid_generator = train_model.
 ↪get_image_data_generator(nih_xrays_train_df,batch_size=BATCH_SIZE,image_size=IMAGE_SIZE,lab
```

```
Found 73141 validated image filenames belonging to 15 classes.
Found 73141 validated image filenames belonging to 15 classes.
```

1.2.1 Visualize Images

[8]:
```
t_x, t_y = next(train_generator)
fig, m_axs = plt.subplots(4, 4, figsize = (16, 16))
for (c_x, c_y, c_ax) in zip(t_x, t_y, m_axs.flatten()):
    c_ax.imshow(c_x[:,:,0], cmap = 'bone', vmin = -1.5, vmax = 1.5)
    c_ax.set_title(', '.join([n_class for n_class, n_score in zip(all_labels,␣
 ↪c_y)
                              if n_score>0.5]))
    c_ax.axis('off')
```



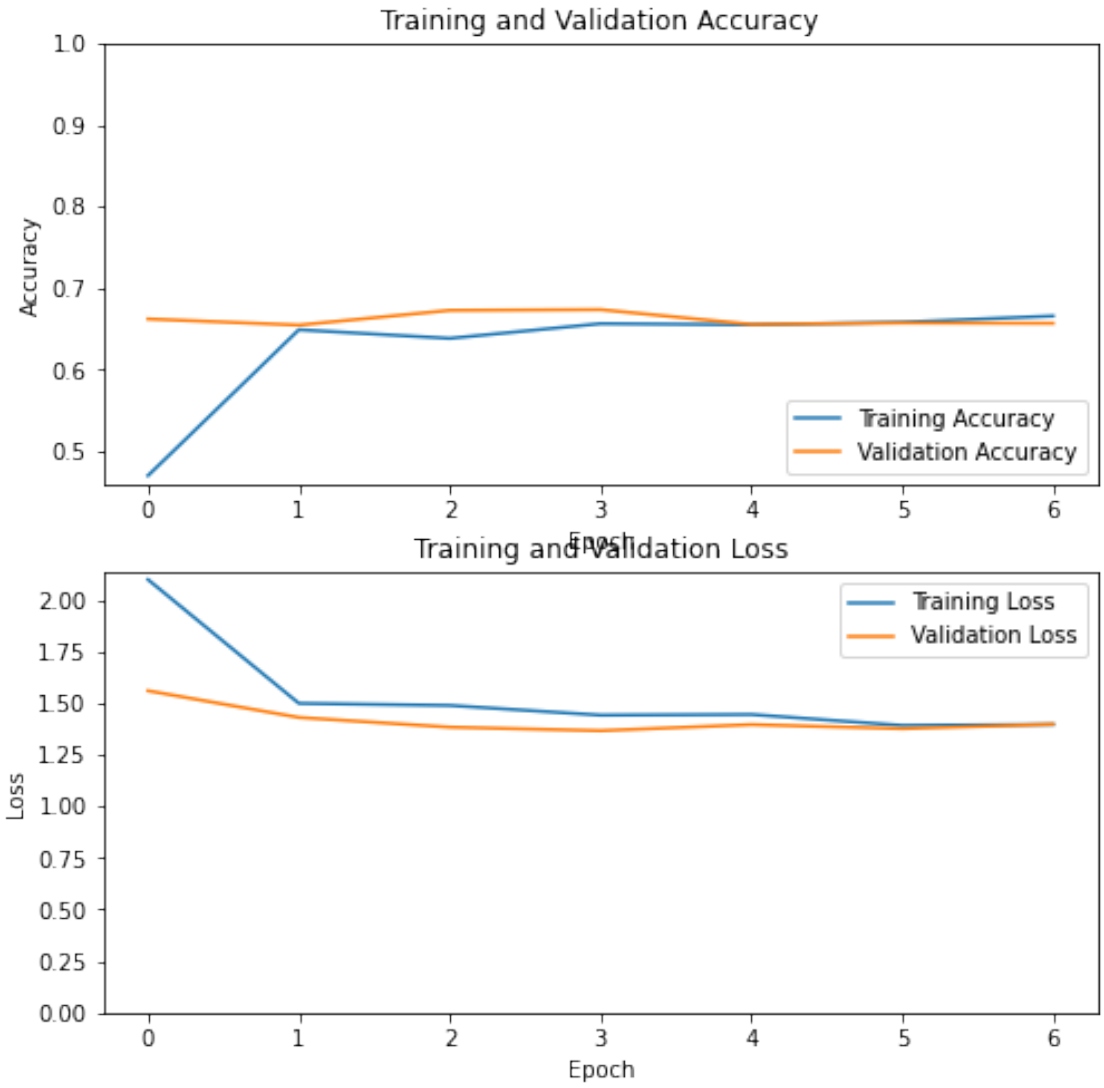
1.3 Experiment 1: Classification using all data

1.3.1 ResNetV250

[9]:
```
model_name = PRETRAINED_MODELS[0]
model = train_and_validate_model(model_name = model_name,␣
 ↪train_generator=train_generator, valid_generator=valid_generator)
# for now saving resnetv2 as best model
model.save('models/'+ model_name + 'exp1')
```

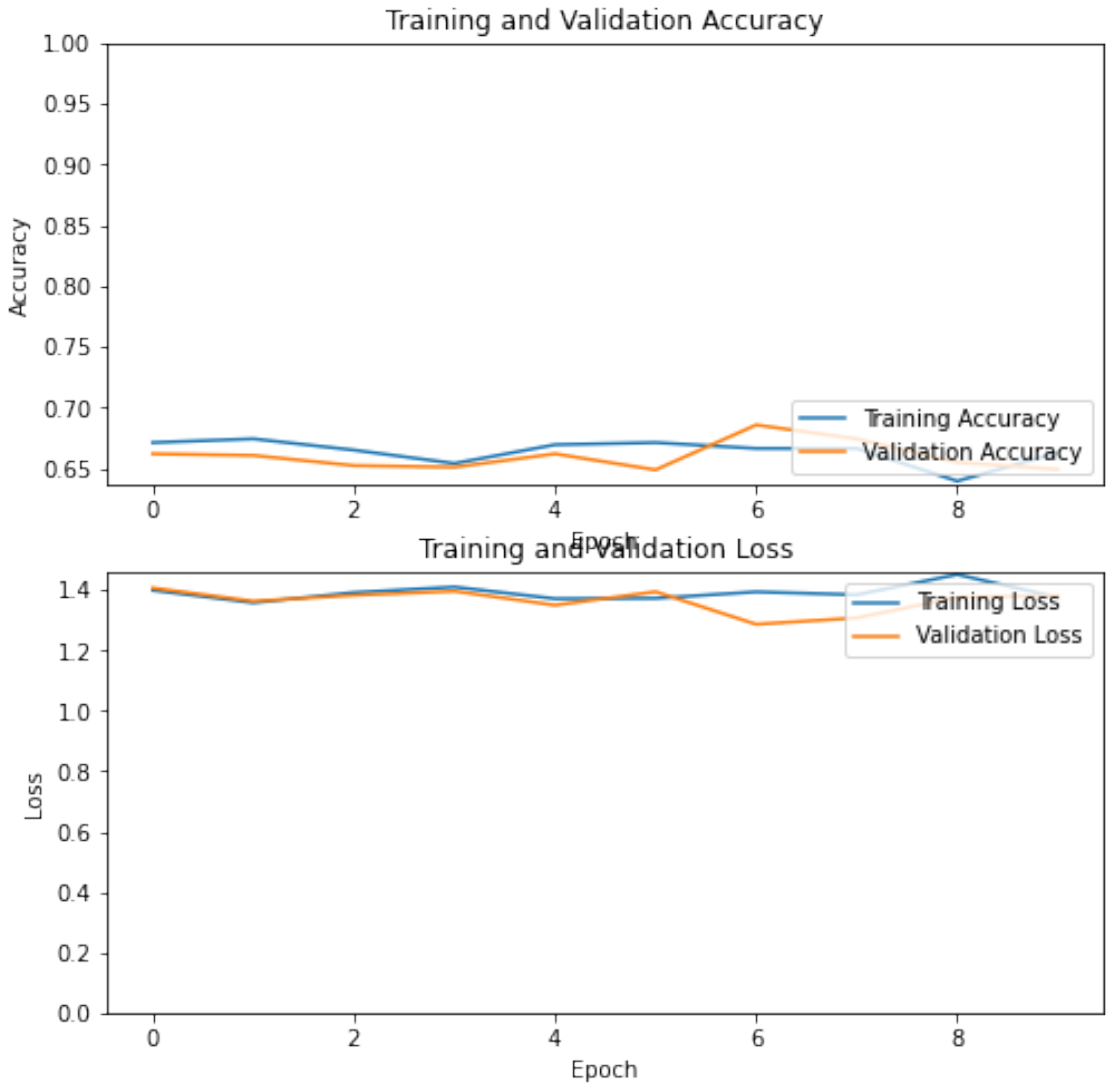
```
ResNet50V2
learning rate 0.01
```

