Skill-10

2100030910

Sec-23

Main.py

```
import numpy as np
import pandas as pd
import os
import model6 as mc

train_directory =
r'C:\Users\dell\PycharmProjex\deepLearn\Skill\genData\train'
valid_directory =
r'C:\Users\dell\PycharmProjex\deepLearn\Skill\genData\valid'

cloud_directory =
r'C:\Users\dell\PycharmProjex\deepLearn\Skill\data\cloudy'
cloud_train_directory =
r'C:\Users\dell\PycharmProjex\deepLearn\Skill\genData\train\cloudy'
cloud_valid_directory =
r'C:\Users\dell\PycharmProjex\deepLearn\Skill\genData\valid\cloudy'

water_directory = r'C:\Users\dell\PycharmProjex\deepLearn\Skill\data\water'
water_train_directory =
r'C:\Users\dell\PycharmProjex\deepLearn\Skill\genData\train\water'
water_valid_directory =
r'C:\Users\dell\PycharmProjex\deepLearn\Skill\genData\valid\water'

green_directory =
r'C:\Users\dell\PycharmProjex\deepLearn\Skill\data\green_area'
green_train_directory =
r'C:\Users\dell\PycharmProjex\deepLearn\Skill\genData\train\green'
green_valid_directory =
r'C:\Users\dell\PycharmProjex\deepLearn\Skill\genData\valid\green'

desert_directory =
r'C:\Users\dell\PycharmProjex\deepLearn\Skill\data\desert'
desert_train_directory =
r'C:\Users\dell\PycharmProjex\deepLearn\Skill\genData\train\desert'
desert_valid_directory =
r'C:\Users\dell\PycharmProjex\deepLearn\Skill\genData\valid\desert'

cloud_image_files = [f for f in os.listdir(cloud_directory) if
                     f.lower().endswith(('.jpg', '.jpeg'))]


water_image_files = [f for f in os.listdir(water_directory) if
                     f.lower().endswith(('.jpg', '.jpeg'))]
```

```
desert_image_files = [f for f in os.listdir(desert_directory) if
                      f.lower().endswith(('.jpg', '.jpeg'))]


green_image_files = [f for f in os.listdir(green_directory) if
                     f.lower().endswith(('.jpg', '.jpeg'))]



os.makedirs(train_directory, exist_ok=True)
os.makedirs(valid_directory, exist_ok=True)


import os
import numpy as np
import matplotlib.pyplot as plt
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense,
BatchNormalization, Dropout

# Directories
train_directory =
r'C:\Users\dell\PycharmProjex\deepLearn\Skill\genData\train'
valid_directory =
r'C:\Users\dell\PycharmProjex\deepLearn\Skill\genData\valid'

# Image data preprocessing
train_datagen = ImageDataGenerator(
    rescale=1.0 / 255,
    rotation_range=40,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True,
    fill_mode='nearest'
)

validation_datagen = ImageDataGenerator(rescale=1.0 / 255)

batch_size = 32

train_generator = train_datagen.flow_from_directory(
    train_directory,
    target_size=(224, 224),
    batch_size=batch_size,
    class_mode='categorical',
    shuffle=True
)

validation_generator = validation_datagen.flow_from_directory(
    valid_directory,
    target_size=(224, 224),
    batch_size=batch_size,
    class_mode='categorical',
    shuffle=False
)

# Sample CNN model
```

```
mc2=mc.Modelsc()
model2=mc2.rnn_model()

# Display model summary
model2.summary()

# Training the model
epochs = 4

history = model2.fit(
    train_generator,
    epochs=epochs,
    validation_data=validation_generator
)

# Plotting accuracy and loss curves
plt.figure(figsize=(12, 4))

plt.subplot(1, 2, 1)
plt.plot(history.history['accuracy'], label='Training Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
plt.title('Training and Validation Accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()

plt.subplot(1, 2, 2)
plt.plot(history.history['loss'], label='Training Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.title('Training and Validation Loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()

plt.show()
```

