Skill9

2100030910

Sec-23

Main.py

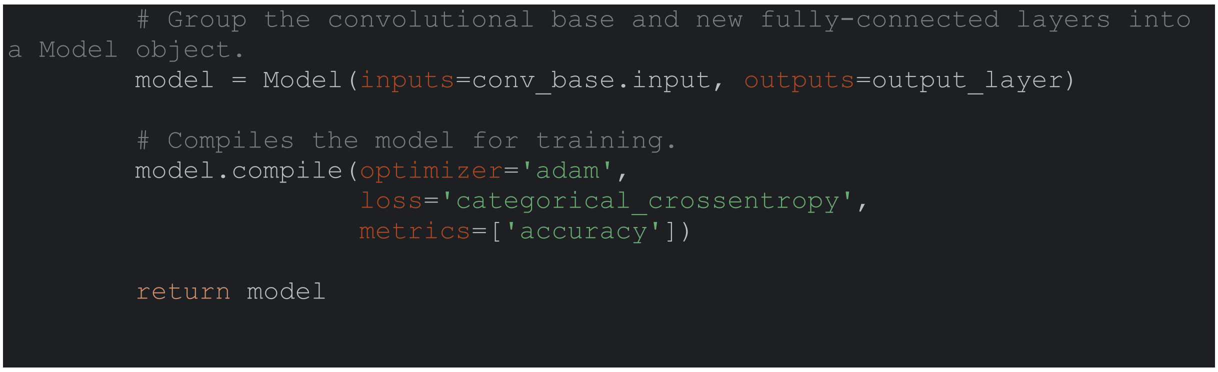
```
import numpy as np
import pandas as pd
import os
import model3 as mc

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

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

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

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

desert_directory = r'C:\Users\dell\PycharmProjex\dlSkill\Skill\data\desert'
desert_train_directory =
r'C:\Users\dell\PycharmProjex\dlSkill\Skill\genData\train\desert'
desert_valid_directory =
r'C:\Users\dell\PycharmProjex\dlSkill\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\dlSkill\Skill\genData\train'
valid_directory =
r'C:\Users\dell\PycharmProjex\dlSkill\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.cnn_vgg()

# Display model summary
model2.summary()
```

