COVID-19 X-ray Classification

1. Objective

Coronavirus disease (COVID-19) is an respiratory infectious disease caused by the SARS-CoV-2 virus.

Chest x-rays can be used in diagnosis and follow-up patients with COVID-19 pneumonia. Since the state-so-the-medical-staff, extremely diagnosis of COVID-19 from chest x-ray carries up personnel needs.

```
# Import the API
import os, sys, zipfile
import tensorflow as tf, keras, numpy as np
from keras.models import Sequential
from keras.layers import *
from keras.utils import *
from keras.preprocessing.image import ImageGenerator
from keras.preprocessing.image.dirpy2_iterator import DirpyIterator
from keras.callbacks import ModelCheckpoint, EarlyStopping, ReduceLROnPlateau, CsvLogger
from keras.callbacks.tensorboard_logs.tensorboard_callbacks.TensorBoard, LearningRateScheduler, ReduceLROnPlateau
from keras.callbacks.tensorboard_logs.tensorboard_callbacks.TensorBoard, LearningRateScheduler, ReduceLROnPlateau, CsvLogger
from keras.callbacks.tensorboard_logs.tensorboard_callbacks.TensorBoard, LearningRateScheduler, ReduceLROnPlateau, CsvLogger, ModelCheckpoint
```

```
# Define of initial image to training directory.
# Define of Image image to training directory.
# Define of Training image to training directory.
```

```
number of initial image to training directory: 100
number of Image image to training directory: 100
number of Training image to training directory: 100
```

```
# get list of image dirs
# get list of image dirs
# get list of image dirs
# image_dirs = os.listdir(initial_image_dir)
# image_dirs = os.listdir(Image_image_dir)
# image_dirs = os.listdir(Training_image_dir)

# from image import ImageFilter image file to training list
# from image import ImageFilter image file to training list
# from image import ImageFilter image file to training list

# print("Number of initial image to training directory: ", initial_image_dirs)
# print("Number of Image image to training directory: ", Image_image_dirs)
# print("Number of Training image to training directory: ", Training_image_dirs)
```

```
images image file to training list: 1000, 1000
Training image file to training list: 1000, 1000
```

```
# Define of image to training directory.
# Define of Image image to training directory.
# Define of Training image to training directory.

# image_dir = "./initial_image"
# image_dir = "./Image"
# image_dir = "./Training"

# Define of image to training directory.
# Define of Image image to training directory.
# Define of Training image to training directory.

# initial_image_dir = os.path.join(os.path.dirname(__file__), "initial_image_dir")
# Image_image_dir = os.path.join(os.path.dirname(__file__), "Image_image_dir")
# Training_image_dir = os.path.join(os.path.dirname(__file__), "Training_image_dir")
```

```
number of initial image to training directory: 100
number of Image image to training directory: 100
number of Training image to training directory: 100
```

Data Augmentation

To enlarge a small augmentation using image data with 1 except positive image from 3 to 1 (1 zoom to 0.5, 0.8 by rotation by 0.5, 0.8 by shifting by 0.5, 0.8 by shear by 0.5) is incorporated.