```
Downloading ResNet50V2
Epoch 1/10
71/71 [==============================] - 72s 974ms/step - loss: 2.0975 -
accuracy: 0.4701 - val_loss: 1.5592 - val_accuracy: 0.6620
Epoch 2/10
71/71 [==============================] - 68s 964ms/step - loss: 1.4983 -
accuracy: 0.6488 - val_loss: 1.4300 - val_accuracy: 0.6545
Epoch 3/10
71/71 [==============================] - 68s 963ms/step - loss: 1.4883 -
accuracy: 0.6382 - val_loss: 1.3830 - val_accuracy: 0.6725
Epoch 4/10
71/71 [==============================] - 68s 964ms/step - loss: 1.4413 -
accuracy: 0.6562 - val_loss: 1.3659 - val_accuracy: 0.6734
Epoch 5/10
71/71 [==============================] - 68s 957ms/step - loss: 1.4443 -
accuracy: 0.6549 - val_loss: 1.3955 - val_accuracy: 0.6554
Epoch 6/10
71/71 [==============================] - 67s 956ms/step - loss: 1.3914 -
accuracy: 0.6580 - val_loss: 1.3758 - val_accuracy: 0.6571

Epoch 00006: ReduceLROnPlateau reducing learning rate to 0.0019999999552965165.
Epoch 7/10
71/71 [==============================] - 68s 963ms/step - loss: 1.3962 -
accuracy: 0.6656 - val_loss: 1.3976 - val_accuracy: 0.6567
Restoring model weights from the end of the best epoch.
Epoch 00007: early stopping
ResNet50V2 Accuracy and Loss plots
```




```
Fine-Tuned ResNet50V2 Training and Validation:
Epoch 1/10
71/71 [==============================] - 71s 977ms/step - loss: 1.3992 -
accuracy: 0.6712 - val_loss: 1.4069 - val_accuracy: 0.6620
Epoch 2/10
71/71 [==============================] - 68s 965ms/step - loss: 1.3581 -
accuracy: 0.6743 - val_loss: 1.3633 - val_accuracy: 0.6607
Epoch 3/10
71/71 [==============================] - 68s 967ms/step - loss: 1.3919 -
accuracy: 0.6651 - val_loss: 1.3828 - val_accuracy: 0.6523
```

```
Epoch 4/10
71/71 [==============================] - 68s 965ms/step - loss: 1.4102 -
accuracy: 0.6540 - val_loss: 1.3963 - val_accuracy: 0.6510

Epoch 00004: ReduceLROnPlateau reducing learning rate to 0.00020000000949949026.
Epoch 5/10
71/71 [==============================] - 68s 963ms/step - loss: 1.3714 -
accuracy: 0.6695 - val_loss: 1.3500 - val_accuracy: 0.6620
Epoch 6/10
71/71 [==============================] - 68s 966ms/step - loss: 1.3728 -
accuracy: 0.6712 - val_loss: 1.3946 - val_accuracy: 0.6488
Epoch 7/10
71/71 [==============================] - 68s 961ms/step - loss: 1.3941 -
accuracy: 0.6664 - val_loss: 1.2868 - val_accuracy: 0.6857
Epoch 8/10
71/71 [==============================] - 68s 960ms/step - loss: 1.3845 -
accuracy: 0.6664 - val_loss: 1.3077 - val_accuracy: 0.6743
Epoch 9/10
71/71 [==============================] - 68s 965ms/step - loss: 1.4515 -
accuracy: 0.6395 - val_loss: 1.3760 - val_accuracy: 0.6549

Epoch 00009: ReduceLROnPlateau reducing learning rate to 4.0000001899898055e-05.
Epoch 10/10
71/71 [==============================] - 68s 967ms/step - loss: 1.3760 -
accuracy: 0.6629 - val_loss: 1.3819 - val_accuracy: 0.6492
Restoring model weights from the end of the best epoch.
Epoch 00010: early stopping
Fine-Tuned ResNet50V2 Accuracy and Loss plots
```




```
INFO:tensorflow:Assets written to: models/ResNet50V2exp1/assets
```

```
/home/jay/Temple/apps/envs/env/lib/python3.9/site-
packages/keras/utils/generic_utils.py:494: CustomMaskWarning: Custom mask layers
require a config and must override get_config. When loading, the custom mask
layer must be passed to the custom_objects argument.
  warnings.warn('Custom mask layers require a config and must override '
```

1.3.2 MobileNetV2

[10]:
```
# model_name = PRETRAINED_MODELS[1]
# model = train_and_validate_model(model_name = model_name,␣
 ↪train_generator=train_generator, valid_generator=valid_generator)
# # for now saving resnetv2 as best model
```