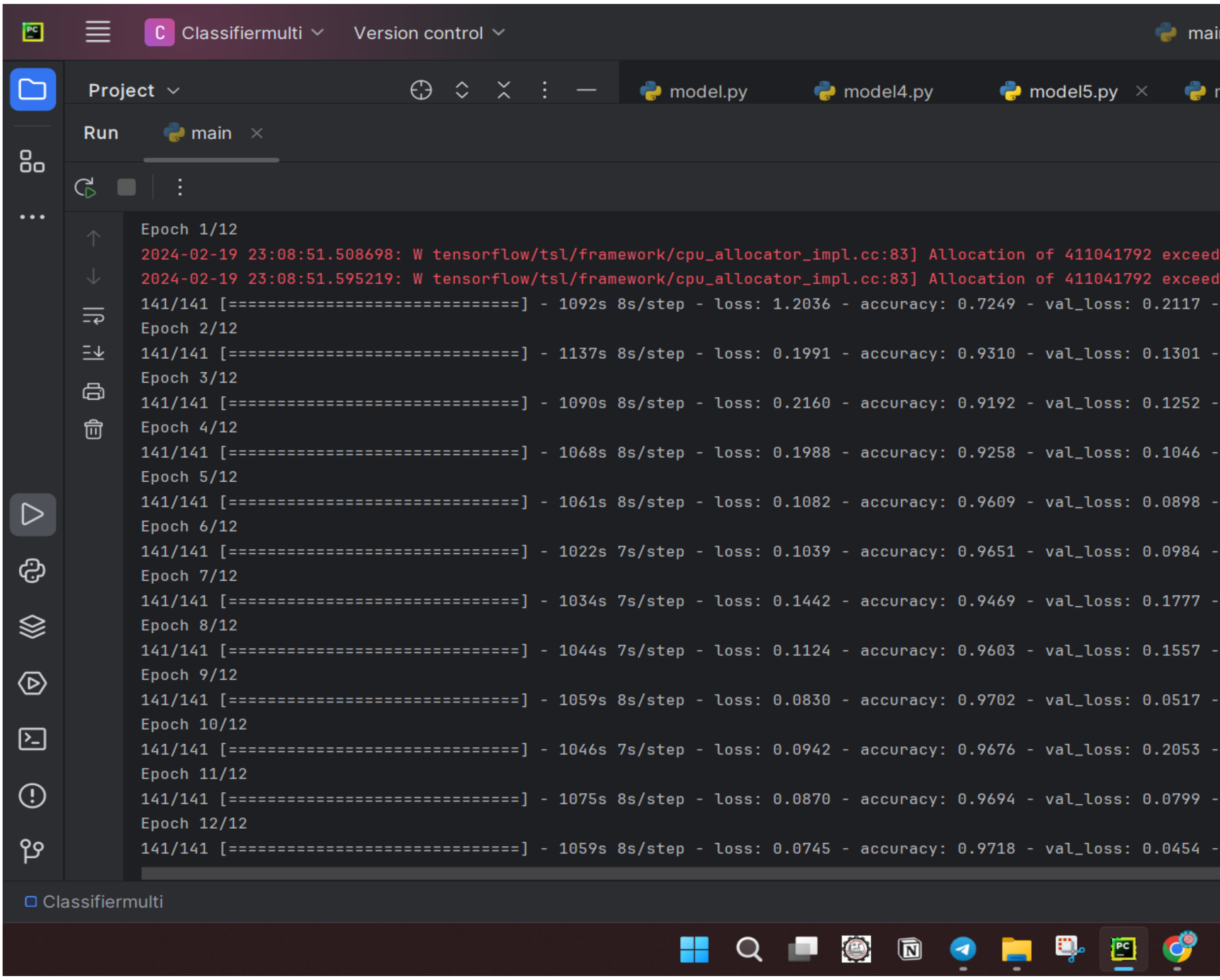
```
# Training the model
epochs = 12

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
from tensorflow import keras
from tensorflow.keras import layers
from keras.applications.vgg16 import VGG16, preprocess_input
class Modelsc:
    def cnn_vgg(self):

        conv_base = VGG16(include_top=False,
                          weights='imagenet',
                          input_shape=(224, 224, 3))


        for layer in conv_base.layers:
            layer.trainable = False
        # Create a new 'top' of the model (i.e. fully-connected layers).
        # This is 'bootstrapping' a new top_model onto the pretrained 
layers.
        top_model = conv_base.output
        top_model = Flatten(name="flatten")(top_model)
        top_model = Dense(4096, activation='relu')(top_model)
        top_model = Dense(1072, activation='relu')(top_model)
        top_model = Dropout(0.2)(top_model)
        output_layer = Dense(4, activation='softmax')(top_model)
```



```
    # Group the convolutional base and new fully-connected layers into
a Model object.
    model = Model(inputs=conv_base.input, outputs=output_layer)

    # Compiles the model for training.
    model.compile(optimizer='adam',
                  loss='categorical_crossentropy',
                  metrics=['accuracy'])

    return model
```



```
Epoch 1/12
2024-02-19 23:08:51.508698: W tensorflow/tsl/framework/cpu_allocator_impl.cc:83] Allocation of 411041792 exceed
2024-02-19 23:08:51.595219: W tensorflow/tsl/framework/cpu_allocator_impl.cc:83] Allocation of 411041792 exceed
141/141 [==============================] - 1092s 8s/step - loss: 1.2036 - accuracy: 0.7249 - val_loss: 0.2117 -
Epoch 2/12
141/141 [==============================] - 1137s 8s/step - loss: 0.1991 - accuracy: 0.9310 - val_loss: 0.1301 -
Epoch 3/12
141/141 [==============================] - 1090s 8s/step - loss: 0.2160 - accuracy: 0.9192 - val_loss: 0.1252 -
Epoch 4/12
141/141 [==============================] - 1068s 8s/step - loss: 0.1988 - accuracy: 0.9258 - val_loss: 0.1046 -
Epoch 5/12
141/141 [==============================] - 1061s 8s/step - loss: 0.1082 - accuracy: 0.9609 - val_loss: 0.0898 -
Epoch 6/12
141/141 [==============================] - 1022s 7s/step - loss: 0.1039 - accuracy: 0.9651 - val_loss: 0.0984 -
Epoch 7/12
141/141 [==============================] - 1034s 7s/step - loss: 0.1442 - accuracy: 0.9469 - val_loss: 0.1777 -
Epoch 8/12
141/141 [==============================] - 1044s 7s/step - loss: 0.1124 - accuracy: 0.9603 - val_loss: 0.1557 -
Epoch 9/12
141/141 [==============================] - 1059s 8s/step - loss: 0.0830 - accuracy: 0.9702 - val_loss: 0.0517 -
Epoch 10/12
141/141 [==============================] - 1046s 7s/step - loss: 0.0942 - accuracy: 0.9676 - val_loss: 0.2053 -
Epoch 11/12
141/141 [==============================] - 1075s 8s/step - loss: 0.0870 - accuracy: 0.9694 - val_loss: 0.0799 -
Epoch 12/12
141/141 [==============================] - 1059s 8s/step - loss: 0.0745 - accuracy: 0.9718 - val_loss: 0.0454 -
```

