Model Development Phase Template

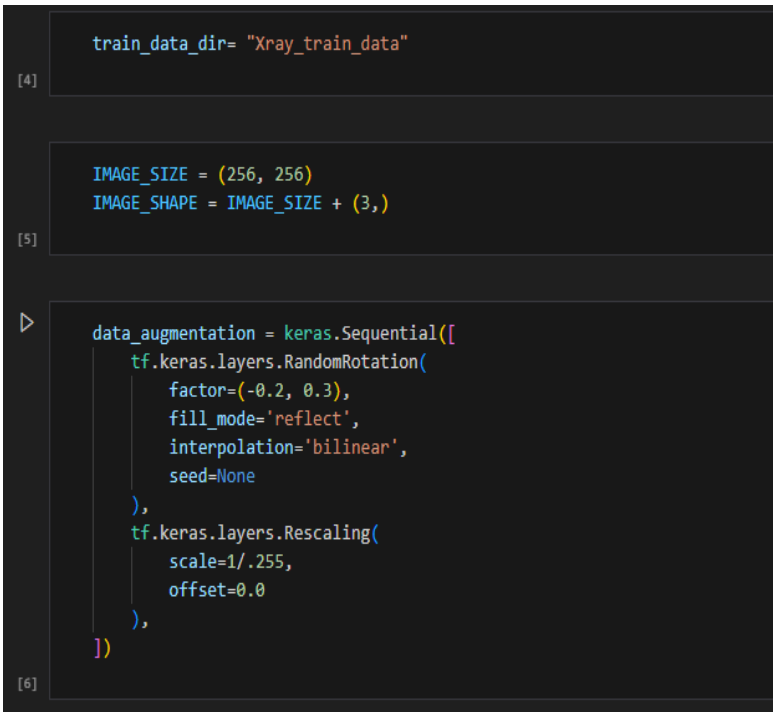
Date	11-07-2024
Team ID	SWTID1720433291
Project Title	CovidVision: Advanced COVID-19 Detection From Lung X-Rays With Deep Learning
Maximum Marks	10 Marks

Initial Model Training Code, Model Validation and Evaluation Report

The initial model training code will be showcased in the future through a screenshot. The model validation and evaluation report will include a summary and training and validation performance metrics for multiple models, presented through respective screenshots.

Initial Model Training Code (5 marks):

CNN:



```
[4]
train_data_dir= "Xray_train_data"
```

```
[5]
IMAGE_SIZE = (256, 256)
IMAGE_SHAPE = IMAGE_SIZE + (3,)
```

```
[6]
data_augmentation = keras.Sequential([
    tf.keras.layers.RandomRotation(
        factor=(-0.2, 0.3),
        fill_mode='reflect',
        interpolation='bilinear',
        seed=None
    ),
    tf.keras.layers.Rescaling(
        scale=1/.255,
        offset=0.0
    ),
])
```

```
model=tf.keras.Sequential([
    data_augmentation,
    tf.keras.layers.Conv2D(8, (3,3), activation='relu', input_shape=IMAGE_SHAPE),
    tf.keras.layers.MaxPooling2D(pool_size=(2, 2), strides=None, padding="valid"),
    tf.keras.layers.Conv2D(16, (3,3), activation='relu'),
    tf.keras.layers.MaxPooling2D(pool_size=(2, 2), strides=None, padding="valid"),
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(32),
    tf.keras.layers.Dropout(.2, input_shape=(32,)),
    tf.keras.layers.Dense(1, activation='sigmoid')
])
```

[7]

```
from tensorflow.keras.optimizers import Adam

model.compile(optimizer=Adam(learning_rate=0.0001),
              loss='binary_crossentropy',
              metrics = ['accuracy'])
```

[42]

```
from tensorflow.keras.preprocessing.image import ImageDataGenerator
img_height, img_width=64,64
batch_size=16
train_datagen = ImageDataGenerator(validation_split=0.3) # set validation split

train_generator = train_datagen.flow_from_directory(
    train_data_dir,
    target_size=(img_height, img_width),
    batch_size=batch_size,
    class_mode='binary',
    subset='training') # set as training data
# Splitting images for validation set
validation_generator = train_datagen.flow_from_directory(
    train_data_dir, # same directory as training data
    target_size=(img_height, img_width),
    batch_size=batch_size,
    class_mode='binary',
    subset='validation') # set as validation data
```