```
# model.save('models/'+ model_name + 'exp1')
```

1.3.3 VGG16

[11]:
```
# model_name = PRETRAINED_MODELS[2]
# model = train_and_validate_model(model_name = model_name,␣
 ↪train_generator=train_generator, valid_generator=valid_generator)
# # for now saving resnetv2 as best model
# model.save('models/'+ model_name + 'exp1')
```

1.4 Experiment 2: Balance the dataset

[12]:
```
nih_xrays_df = pd.read_csv('data/processed/prepared_data_entry_2017.csv')
```

[15]:
```
def sample_with_weights(df, all_labels, num_samples: int = 40000):
    for lbl in all_labels:
        df[lbl] = df['finding_label'].map(lambda find: 1 if lbl in find else 0)
    df['encoding'] = [[1 if l in lbl.split('|') else 0 for l in all_labels] for␣
 ↪lbl in nih_xrays_df['finding_label']]

    class_count = {}
    for lbl in all_labels:
        class_count[lbl] = df[lbl].sum()

    classweight = {}
    for lbl in all_labels :
        classweight[lbl] = 1/class_count[lbl]

    classweight['NoFinding'] /= 2   #Extra penalising the none class
    def apply_weights(row):
        weight = 0
        for lbl in all_labels:
            if(row[lbl]==1):
                weight += classweight[lbl]
        return weight
    new_weights = df.apply(apply_weights, axis=1)
    sampled_data = df.sample(50000, weights = new_weights)
    from sklearn.model_selection import GroupShuffleSplit
    from sklearn.model_selection import train_test_split

    nih_required_columns = {
            'patient_id',
            'image_name',
            'path',
            'finding_label'
```

```
        }


    sampled_data = sampled_data[nih_required_columns]


    group_shuffle_split = GroupShuffleSplit(n_splits=1, train_size=0.8,␣
 ↪random_state=42)

    for train_idx, valid_idx in group_shuffle_split.split(sampled_data[:None],\
        groups=sampled_data[:None]['patient_id'].values):
        train_df = sampled_data.iloc[train_idx]
        valid_df = sampled_data.iloc[valid_idx]

    return train_df, valid_df
```

[16]:
```
train_df, valid_df =␣
 ↪sample_with_weights(nih_xrays_df,all_labels,num_samples=40000)
sampled_train_gen = train_model.
 ↪get_image_data_generator(train_df,batch_size=BATCH_SIZE,image_size=IMAGE_SIZE,lables=all_la
sampled_valid_gen = train_model.
 ↪get_image_data_generator(valid_df,batch_size=BATCH_SIZE,image_size=IMAGE_SIZE,lables=all_la
```

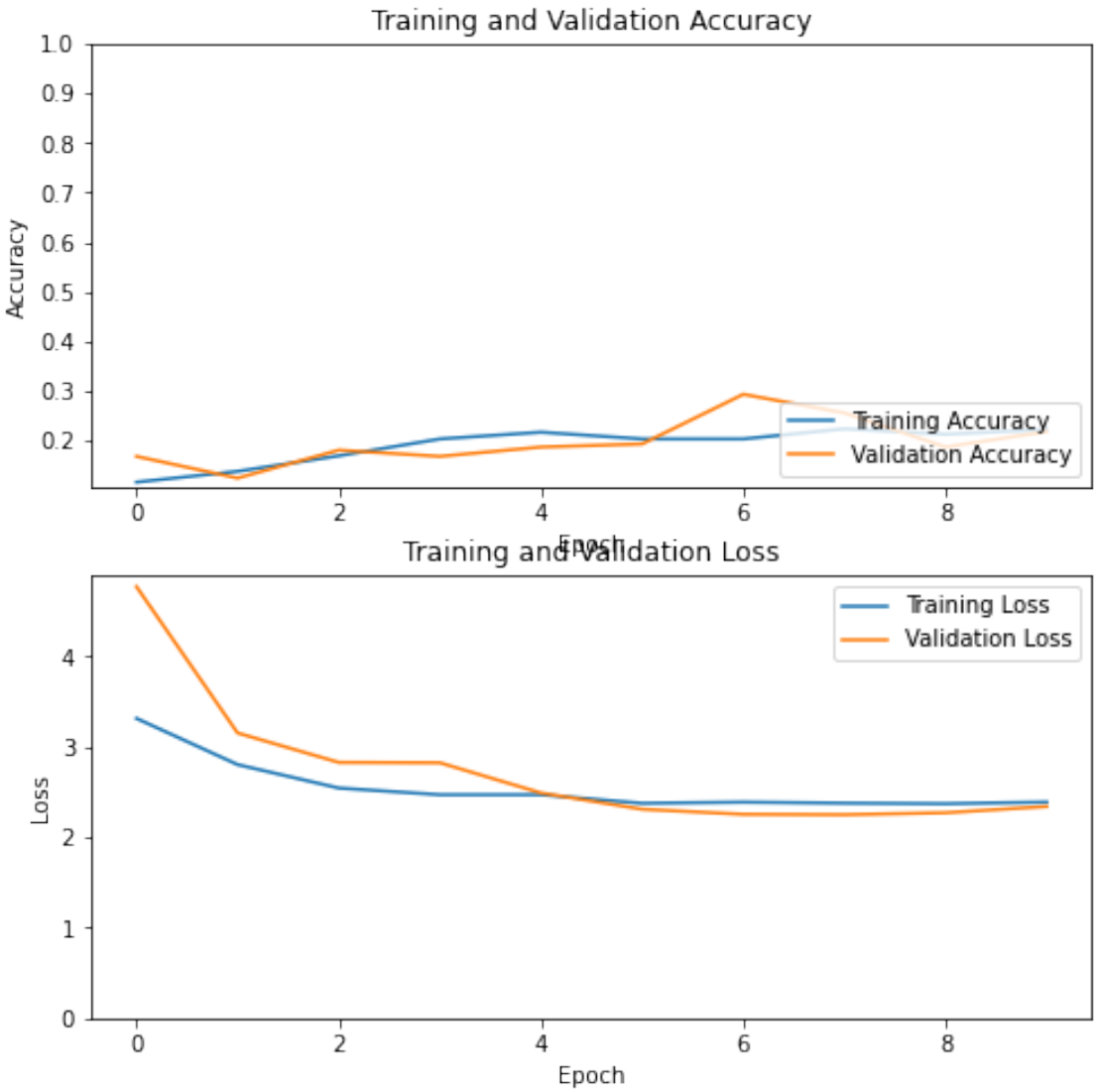
```
Found 24062 validated image filenames belonging to 15 classes.
Found 5912 validated image filenames belonging to 15 classes.
```

1.4.1 ResNet50V2

[17]:
```
model_name = PRETRAINED_MODELS[0]
model = train_and_validate_model(model_name = model_name,␣
 ↪train_generator=sampled_train_gen, valid_generator=sampled_valid_gen)
# for now saving resnetv2 as best model
model.save('models/'+ model_name + 'exp2')
```

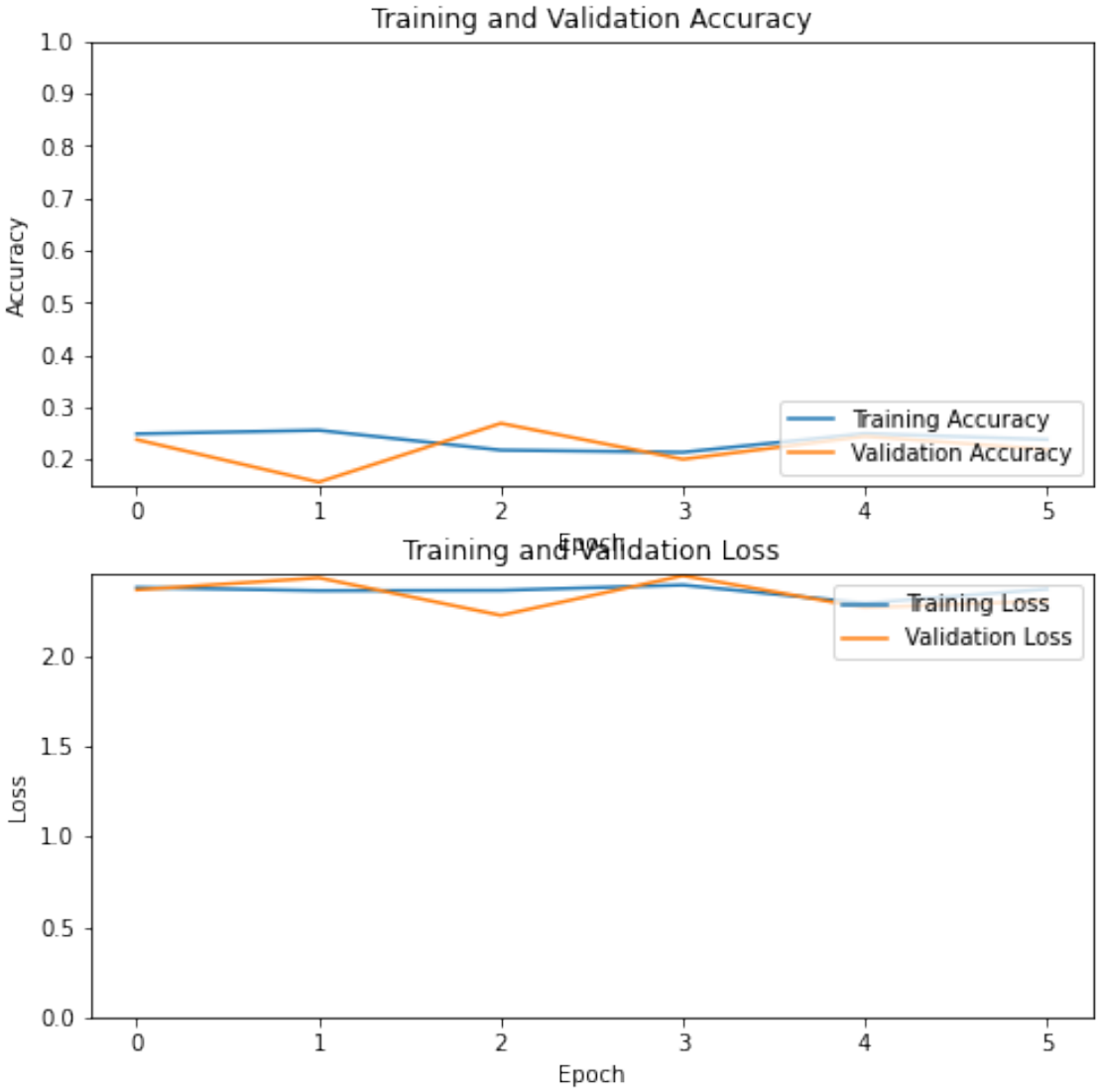
```
ResNet50V2
learning rate 0.01
Downloading ResNet50V2
Epoch 1/10
23/23 [==============================] - 17s 653ms/step - loss: 3.3159 -
accuracy: 0.1168 - val_loss: 4.7731 - val_accuracy: 0.1688
Epoch 2/10
23/23 [==============================] - 14s 611ms/step - loss: 2.8046 -
accuracy: 0.1386 - val_loss: 3.1542 - val_accuracy: 0.1250
Epoch 3/10
23/23 [==============================] - 14s 614ms/step - loss: 2.5455 -
accuracy: 0.1698 - val_loss: 2.8292 - val_accuracy: 0.1813
Epoch 4/10
```