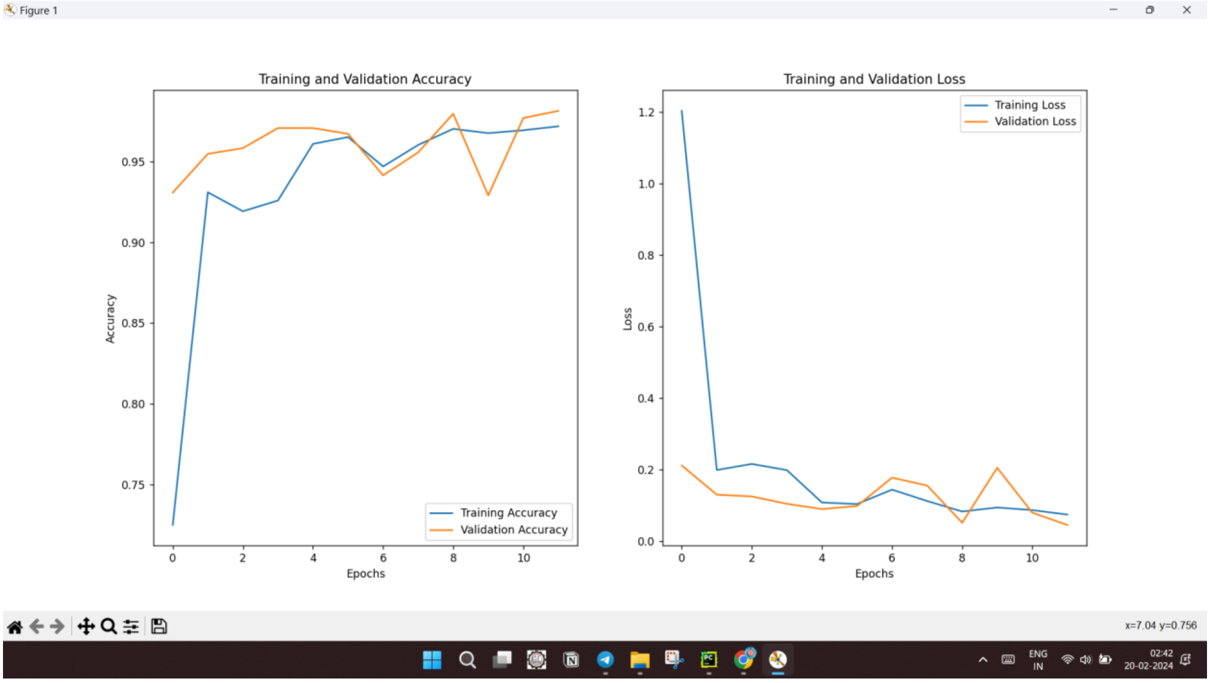


Figure 1
Training and Validation Accuracy
Accuracy
0.75
0.80
0.85
0.90
0.95
0
2
4
6
8
10
Epochs
Training Accuracy
Validation Accuracy
Training and Validation Loss
Loss
0.0
0.2
0.4
0.6
0.8
1.0
1.2
0
2
4
6
8
10
Epochs
Training Loss
Validation Loss
x=7.04 y=0.756
ENG
IN
02:42
20-02-2024