[43]

```
Found 14729 images belonging to 2 classes.
Found 6311 images belonging to 2 classes.
```

XCEPTION:

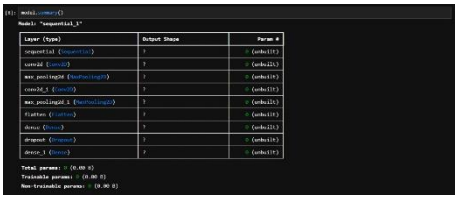
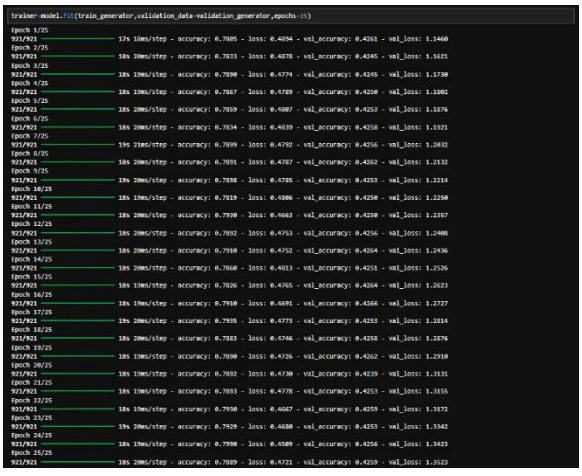
```
train_datagen = ImageDataGenerator(preprocessing_function=tf.keras.applications.xception.preprocess_input,zoom_range=0.1,brightness_range=[0.5,1.3],
                                   width_shift_range=0.1,height_shift_range=0.1,validation_split=0.1)
test_datagen=ImageDataGenerator(preprocessing_function=tf.keras.applications.xception.preprocess_input)
```

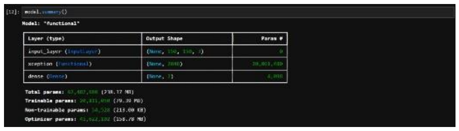
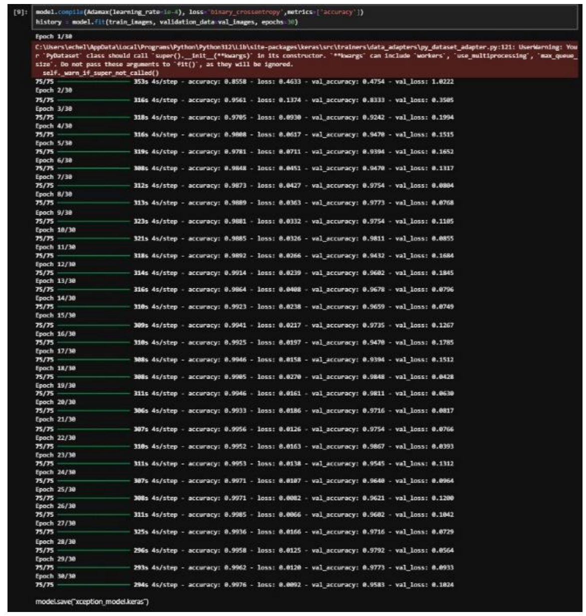
```
BATCH_SIZE=64
path="Coronahack-Chest-XRay-Dataset/Coronahack-Chest-XRay-Dataset/"
train_images=train_datagen.flow_from_dataframe(dataframe=df_meta[df_meta["Dataset_type"]=="TRAIN"],x_col='X_ray_image_name',y_col='Label',
                                               color_mode='rgb', class_mode='categorical',target_size=(150,150),
                                               batch_size=BATCH_SIZE,shuffle=True,seed=1234,subset='training',
                                               directory=path+"/train")

val_images=train_datagen.flow_from_dataframe(dataframe=df_meta[df_meta["Dataset_type"]=="TRAIN"],x_col='X_ray_image_name',y_col='Label',color_mode='rgb',
                                             class_mode='categorical',target_size=(150,150),batch_size=BATCH_SIZE,shuffle=True,seed=1234,
                                             subset='validation', directory=path+"/train")

test_images = test_datagen.flow_from_dataframe(dataframe=df_meta[df_meta["Dataset_type"]=="TEST"],x_col='X_ray_image_name',y_col='Label',
                                               color_mode='rgb',class_mode='categorical',target_size=(150,150),batch_size=BATCH_SIZE,shuffle=False,
                                               directory=path+"/test")
```

```
inputs = tf.keras.layers.Input((150,150,3))
base_model=tf.keras.applications.xception.Xception(include_top=False, weights="imagenet",input_shape=(150,150,3), pooling='avg')
x=base_model(inputs)
output=layers.Dense(2, activation='sigmoid')(x)
model=tf.keras.models.Model(inputs=inputs, outputs=output)
```

```
model.compile(Adamax(learning_rate=1e-4), loss='binary_crossentropy',metrics=['accuracy'])
history = model.fit(train_images, validation_data=val_images, epochs=30)
```