```
23/23 [==============================] - 14s 612ms/step - loss: 2.4717 -
accuracy: 0.2038 - val_loss: 2.8251 - val_accuracy: 0.1688
Epoch 5/10
23/23 [==============================] - 14s 605ms/step - loss: 2.4716 -
accuracy: 0.2174 - val_loss: 2.4911 - val_accuracy: 0.1875
Epoch 6/10
23/23 [==============================] - 14s 612ms/step - loss: 2.3768 -
accuracy: 0.2038 - val_loss: 2.3113 - val_accuracy: 0.1937
Epoch 7/10
23/23 [==============================] - 14s 606ms/step - loss: 2.3898 -
accuracy: 0.2038 - val_loss: 2.2551 - val_accuracy: 0.2937
Epoch 8/10
23/23 [==============================] - 14s 611ms/step - loss: 2.3784 -
accuracy: 0.2242 - val_loss: 2.2508 - val_accuracy: 0.2562
Epoch 9/10
23/23 [==============================] - 14s 606ms/step - loss: 2.3726 -
accuracy: 0.2133 - val_loss: 2.2724 - val_accuracy: 0.1875
Epoch 10/10
23/23 [==============================] - 14s 612ms/step - loss: 2.3901 -
accuracy: 0.2228 - val_loss: 2.3416 - val_accuracy: 0.2188

Epoch 00010: ReduceLROnPlateau reducing learning rate to 0.0019999999552965165.
ResNet50V2 Accuracy and Loss plots
```




```
Fine-Tuned ResNet50V2 Training and Validation:
Epoch 1/10
23/23 [==============================] - 17s 652ms/step - loss: 2.3763 -
accuracy: 0.2486 - val_loss: 2.3634 - val_accuracy: 0.2375
Epoch 2/10
23/23 [==============================] - 14s 611ms/step - loss: 2.3587 -
accuracy: 0.2554 - val_loss: 2.4299 - val_accuracy: 0.1562
Epoch 3/10
23/23 [==============================] - 14s 613ms/step - loss: 2.3604 -
```

```
accuracy: 0.2174 - val_loss: 2.2225 - val_accuracy: 0.2688
Epoch 4/10
23/23 [==============================] - 14s 610ms/step - loss: 2.3900 -
accuracy: 0.2133 - val_loss: 2.4417 - val_accuracy: 0.2000
Epoch 5/10
23/23 [==============================] - 14s 616ms/step - loss: 2.2899 -
accuracy: 0.2493 - val_loss: 2.2641 - val_accuracy: 0.2438

Epoch 00005: ReduceLROnPlateau reducing learning rate to 0.00020000000949949026.
Epoch 6/10
23/23 [==============================] - 14s 601ms/step - loss: 2.3673 -
accuracy: 0.2378 - val_loss: 2.3019 - val_accuracy: 0.2188
Restoring model weights from the end of the best epoch.
Epoch 00006: early stopping
Fine-Tuned ResNet50V2 Accuracy and Loss plots
```



