Model Development Phase Template

Date	19 July 2024
Name	Sourabh Sanjay Dabhade
Project Title	Greenclassify: Deep Learning-Based Approach For Vegetable Image Classification
Maximum Marks	10 Marks

Initial Model Training Code, Model Validation and Evaluation Report

The Initial Model Training Code employs selected algorithms, including CNN, Xception, Inception, ResNet50, and VGG16, on the vegetable image dataset, setting the foundation for effective image classification. The subsequent Model Validation and Evaluation Report rigorously assesses model performance using metrics such as accuracy and precision to ensure reliability and effectiveness in accurately classifying various vegetable types. This comprehensive approach ensures that the models are robust and capable of performing well in real-world scenarios.

Initial Model Training Code (5 marks):

1. CNN (Convolutional Neural Network)

```python
tf.random.set_seed(1234)
model = Sequential()
```

```python
## Add layers to cnn model

# INPUT AND HIDDEN LAYERS

# Convolutional Layer
model.add(Conv2D(filters = 32,
                 kernel_size = 3,
                 padding = "same",
                 activation = "relu",
                 input_shape = [224, 224, 3])
        )

# Pooling Layer
model.add(MaxPooling2D(pool_size = (2,2)))

# Convolutional Layer
model.add(Conv2D(filters = 64,
                 kernel_size = 3,
                 padding = "same",
                 activation = "relu",)
         )
```

```python
# Pooling Layer
model.add(MaxPooling2D())

# CLASSIFICATION

# Flatten Layer
model.add(Flatten())

# Fully Connected Layer
model.add(Dense(128, activation = "relu"))

# Output Layer
model.add(Dense(15, activation = "softmax"))
```

2. VGG16

```python
from tensorflow.keras.applications.vgg16 import VGG16
from tensorflow.keras.layers import Dense,Flatten
from tensorflow.keras.models import Model
tf.random.set_seed(1234)
```

```python
vgg = VGG16(include_top=False,input_shape=(224,224,3))
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/vgg16/vgg16_weights_tf_dim_ordering_tf_kerne
ls_notop.h5
58889256/58889256 [==============================] - 0s 0us/step
```

```
: for layer in vgg.layers:
    layer.trainable=False
```

```
: x = Flatten()(vgg.output)
```

```
: output = Dense(15,activation='softmax')(x)
```

```
: vgg16 = Model(vgg.input,output)
```

3.   ResNet50

```
from tensorflow.keras.applications.resnet50 import ResNet50
tf.random.set_seed(1234)
```

```
resnet50 = ResNet50(include_top=False,input_shape=(224,224,3))
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/resnet/resnet50_weights_tf_dim_ordering_tf_k
ernels_notop.h5
94765736/94765736 [==============================] - 5s 0us/step
```

```
: for layer in resnet50.layers:
    layer.trainable=False
```

```
: x = Flatten()(resnet50.output)
  output = Dense(15,activation='softmax')(x)
  resnet50 = Model(resnet50.input,output)
```

4.   Inception

```
train = train_gen.flow_from_directory(train_path,target_size=(299,299),batch_size=64)
val = val_gen.flow_from_directory(validation_path,target_size=(299,299),batch_size=64)
```

```
Found 15000 images belonging to 15 classes.
Found 3000 images belonging to 15 classes.
```

```
from tensorflow.keras.applications.inception_v3 import InceptionV3
tf.random.set_seed(1234)
```

```
for layer in inceptionV3.layers:
    layer.trainable=False
```

```
x = Flatten()(inceptionV3.output)
```

```
output = Dense(15,activation='softmax')(x)
```

```
inceptionV3 = Model(inceptionV3.input,output)
```

5. Xception

```
train = train_gen.flow_from_directory(train_path,target_size=(299,299),batch_size=64)
val = val_gen.flow_from_directory(validation_path,target_size=(299,299),batch_size=64)

Found 15000 images belonging to 15 classes.
Found 3000 images belonging to 15 classes.
```

```
from tensorflow.keras.applications.xception import Xception
tf.random.set_seed(1234)
```

```
Xception1 = Xception(include_top=False,input_shape=(299,299,3))
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/xception/xception_weights_tf_dim_ordering_tf
_kernels_notop.h5
83683744/83683744 [==============================] - 5s 0us/step
```

```
for layer in Xception1.layers:
    layer.trainable=False
```

```
x = Flatten()(Xception1.output)
```

```
output = Dense(15,activation='softmax')(x)
```

```
Xception1 = Model(Xception1.input,output)
```

Model Validation and Evaluation Report (5 marks):

Model	Summary	Training and Validation Performance Metrics

| CNN (Convolutional Neural Network) | ```
model.summary()
Model: "sequential"

Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 224, 224, 32)      896

max_pooling2d (MaxPooling2   (None, 112, 112, 32)      0
D)

conv2d_1 (Conv2D)            (None, 112, 112, 64)      18496

max_pooling2d_1 (MaxPoolin   (None, 56, 56, 64)        0
g2D)

flatten (Flatten)            (None, 200704)            0

dense (Dense)                (None, 128)               25690240

dense_1 (Dense)              (None, 15)                1935

=================================================================
Total params: 25711567 (98.08 MB)
Trainable params: 25711567 (98.08 MB)
Non-trainable params: 0 (0.00 Byte)
``` | ```
early_stopping = keras.callbacks.EarlyStopping(monitor='val_accuracy', restore_best_weights=True)

model.compile(optimizer="adam",
              loss="categorical_crossentropy",
              metrics=["accuracy"])

hist = model.fit(train_data,
                 epochs=10,
                 verbose=1,
                 validation_data=validation_data,
                 steps_per_epoch=15000//64,
                 validation_steps=3000//64,
                 callbacks=[early_stopping])
```

```
Epoch 1/10
234/234 [==============================] - 190s 806ms/step - loss: 0.2720 - accuracy: 0.9162 - val_loss: 0.2717 - val_accuracy: 0.9215
Epoch 2/10
234/234 [==============================] - 189s 808ms/step - loss: 0.1906 - accuracy: 0.9407 - val_loss: 0.2138 - val_accuracy: 0.9406
Epoch 3/10
234/234 [==============================] - 186s 795ms/step - loss: 0.1525 - accuracy: 0.9531 - val_loss: 0.2309 - val_accuracy: 0.9351
``` |

VGG16

```
Model: "model"

 Layer (type)                  Output Shape              Param #
=================================================================
 input_1 (InputLayer)          [(None, 224, 224, 3)]     0

 block1_conv1 (Conv2D)         (None, 224, 224, 64)      1792

 block1_conv2 (Conv2D)         (None, 224, 224, 64)      36928

 block1_pool (MaxPooling2D)    (None, 112, 112, 64)      0

 block2_conv1 (Conv2D)         (None, 112, 112, 128)     73856

 block2_conv2 (Conv2D)         (None, 112, 112, 128)     147584

 block2_pool (MaxPooling2D)    (None, 56, 56, 128)       0

 block3_conv1 (Conv2D)         (None, 56, 56, 256)       295168

 block3_conv2 (Conv2D)         (None, 56, 56, 256)       590080

 block3_conv3 (Conv2D)         (None, 56, 56, 256)       590080

 block3_pool (MaxPooling2D)    (None, 28, 28, 256)       0

 block4_conv1 (Conv2D)         (None, 28, 28, 512)       1180160

 block4_conv2 (Conv2D)         (None, 28, 28, 512)       2359808

 block4_conv3 (Conv2D)         (None, 28, 28, 512)       2359808

 block4_pool (MaxPooling2D)    (None, 14, 14, 512)       0

 block5_conv1 (Conv2D)         (None, 14, 14, 512)       2359808

 block5_conv2 (Conv2D)         (None, 14, 14, 512)       2359808

 block5_conv3 (Conv2D)         (None, 14, 14, 512)       2359808

 block5_pool (MaxPooling2D)    (None, 7, 7, 512)         0

 flatten_1 (Flatten)          (None, 25088)             0

 dense_2 (Dense)              (None, 15)                376335

=================================================================
Total params: 15091023 (57.57 MB)
Trainable params: 376335 (1.44 MB)
Non-trainable params: 14714688 (56.13 MB)
```

```python
vgg16.compile(loss='categorical_crossentropy',optimizer='adam',metrics=['accuracy'])

early_stopping = keras.callbacks.EarlyStopping(monitor='val_accuracy', restore_best_weights=True)

hist1=vgg16.fit(train_data,validation_data=validation_data,epochs=5,callbacks=[early_stopping])
```

```
Epoch 1/5
235/235 [==============================] - 248s 1s/step - loss: 0.0158 - accuracy: 0.9967 - val_loss: 0.0269 - val_accuracy: 0.9917
Epoch 2/5
235/235 [==============================] - 210s 895ms/step - loss: 0.0122 - accuracy: 0.9969 - val_loss: 0.0697 - val_accuracy: 0.9780
```

ResNet50

```
Model: "model"

Layer (type)                    Output Shape          Param #     Connected to
==================================================================================================
input_1 (InputLayer)            [(None, 224, 224, 3)]  0          []

conv1_pad (ZeroPadding2D)       (None, 230, 230, 3)    0          ['input_1[0][0]']

conv1_conv (Conv2D)             (None, 112, 112, 64)   9472       ['conv1_pad[0][0]']

conv1_bn (BatchNormalizati      (None, 112, 112, 64)   256        ['conv1_conv[0][0]']
on)

conv1_relu (Activation)         (None, 112, 112, 64)   0          ['conv1_bn[0][0]']

pool1_pad (ZeroPadding2D)       (None, 114, 114, 64)   0          ['conv1_relu[0][0]']

pool1_pool (MaxPooling2D)       (None, 56, 56, 64)     0          ['pool1_pad[0][0]']
```