Model Validation and Evaluation Report (5 marks):

Model	Summary	Training and Validation Performance Metrics
CNN		

XCEPTION		

```
[12]: model.summary()

Model: "functional"

| Layer (type)               | Output Shape          | Param #    |
| input_layer (InputLayer)   | (None, 150, 150, 3)   | 0          |
| xception (Functional)      | (None, 2048)          | 20,861,480 |
| dense (Dense)              | (None, 2)             | 4,098      |

Total params: [illegible] (238.17 MB)
Trainable params: [illegible] (79.39 MB)
Non-trainable params: 54,528 (213.00 KB)
Optimizer params: [illegible] (158.78 MB)
```

```
[9]: model.compile(Adamax(learning_rate=1e-4), loss='binary_crossentropy',metrics=['accuracy'])
history = model.fit(train_images, validation_data=val_images, epochs=30)

Epoch 1/30
C:\Users\echel\AppData\Local\Programs\Python\Python312\Lib\site-packages\keras\src\trainers\data_adapters\py_dataset_adapter.py:121: UserWarning: Your `PyDataset` class should call `super().__init__(**kwargs)` in its constructor. `**kwargs` can include `workers`, `use_multiprocessing`, `max_queue_size`. Do not pass these arguments to `fit()`, as they will be ignored.
  self._warn_if_super_not_called()
75/75 ━━━━━━━━━━━━━━━━━━━━ 353s 4s/step - accuracy: 0.8558 - loss: 0.4633 - val_accuracy: 0.4754 - val_loss: 1.0222
Epoch 2/30
75/75 ━━━━━━━━━━━━━━━━━━━━ 316s 4s/step - accuracy: 0.9561 - loss: 0.1374 - val_accuracy: 0.8333 - val_loss: 0.3505
Epoch 3/30
75/75 ━━━━━━━━━━━━━━━━━━━━ 318s 4s/step - accuracy: 0.9705 - loss: 0.0930 - val_accuracy: 0.9242 - val_loss: 0.1994
Epoch 4/30
75/75 ━━━━━━━━━━━━━━━━━━━━ 316s 4s/step - accuracy: 0.9808 - loss: 0.0617 - val_accuracy: 0.9470 - val_loss: 0.1515
Epoch 5/30
75/75 ━━━━━━━━━━━━━━━━━━━━ 319s 4s/step - accuracy: 0.9781 - loss: 0.0711 - val_accuracy: 0.9394 - val_loss: 0.1652
Epoch 6/30
75/75 ━━━━━━━━━━━━━━━━━━━━ 308s 4s/step - accuracy: 0.9848 - loss: 0.0451 - val_accuracy: 0.9470 - val_loss: 0.1317
Epoch 7/30
75/75 ━━━━━━━━━━━━━━━━━━━━ 312s 4s/step - accuracy: 0.9873 - loss: 0.0427 - val_accuracy: 0.9754 - val_loss: 0.0804
Epoch 8/30
75/75 ━━━━━━━━━━━━━━━━━━━━ 313s 4s/step - accuracy: 0.9889 - loss: 0.0363 - val_accuracy: 0.9773 - val_loss: 0.0768
Epoch 9/30
75/75 ━━━━━━━━━━━━━━━━━━━━ 323s 4s/step - accuracy: 0.9881 - loss: 0.0332 - val_accuracy: 0.9754 - val_loss: 0.1105
Epoch 10/30
75/75 ━━━━━━━━━━━━━━━━━━━━ 321s 4s/step - accuracy: 0.9885 - loss: 0.0326 - val_accuracy: 0.9811 - val_loss: 0.0855
Epoch 11/30