```
INFO:tensorflow:Assets written to: models/ResNet50V2exp2/assets

/home/jay/Temple/apps/envs/env/lib/python3.9/site-
packages/keras/utils/generic_utils.py:494: CustomMaskWarning: Custom mask layers
require a config and must override get_config. When loading, the custom mask
layer must be passed to the custom_objects argument.
  warnings.warn('Custom mask layers require a config and must override '
```

1.4.2 MobileNetV2

[ ]:
```
# model_name = PRETRAINED_MODELS[1]
# model = train_and_validate_model(model_name = model_name,␣
 ↪train_generator=sampled_train_gen, valid_generator=sampled_valid_gen)
# # for now saving resnetv2 as best model
# model.save('models/'+ model_name + 'exp2')
```

1.4.3 VGG16

[ ]:
```
# model_name = PRETRAINED_MODELS[2]
# model = train_and_validate_model(model_name = model_name,␣
 ↪train_generator=sampled_train_gen, valid_generator=sampled_valid_gen)
# # for now saving resnetv2 as best model
# model.save('models/'+ model_name + 'exp2')
```

1.4.4 Experiment 3: Sub Sampling Classes

[19]:
```
sub_samples = ['Cardiomegaly','Effusion','Emphysema', 'Fibrosis',␣
 ↪'Infiltration', 'Pneumonia', 'Pneumothorax','Pleural_Thickening']
```

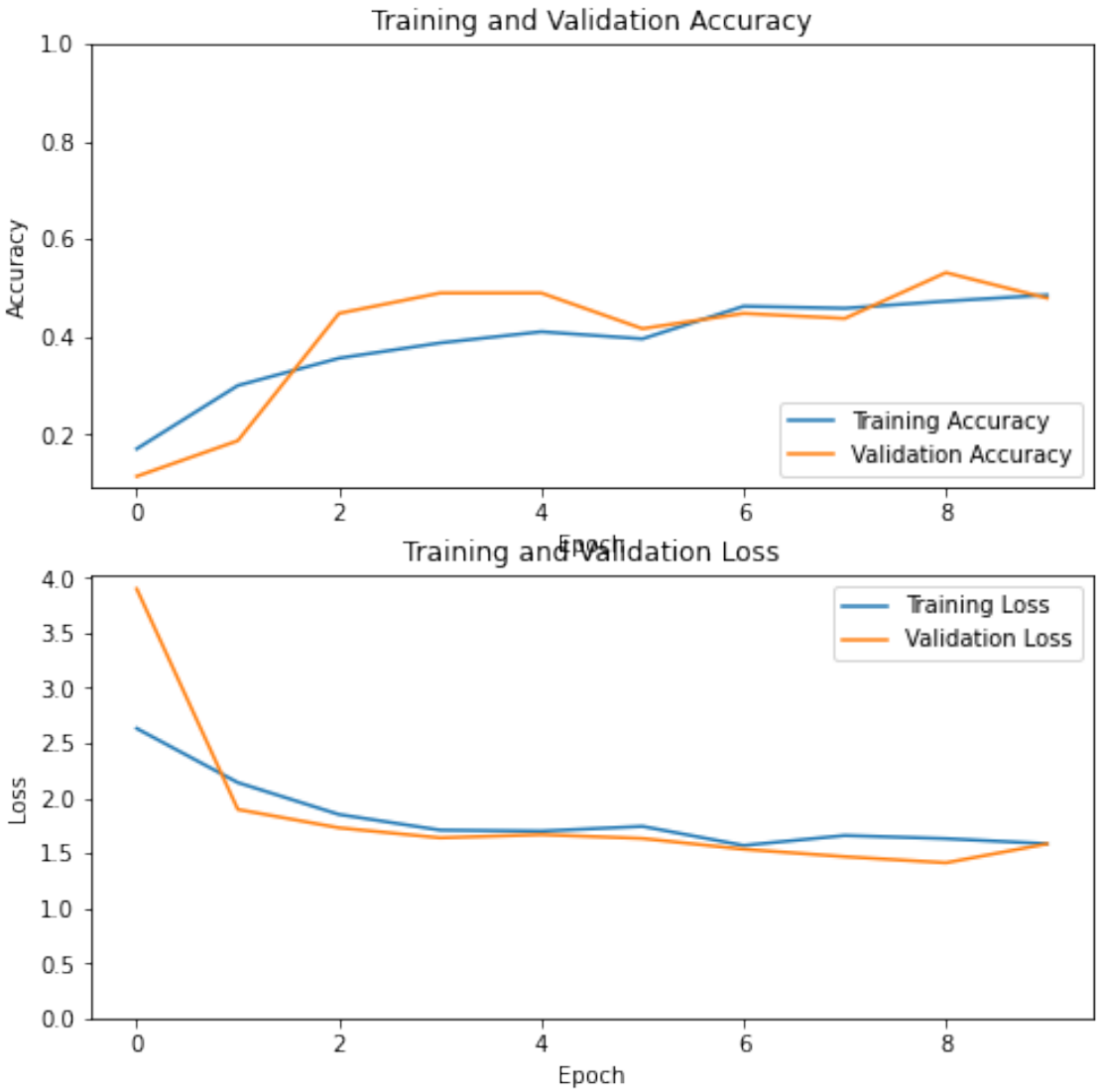
[20]:
```
sub_nih_xrays_train_df  = nih_xrays_train_df␣
 ↪[nih_xrays_train_df['finding_label'].isin(sub_samples)]
sub_nih_xrays_valid_df = nih_xrays_valid_df[nih_xrays_valid_df['finding_label'].
 ↪isin(sub_samples)]
sub_sampled_train_gen = train_model.
 ↪get_image_data_generator(sub_nih_xrays_train_df,batch_size=BATCH_SIZE,image_size=IMAGE_SIZE
sub_sampled_valid_gen = train_model.
 ↪get_image_data_generator(sub_nih_xrays_valid_df,batch_size=BATCH_SIZE,image_size=IMAGE_SIZE
```