For the image shown, the resulting is created.

```
# create image data generator for augmentation
# ImageDataGenerator = ImageDataGenerator(rotation_range=0.5,
                                          zoom_range=0.5,
                                          shear_range=0.5,
                                          width_shift_range=0.5,
                                          height_shift_range=0.5,
                                          horizontal_flip=True)

# train_data_generator = ImageDataGenerator(rescale=1/255)

# create augmented images with training to original image with target
# image_size = (256, 256)
# train_data = train_data_generator.flow_from_directory(image_dir, target_size=image_size, batch_size=32, class_mode='categorical')
# test_data = test_data_generator.flow_from_directory(image_dir, target_size=image_size, batch_size=32, class_mode='categorical')

# print(train_data.class_indices)
```

```
Found 2000 images belonging to 2 classes.
Found 2000 images belonging to 2 classes.
{'initial_image': 0, 'Image': 1}
```

```
# import augmentation train data
# print(train_data.image_shape, train_data.label_of_classes, classes)
# print("Number of initial image to training directory: ", initial_image_dir)
# print(test_data)
```

```
(256, 256, 3) [0 1] 2
Number of initial image to training directory:  ./initial_image
<DirpyIterator: mode='categorical', batch_size=32, image_shape=(256, 256), classes=2, shuffle=True, dtype=float32>
```

```
# create image data generator for augmentation
# ImageDataGenerator = ImageDataGenerator(rotation_range=0.5,
                                          zoom_range=0.5,
                                          shear_range=0.5,
                                          width_shift_range=0.5,
                                          height_shift_range=0.5,
                                          horizontal_flip=True)

# train_data_generator = ImageDataGenerator(rescale=1/255)

# create augmented images with training to original image with target
# image_size = (256, 256)
# train_data = train_data_generator.flow_from_directory(image_dir, target_size=image_size, batch_size=32, class_mode='categorical')
# test_data = test_data_generator.flow_from_directory(image_dir, target_size=image_size, batch_size=32, class_mode='categorical')

# print(train_data.class_indices)
```

```
Found 2000 images belonging to 2 classes.
Found 2000 images belonging to 2 classes.
{'initial_image': 0, 'Image': 1}
```

```
# import augmentation train data
# print(train_data.image_shape, train_data.label_of_classes, classes)
# print("Number of initial image to training directory: ", initial_image_dir)
# print(test_data)
```

```
(256, 256, 3) [0 1] 2
Number of initial image to training directory:  ./initial_image
<DirpyIterator: mode='categorical', batch_size=32, image_shape=(256, 256), classes=2, shuffle=True, dtype=float32>
```

```
# Define of image to training directory.
# Define of Image image to training directory.
# Define of Training image to training directory.

# image_dir = "./initial_image"
# image_dir = "./Image"
# image_dir = "./Training"

# Define of image to training directory.
# Define of Image image to training directory.
# Define of Training image to training directory.

# initial_image_dir = os.path.join(os.path.dirname(__file__), "initial_image_dir")
# Image_image_dir = os.path.join(os.path.dirname(__file__), "Image_image_dir")
# Training_image_dir = os.path.join(os.path.dirname(__file__), "Training_image_dir")
```

```
number of initial image to training directory: 100
number of Image image to training directory: 100
number of Training image to training directory: 100
```