75/75 ━━━━━━━━━━━━━━━━━━━━ 318s 4s/step - accuracy: 0.9892 - loss: 0.0266 - val_accuracy: 0.9432 - val_loss: 0.1684
Epoch 12/30
75/75 ━━━━━━━━━━━━━━━━━━━━ 314s 4s/step - accuracy: 0.9914 - loss: 0.0239 - val_accuracy: 0.9602 - val_loss: 0.1845
Epoch 13/30
75/75 ━━━━━━━━━━━━━━━━━━━━ 316s 4s/step - accuracy: 0.9864 - loss: 0.0408 - val_accuracy: 0.9678 - val_loss: 0.0796
Epoch 14/30
75/75 ━━━━━━━━━━━━━━━━━━━━ 310s 4s/step - accuracy: 0.9923 - loss: 0.0238 - val_accuracy: 0.9659 - val_loss: 0.0749
Epoch 15/30
75/75 ━━━━━━━━━━━━━━━━━━━━ 309s 4s/step - accuracy: 0.9941 - loss: 0.0217 - val_accuracy: 0.9735 - val_loss: 0.1267
Epoch 16/30
75/75 ━━━━━━━━━━━━━━━━━━━━ 310s 4s/step - accuracy: 0.9925 - loss: 0.0197 - val_accuracy: 0.9470 - val_loss: 0.1785
Epoch 17/30
75/75 ━━━━━━━━━━━━━━━━━━━━ 308s 4s/step - accuracy: 0.9946 - loss: 0.0158 - val_accuracy: 0.9394 - val_loss: 0.1512
Epoch 18/30
75/75 ━━━━━━━━━━━━━━━━━━━━ 308s 4s/step - accuracy: 0.9905 - loss: 0.0270 - val_accuracy: 0.9848 - val_loss: 0.0428
Epoch 19/30
75/75 ━━━━━━━━━━━━━━━━━━━━ 311s 4s/step - accuracy: 0.9946 - loss: 0.0161 - val_accuracy: 0.9811 - val_loss: 0.0630
Epoch 20/30
75/75 ━━━━━━━━━━━━━━━━━━━━ 306s 4s/step - accuracy: 0.9933 - loss: 0.0186 - val_accuracy: 0.9716 - val_loss: 0.0817
Epoch 21/30
75/75 ━━━━━━━━━━━━━━━━━━━━ 307s 4s/step - accuracy: 0.9956 - loss: 0.0126 - val_accuracy: 0.9754 - val_loss: 0.0766
Epoch 22/30
75/75 ━━━━━━━━━━━━━━━━━━━━ 310s 4s/step - accuracy: 0.9952 - loss: 0.0163 - val_accuracy: 0.9867 - val_loss: 0.0393
Epoch 23/30
75/75 ━━━━━━━━━━━━━━━━━━━━ 311s 4s/step - accuracy: 0.9953 - loss: 0.0138 - val_accuracy: 0.9545 - val_loss: 0.1312
Epoch 24/30
75/75 ━━━━━━━━━━━━━━━━━━━━ 307s 4s/step - accuracy: 0.9971 - loss: 0.0107 - val_accuracy: 0.9640 - val_loss: 0.0964
Epoch 25/30
75/75 ━━━━━━━━━━━━━━━━━━━━ 308s 4s/step - accuracy: 0.9971 - loss: 0.0082 - val_accuracy: 0.9621 - val_loss: 0.1200
Epoch 26/30
75/75 ━━━━━━━━━━━━━━━━━━━━ 311s 4s/step - accuracy: 0.9985 - loss: 0.0066 - val_accuracy: 0.9602 - val_loss: 0.1042
Epoch 27/30
75/75 ━━━━━━━━━━━━━━━━━━━━ 325s 4s/step - accuracy: 0.9936 - loss: 0.0166 - val_accuracy: 0.9716 - val_loss: 0.0729
Epoch 28/30
75/75 ━━━━━━━━━━━━━━━━━━━━ 296s 4s/step - accuracy: 0.9958 - loss: 0.0125 - val_accuracy: 0.9792 - val_loss: 0.0564
Epoch 29/30
75/75 ━━━━━━━━━━━━━━━━━━━━ 293s 4s/step - accuracy: 0.9962 - loss: 0.0120 - val_accuracy: 0.9773 - val_loss: 0.0933
Epoch 30/30
75/75 ━━━━━━━━━━━━━━━━━━━━ 294s 4s/step - accuracy: 0.9976 - loss: 0.0092 - val_accuracy: 0.9583 - val_loss: 0.1024

model.save("xception_model.keras")
```