```
Found 15794 validated image filenames belonging to 8 classes.
Found 4060 validated image filenames belonging to 8 classes.
```

1.4.5 ResNet50V2

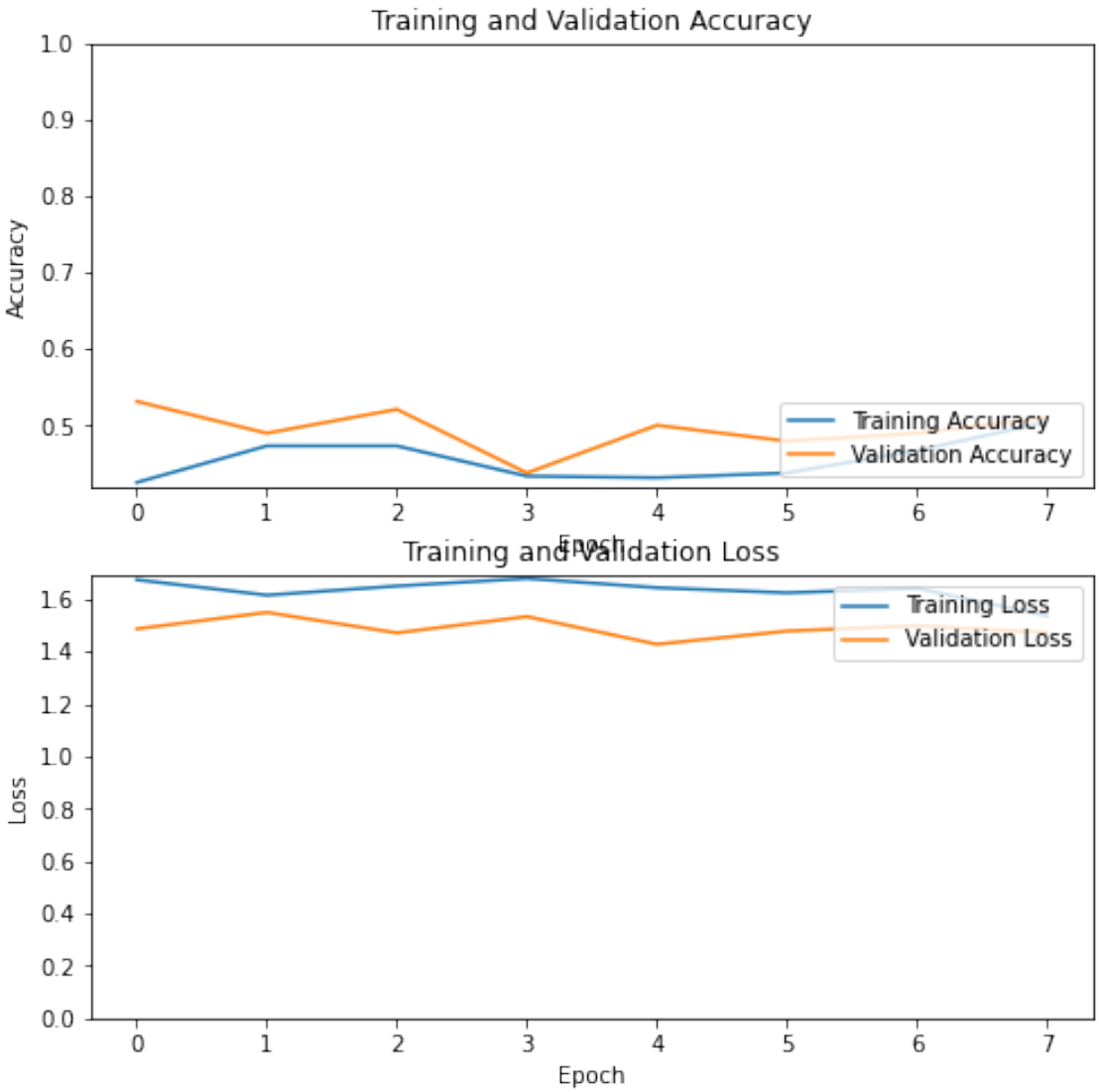
[21]:
```
model_name = PRETRAINED_MODELS[0]
model = train_and_validate_model(model_name = model_name,␣
 ↪train_generator=sub_sampled_train_gen,␣
 ↪valid_generator=sub_sampled_valid_gen, num_classes=len(sub_samples))
# for now saving resnetv2 as best model
model.save('models/'+ model_name + 'exp3')
```

```
ResNet50V2
learning rate 0.01
Downloading ResNet50V2
Epoch 1/10
15/15 [==============================] - 13s 689ms/step - loss: 2.6315 -
accuracy: 0.1708 - val_loss: 3.8989 - val_accuracy: 0.1146
Epoch 2/10
15/15 [==============================] - 9s 598ms/step - loss: 2.1399 -
accuracy: 0.3004 - val_loss: 1.8957 - val_accuracy: 0.1875
Epoch 3/10
15/15 [==============================] - 9s 607ms/step - loss: 1.8503 -
accuracy: 0.3562 - val_loss: 1.7292 - val_accuracy: 0.4479
Epoch 4/10
15/15 [==============================] - 9s 618ms/step - loss: 1.7085 -
accuracy: 0.3875 - val_loss: 1.6382 - val_accuracy: 0.4896
Epoch 5/10
15/15 [==============================] - 9s 611ms/step - loss: 1.6991 -
accuracy: 0.4104 - val_loss: 1.6675 - val_accuracy: 0.4896
Epoch 6/10
15/15 [==============================] - 9s 611ms/step - loss: 1.7425 -
accuracy: 0.3958 - val_loss: 1.6325 - val_accuracy: 0.4167
Epoch 7/10
15/15 [==============================] - 9s 609ms/step - loss: 1.5691 -
accuracy: 0.4625 - val_loss: 1.5348 - val_accuracy: 0.4479
Epoch 8/10
15/15 [==============================] - 9s 622ms/step - loss: 1.6595 -
accuracy: 0.4583 - val_loss: 1.4673 - val_accuracy: 0.4375
Epoch 9/10
15/15 [==============================] - 9s 613ms/step - loss: 1.6310 -
accuracy: 0.4729 - val_loss: 1.4118 - val_accuracy: 0.5312
Epoch 10/10
15/15 [==============================] - 9s 607ms/step - loss: 1.5848 -
accuracy: 0.4854 - val_loss: 1.5833 - val_accuracy: 0.4792
ResNet50V2 Accuracy and Loss plots
```




```
Fine-Tuned ResNet50V2 Training and Validation:
Epoch 1/10
15/15 [==============================] - 12s 685ms/step - loss: 1.6733 -
accuracy: 0.4250 - val_loss: 1.4860 - val_accuracy: 0.5312
Epoch 2/10
15/15 [==============================] - 9s 613ms/step - loss: 1.6144 -
accuracy: 0.4729 - val_loss: 1.5490 - val_accuracy: 0.4896
Epoch 3/10
15/15 [==============================] - 9s 610ms/step - loss: 1.6493 -
```

```
accuracy: 0.4729 - val_loss: 1.4706 - val_accuracy: 0.5208
Epoch 4/10
15/15 [==============================] - 9s 620ms/step - loss: 1.6783 -
accuracy: 0.4333 - val_loss: 1.5330 - val_accuracy: 0.4375
Epoch 5/10
15/15 [==============================] - 9s 611ms/step - loss: 1.6429 -
accuracy: 0.4313 - val_loss: 1.4272 - val_accuracy: 0.5000
Epoch 6/10
15/15 [==============================] - 9s 606ms/step - loss: 1.6235 -
accuracy: 0.4375 - val_loss: 1.4777 - val_accuracy: 0.4792
Epoch 7/10
15/15 [==============================] - 9s 604ms/step - loss: 1.6404 -
accuracy: 0.4667 - val_loss: 1.4978 - val_accuracy: 0.4896

Epoch 00007: ReduceLROnPlateau reducing learning rate to 0.00020000000949949026.
Epoch 8/10
15/15 [==============================] - 9s 626ms/step - loss: 1.5356 -
accuracy: 0.5042 - val_loss: 1.4719 - val_accuracy: 0.5104
Restoring model weights from the end of the best epoch.
Epoch 00008: early stopping
Fine-Tuned ResNet50V2 Accuracy and Loss plots
```




```
INFO:tensorflow:Assets written to: models/ResNet50V2exp3/assets
```

```
/home/jay/Temple/apps/envs/env/lib/python3.9/site-
packages/keras/utils/generic_utils.py:494: CustomMaskWarning: Custom mask layers
require a config and must override get_config. When loading, the custom mask
layer must be passed to the custom_objects argument.
  warnings.warn('Custom mask layers require a config and must override '
```

1.5 MobileNETV2

[ ]:
```
# model_name = PRETRAINED_MODELS[1]
# model = train_and_validate_model(model_name = model_name,␣
 ↪train_generator=sub_sampled_train_gen,␣
 ↪valid_generator=sub_sampled_valid_gen, num_classes=len(sub_samples))
```