Deep Learning Modeling

- Create an model
- Compile the model
- Training model. Save model learning curve and history

```
# import tensorflow.keras.models ;
# from keras import Sequential, Dense, Conv2D, Flatten, Dropout, MaxPooling2D
# model = Sequential()
# model.add(Conv2D(32, (3,3), activation='relu', input_shape=(image_height, image_width, 3)))
# model.add(MaxPooling2D(2,2))
# model.add(Conv2D(64, (3,3), activation='relu'))
# model.add(MaxPooling2D(2,2))
# model.add(Conv2D(128, (3,3), activation='relu'))
# model.add(MaxPooling2D(2,2))
# model.add(Conv2D(128, (3,3), activation='relu'))
# model.add(MaxPooling2D(2,2))
# model.add(Flatten())
# model.add(Dropout(0.5))
# model.add(Dense(512, activation='relu'))
# model.add(Dense(1, activation='sigmoid'))

# model.summary()
```

```
# model.compile(loss='binary_crossentropy',
# optimizer=tf.keras.optimizers.RMSprop(lr=1e-4),
# metrics=['acc'])

# model.save('./initial_image.h5')
# model.save('./Image.h5')
# model.save('./Training.h5')
```

```
# model.fit(train_data,
# steps_per_epoch=100,
# epochs=30,
# validation_data=test_data,
# validation_steps=50,
# verbose=2)
```

```
2022-05-31 12:00:00.00000: I tensorflow/stream_executor/platform/default/dso_loader.cc:44] Successfully opened dynamic library libcuda.so.1
2022-05-31 12:00:00.00000: I tensorflow/stream_executor/cuda/cuda_diagnostic.cc:156] kernel driver does not appear to exist.
Error: Failed to load the native TensorFlow runtime.
See https://www.tensorflow.org/install/errors for some common reasons and solutions.  Include the entire stack trace
above this error message when asking for help.
2022-05-31 12:00:00.00000: I tensorflow/stream_executor/platform/default/dso_loader.cc:44] Successfully opened dynamic library libcudart.so.10.1
2022-05-31 12:00:00.00000: I tensorflow/stream_executor/platform/default/dso_loader.cc:44] Successfully opened dynamic library libcublas.so.10
2022-05-31 12:00:00.00000: I tensorflow/stream_executor/platform/default/dso_loader.cc:44] Successfully opened dynamic library libcublasLt.so.10
2022-05-31 12:00:00.00000: I tensorflow/stream_executor/platform/default/dso_loader.cc:44] Successfully opened dynamic library libcufft.so.10
2022-05-31 12:00:00.00000: I tensorflow/stream_executor/platform/default/dso_loader.cc:44] Successfully opened dynamic library libcurand.so.10
2022-05-31 12:00:00.00000: I tensorflow/stream_executor/platform/default/dso_loader.cc:44] Successfully opened dynamic library libcusolver.so.10
2022-05-31 12:00:00.00000: I tensorflow/stream_executor/platform/default/dso_loader.cc:44] Successfully opened dynamic library libcusparse.so.10
2022-05-31 12:00:00.00000: I tensorflow/stream_executor/platform/default/dso_loader.cc:44] Successfully opened dynamic library libcudnn.so.7
2022-05-31 12:00:00.00000: W tensorflow/stream_executor/platform/default/dso_loader.cc:60] Could not load dynamic library 'libnvinfer.so.6'; dlopen: libnvinfer.so.6: cannot open shared object file: No such file or directory
2022-05-31 12:00:00.00000: W tensorflow/stream_executor/platform/default/dso_loader.cc:60] Could not load dynamic library 'libnvinfer_plugin.so.6'; dlopen: libnvinfer_plugin.so.6: cannot open shared object file: No such file or directory
2022-05-31 12:00:00.00000: W tensorflow/core/platform/tensorflow_gpu.cc:[illegible]] Cannot dlopen some GPU libraries. Skipping registering GPU devices...
2022-05-31 12:00:00.00000: I tensorflow/stream_executor/platform/default/dso_loader.cc:44] Successfully opened dynamic library libcudart.so.10.1
2022-05-31 12:00:00.00000: I tensorflow/stream_executor/platform/default/dso_loader.cc:44] Successfully opened dynamic library libcublas.so.10
2022-05-31 12:00:00.00000: I tensorflow/stream_executor/platform/default/dso_loader.cc:44] Successfully opened dynamic library libcublasLt.so.10
2022-05-31 12:00:00.00000: I tensorflow/stream_executor/platform/default/dso_loader.cc:44] Successfully opened dynamic library libcufft.so.10
2022-05-31 12:00:00.00000: I tensorflow/stream_executor/platform/default/dso_loader.cc:44] Successfully opened dynamic library libcurand.so.10
2022-05-31 12:00:00.00000: I tensorflow/stream_executor/platform/default/dso_loader.cc:44] Successfully opened dynamic library libcusolver.so.10
2022-05-31 12:00:00.00000: I tensorflow/stream_executor/platform/default/dso_loader.cc:44] Successfully opened dynamic library libcusparse.so.10
2022-05-31 12:00:00.00000: I tensorflow/stream_executor/platform/default/dso_loader.cc:44] Successfully opened dynamic library libcudnn.so.7
2022-05-31 12:00:00.00000: W tensorflow/tensorflow/core/platform/profile_utils/cpu_utils.cc:128] Failed to find bogomips or clock in /proc/cpuinfo; cannot determine CPU frequency
2022-05-31 12:00:00.00000: W tensorflow/tensorflow/core/platform/tensorflow_gpu.cc:[illegible]] Can not parse GPU device ordinals from CUDA_VISIBLE_DEVICES "0"
2022-05-31 12:00:00.00000: I tensorflow/core/platform/cpu_feature_guard.cc:142] This TensorFlow binary was not compiled to use: AVX512F
2022-05-31 12:00:00.00000: I tensorflow/stream_executor/platform/default/dso_loader.cc:44] Successfully opened dynamic library libcuda.so.1
2022-05-31 12:00:00.00000: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:[illegible]] Ignoring visible gpu device (device: 0, name: Tesla K80, pci bus id: 0000:00:04.0) with Core Compute Capability 3.7 but Tensorflow requires one with at least [illegible]
2022-05-31 12:00:00.00000: I tensorflow/tensorflow/core/platform/profile_utils/cpu_utils.cc:94] CPU Frequency: [illegible] Hz
2022-05-31 12:00:00.00000: I tensorflow/tensorflow/core/grappler/clusters/single_machine.cc:356] Starting new session
2022-05-31 12:00:00.00000: W tensorflow/tensorflow/core/grappler/optimizers/data/vectorization/wrapped_batch_matmul.cc:[illegible]] Cannot batch 2-D RHS in [illegible]
```