model.py

```
import tensorflow as tf
from keras import Sequential, Input, Model
from keras.src.applications import VGG16
from keras.src.layers import Flatten, Dense, Conv2D, MaxPool2D, Dropout,
SimpleRNN, Reshape
from tensorflow import keras
from tensorflow.keras import layers
from keras.applications.vgg16 import VGG16, preprocess_input
class Modelsc:
    def rnn_model(self):
        input_shape=(224,224,3)
        model = Sequential()
        model.add(Reshape((input_shape[0] * input_shape[1],
input_shape[2]), input_shape=input_shape))
        # Add SimpleRNN layer
        model.add(SimpleRNN(128))

        # Add fully connected layers
        model.add(Dense(4096, activation='relu'))
        model.add(Dense(1072, activation='relu'))
        model.add(Dropout(0.2))
        model.add(Dense(4, activation='softmax'))
        model.compile(optimizer='adam',
```

```
                      loss='categorical_crossentropy',
                      metrics=['accuracy'])
    return model
```

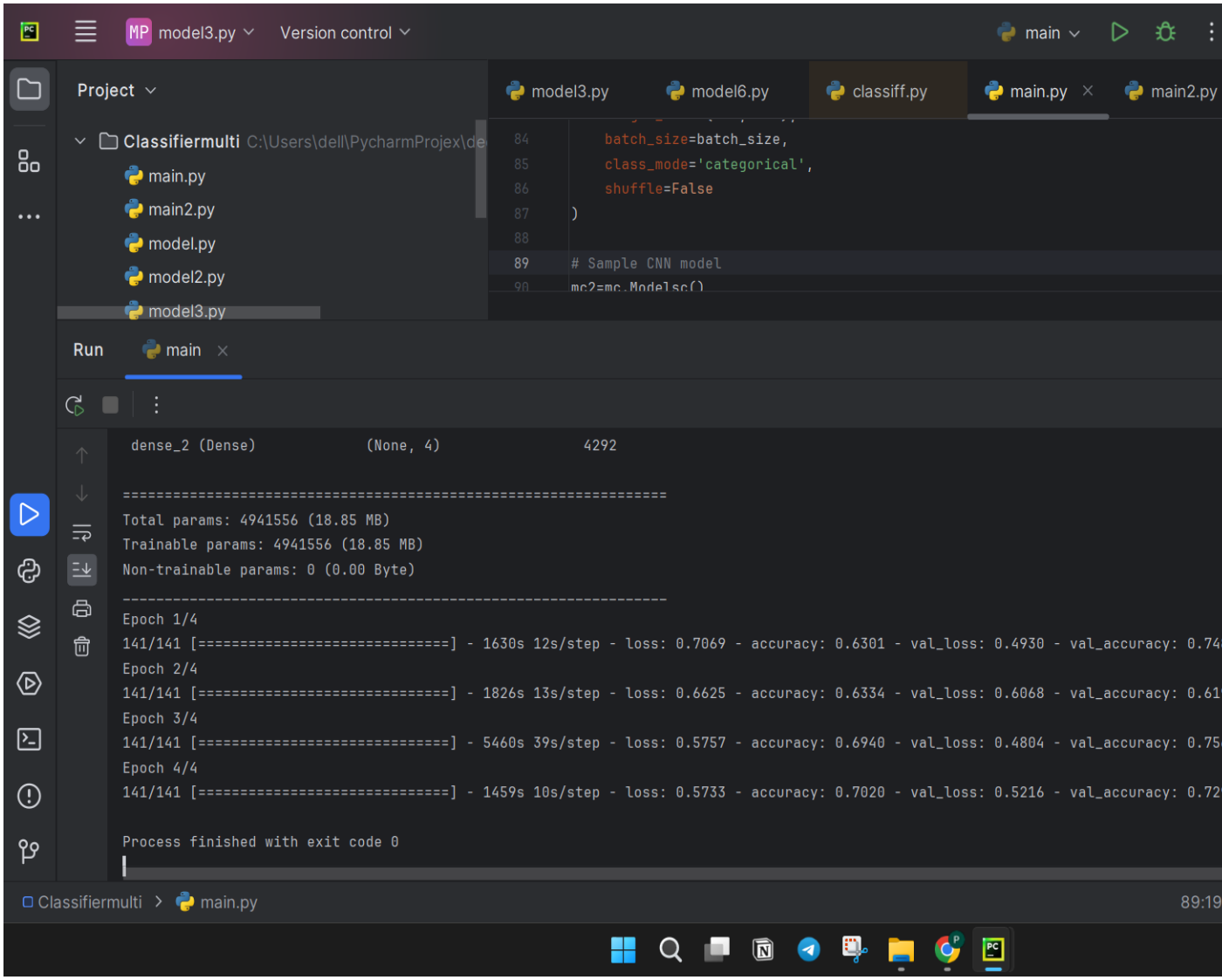
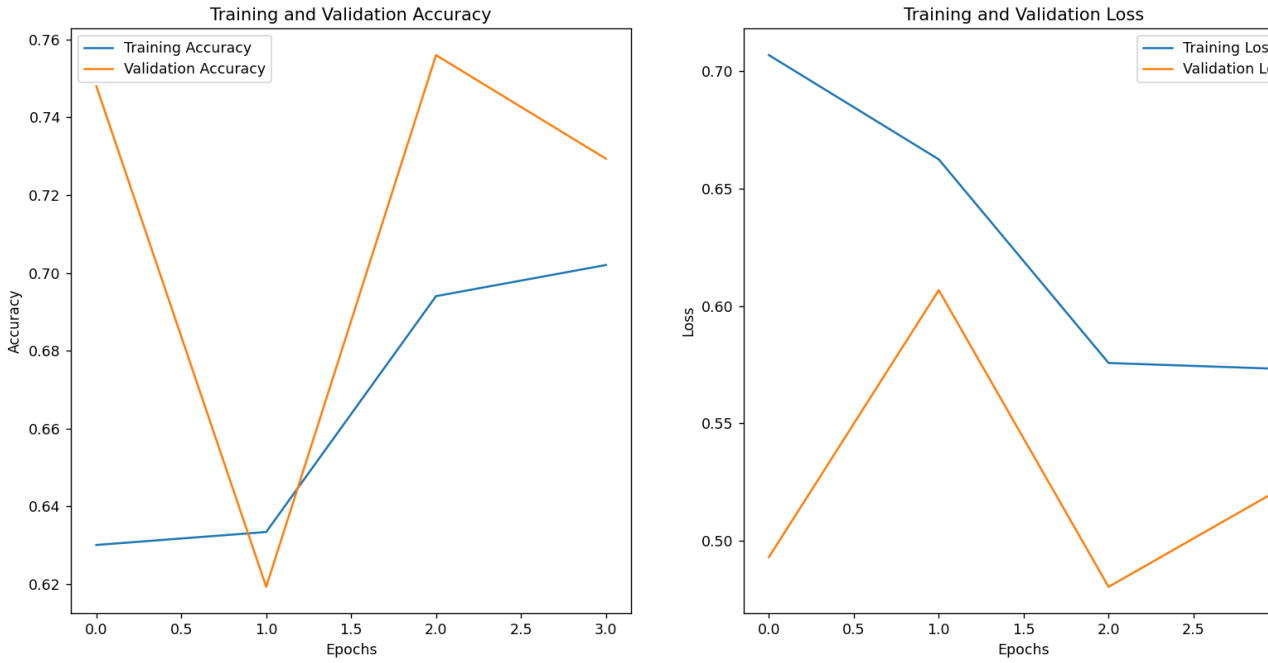
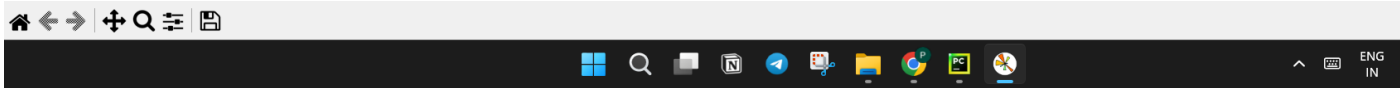


Figure 1





ENG
IN