```
# # for now saving resnetv2 as best model
# model.save('models/'+ model_name + 'exp3')
```

1.6 VGG16

[ ]:
```
# model_name = PRETRAINED_MODELS[2]
# model = train_and_validate_model(model_name = model_name,␣
 ↪train_generator=sub_sampled_train_gen,␣
 ↪valid_generator=sub_sampled_valid_gen,num_classes=len(sub_samples))
# # for now saving resnetv2 as best model
# model.save('models/'+ model_name + 'exp3')
```

1.6.1 Experiment 4: Randoming reducing multiple lables to a single label for an image where multiple lables exist

[22]:
```
import random
reduced_nih_xrays_train_df = nih_xrays_train_df
reduced_nih_xrays_valid_df = nih_xrays_valid_df
reduced_nih_xrays_train_df['finding_label'] =␣
 ↪reduced_nih_xrays_train_df['finding_label'].map( lambda x : random.choice(x.
 ↪split('|')) )
reduced_nih_xrays_valid_df['finding_label'] =␣
 ↪reduced_nih_xrays_valid_df['finding_label'].map( lambda x : random.choice(x.
 ↪split('|')) )
reduced_sampled_train_gen = train_model.
 ↪get_image_data_generator(reduced_nih_xrays_train_df,batch_size=BATCH_SIZE,image_size=IMAGE_
reduced_sampled_valid_gen = train_model.
 ↪get_image_data_generator(reduced_nih_xrays_valid_df,batch_size=BATCH_SIZE,image_size=IMAGE_
```

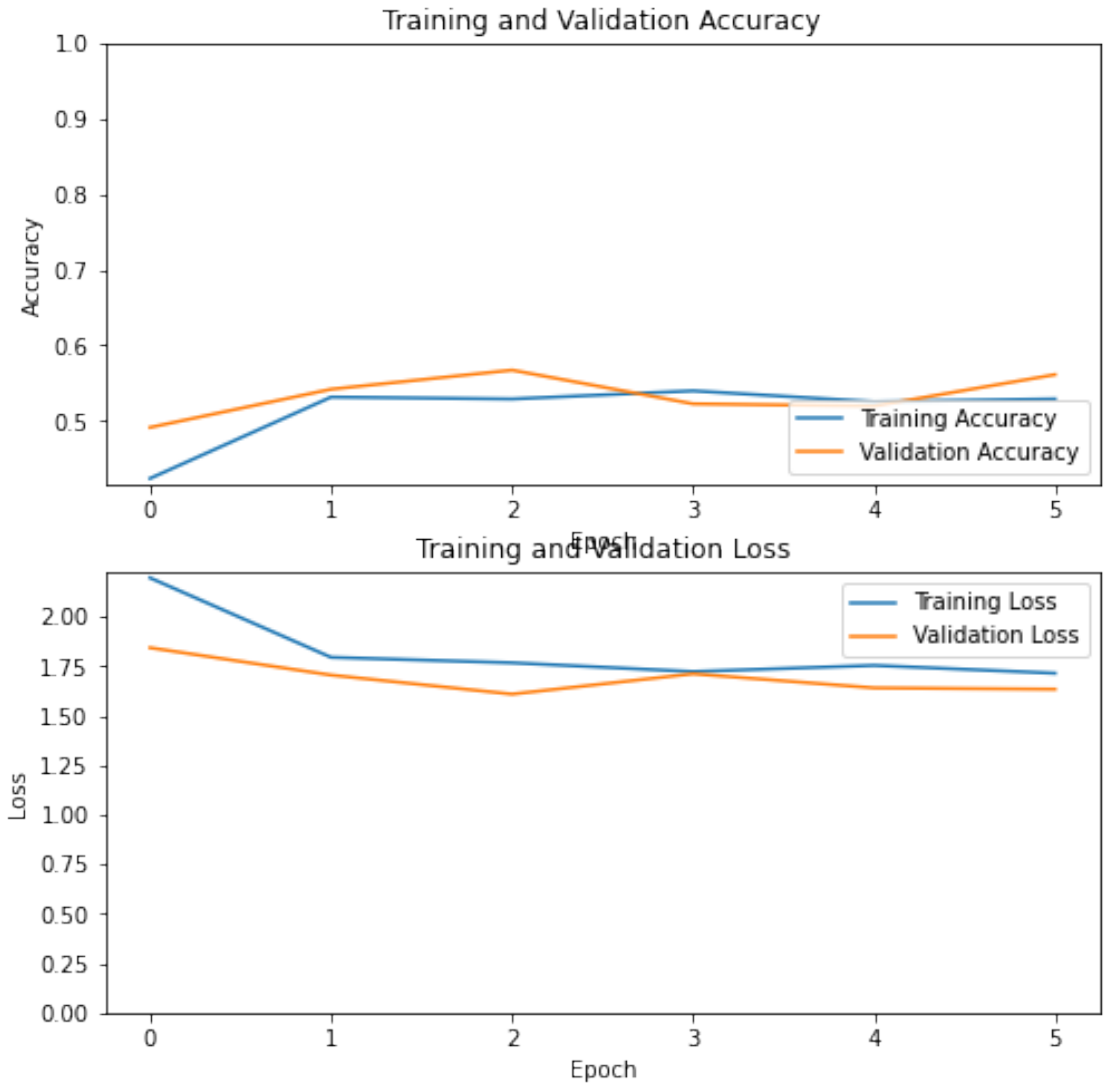
```
Found 89859 validated image filenames belonging to 15 classes.
Found 22245 validated image filenames belonging to 15 classes.
```

1.6.2 ResNet50V2

[23]:
```
model_name = PRETRAINED_MODELS[0]
model = train_and_validate_model(model_name = model_name,␣
 ↪train_generator=reduced_sampled_train_gen,␣
 ↪valid_generator=reduced_sampled_valid_gen)
# for now saving resnetv2 as best model
model.save('models/'+ model_name + 'exp4')
```

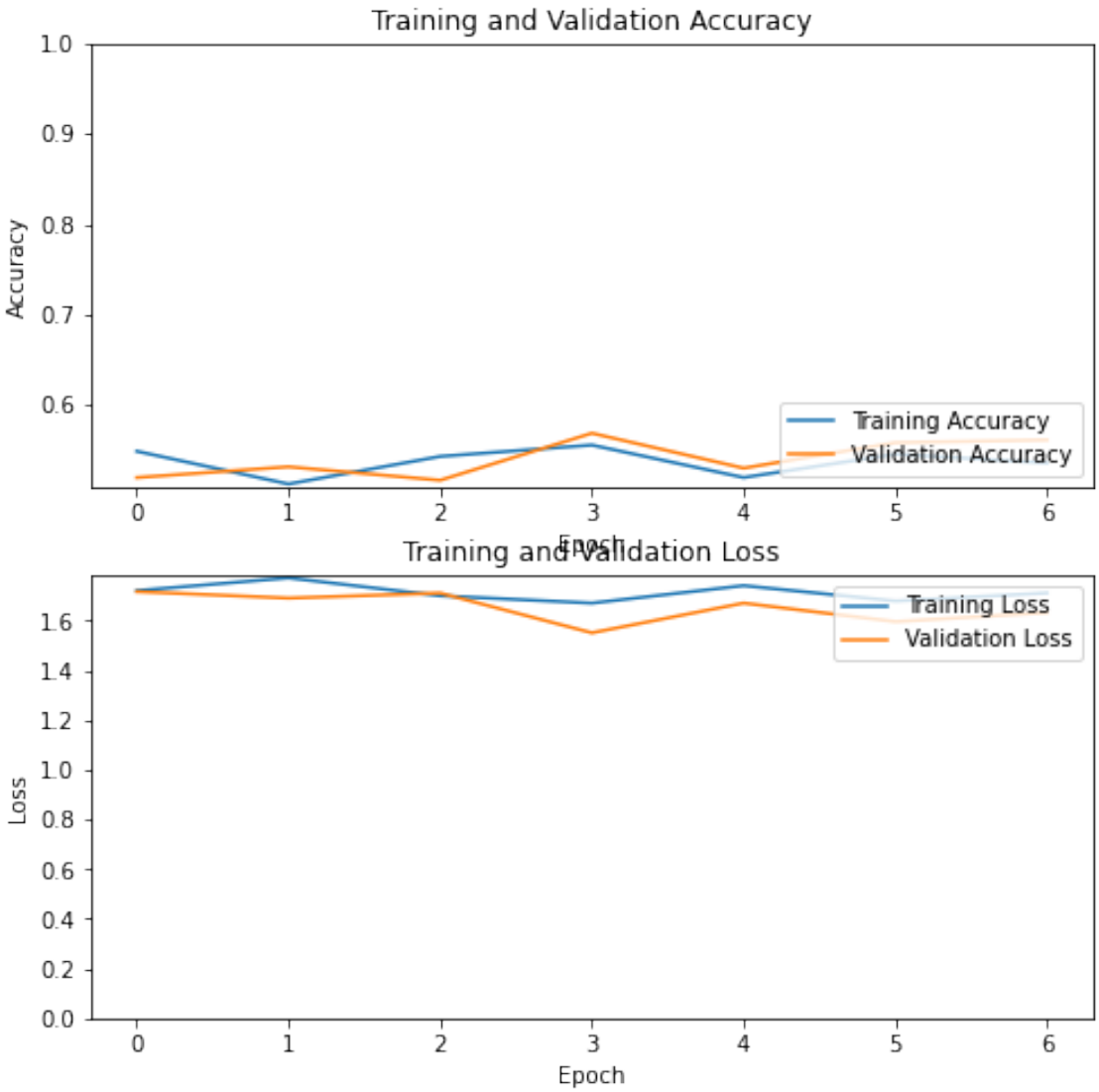
```
ResNet50V2
learning rate 0.01
Downloading ResNet50V2
Epoch 1/10
```

```
87/87 [==============================] - 55s 607ms/step - loss: 2.1965 -
accuracy: 0.4235 - val_loss: 1.8435 - val_accuracy: 0.4911
Epoch 2/10
87/87 [==============================] - 52s 598ms/step - loss: 1.7951 -
accuracy: 0.5312 - val_loss: 1.7055 - val_accuracy: 0.5417
Epoch 3/10
87/87 [==============================] - 52s 595ms/step - loss: 1.7675 -
accuracy: 0.5287 - val_loss: 1.6093 - val_accuracy: 0.5670
Epoch 4/10
87/87 [==============================] - 53s 610ms/step - loss: 1.7230 -
accuracy: 0.5395 - val_loss: 1.7119 - val_accuracy: 0.5223
Epoch 5/10
87/87 [==============================] - 52s 597ms/step - loss: 1.7543 -
accuracy: 0.5251 - val_loss: 1.6407 - val_accuracy: 0.5193

Epoch 00005: ReduceLROnPlateau reducing learning rate to 0.0019999999552965165.
Epoch 6/10
87/87 [==============================] - 52s 600ms/step - loss: 1.7149 -
accuracy: 0.5287 - val_loss: 1.6341 - val_accuracy: 0.5610
Restoring model weights from the end of the best epoch.
Epoch 00006: early stopping
ResNet50V2 Accuracy and Loss plots
```




```
Fine-Tuned ResNet50V2 Training and Validation:
Epoch 1/10
87/87 [==============================] - 55s 608ms/step - loss: 1.7213 -
accuracy: 0.5485 - val_loss: 1.7174 - val_accuracy: 0.5193
Epoch 2/10
87/87 [==============================] - 53s 604ms/step - loss: 1.7726 -
accuracy: 0.5122 - val_loss: 1.6912 - val_accuracy: 0.5312
Epoch 3/10
87/87 [==============================] - 52s 602ms/step - loss: 1.7007 -
accuracy: 0.5427 - val_loss: 1.7115 - val_accuracy: 0.5164
```

```
Epoch 4/10
87/87 [==============================] - 52s 597ms/step - loss: 1.6712 -
accuracy: 0.5553 - val_loss: 1.5516 - val_accuracy: 0.5685
Epoch 5/10
87/87 [==============================] - 52s 598ms/step - loss: 1.7415 -
accuracy: 0.5194 - val_loss: 1.6712 - val_accuracy: 0.5298
Epoch 6/10
87/87 [==============================] - 52s 595ms/step - loss: 1.6796 -
accuracy: 0.5452 - val_loss: 1.5966 - val_accuracy: 0.5580

Epoch 00006: ReduceLROnPlateau reducing learning rate to 0.00020000000949949026.
Epoch 7/10
87/87 [==============================] - 52s 599ms/step - loss: 1.7118 -
accuracy: 0.5356 - val_loss: 1.6326 - val_accuracy: 0.5610
Restoring model weights from the end of the best epoch.
Epoch 00007: early stopping
Fine-Tuned ResNet50V2 Accuracy and Loss plots
```




```
INFO:tensorflow:Assets written to: models/ResNet50V2exp4/assets
```

```
/home/jay/Temple/apps/envs/env/lib/python3.9/site-
packages/keras/utils/generic_utils.py:494: CustomMaskWarning: Custom mask layers
require a config and must override get_config. When loading, the custom mask
layer must be passed to the custom_objects argument.
  warnings.warn('Custom mask layers require a config and must override '
```

1.7 MobileNETV2

[ ]:
```
# model_name = PRETRAINED_MODELS[1]
# model = train_and_validate_model(model_name = model_name,␣
 ↪train_generator=reduced_sampled_train_gen,␣
 ↪valid_generator=reduced_sampled_valid_gen)
```

```
# # for now saving resnetv2 as best model
# model.save('models/'+ model_name + 'exp4')
```

1.8 VGG16

[ ]:
```
# model_name = PRETRAINED_MODELS[2]
# model = train_and_validate_model(model_name = model_name,␣
 ↪train_generator=reduced_sampled_train_gen,␣
 ↪valid_generator=reduced_sampled_valid_gen)
# # for now saving resnetv2 as best model
# model.save('models/'+ model_name + 'exp4')
```

Run this cell from command prompt

jupyter-nbconvert –to pdf COVID-19-Image-Classification-phase1.ipynb

1.8.1 Clean UP

run this cell after completing execution of the notebook

[ ]:
```
# clear gpu memory
from numba import cuda
device = cuda.get_current_device()
device.reset()
```