……………..

```
conv5_block3_3_bn (BatchNo      (None, 7, 7, 2048)     8192       ['conv5_block3_3_conv[0][0]']
rmalization)

conv5_block3_add (Add)          (None, 7, 7, 2048)     0          ['conv5_block2_out[0][0]',
                                                                   'conv5_block3_3_bn[0][0]']

conv5_block3_out (Activati      (None, 7, 7, 2048)     0          ['conv5_block3_add[0][0]']
on)

flatten_1 (Flatten)             (None, 100352)         0          ['conv5_block3_out[0][0]']

dense_1 (Dense)                 (None, 15)             1505295    ['flatten_1[0][0]']

==================================================================================================
Total params: 25093007 (95.72 MB)
Trainable params: 1505295 (5.74 MB)
Non-trainable params: 23587712 (89.98 MB)
```

```
resnet50.compile(loss='categorical_crossentropy',optimizer='adam',metrics=['accuracy'])

early_stopping = keras.callbacks.EarlyStopping(monitor='val_accuracy', restore_best_weights=True) #
hist2=resnet50.fit(train_data,validation_data=validation_data,epochs=5,callbacks=[early_stopping])
```

Before hyperparameter tuning:

```
Epoch 1/5
235/235 [==============================] - 219s 891ms/step - loss: 3.4470 - accuracy: 0.2705 - val_loss: 2.2359 - val_accuracy:
0.3287
Epoch 2/5
235/235 [==============================] - 205s 871ms/step - loss: 1.9393 - accuracy: 0.4190 - val_loss: 1.3620 - val_accuracy:
0.5610
Epoch 3/5
235/235 [==============================] - 203s 865ms/step - loss: 1.7601 - accuracy: 0.4799 - val_loss: 1.1985 - val_accuracy:
0.6340
Epoch 4/5
235/235 [==============================] - 213s 905ms/step - loss: 1.5399 - accuracy: 0.5379 - val_loss: 1.3786 - val_accuracy:
0.5927
```

After hyperparameter tuning:

```
Epoch 1/5
235/235 [==============================] - 212s 862ms/step - loss: 0.0144 - accuracy: 0.9963 - val_loss: 0.0144 - val_accuracy:
0.9970
Epoch 2/5
235/235 [==============================] - 190s 809ms/step - loss: 0.0130 - accuracy: 0.9971 - val_loss: 0.0160 - val_accuracy:
0.9960
Epoch 3/5
235/235 [==============================] - 195s 828ms/step - loss: 0.0091 - accuracy: 0.9974 - val_loss: 0.0108 - val_accuracy:
0.9973
Epoch 4/5
235/235 [==============================] - 193s 822ms/step - loss: 0.0100 - accuracy: 0.9974 - val_loss: 0.0229 - val_accuracy:
0.9913
Epoch 5/5
235/235 [==============================] - 195s 828ms/step - loss: 0.0104 - accuracy: 0.9966 - val_loss: 0.0377 - val_accuracy:
0.9863
```

Inception	```
Model: "model_1"
_____
Layer (type)   Output Shape   Param #   Connected to
===
input_3 (InputLayer)   [(None, 299, 299, 3)] 0   []

conv2d_188 (Conv2D)   (None, 149, 149, 32)   864   ['input_3[0][0]']

batch_normalization_188 (B   (None, 149, 149, 32)   96   ['conv2d_188[0][0]']
atchNormalization)

activation_188 (Activation   (None, 149, 149, 32)   0   ['batch_normalization_188[0][0
)   ]']

. . . .

flatten_2 (Flatten)   (None, 131072)   0   ['mixed10[0][0]']

dense_2 (Dense)   (None, 15)   1966095 ['flatten_2[0][0]']

===
Total params: 23768879 (90.67 MB)
Trainable params: 1966095 (7.50 MB)
Non-trainable params: 21802784 (83.17 MB)
_____
```

```python
inceptionV3.compile(loss='categorical_crossentropy',optimizer='adam',metrics=['accuracy'])

early_stopping = keras.callbacks.EarlyStopping(monitor='val_accuracy', restore_best_weights=True)

hist3= inceptionV3.fit(train,validation_data=val,epochs=5,callbacks=[early_stopping])

Epoch 1/5
235/235 [==============================] - 385s 2s/step - loss: 1.2113 - accuracy: 0.9337 - val_loss: 0.0874 - val_accuracy: 0.9920
Epoch 2/5
235/235 [==============================] - 336s 1s/step - loss: 0.2294 - accuracy: 0.9828 - val_loss: 0.2090 - val_accuracy: 0.9837
``` |
| Xception | ```
Model: "model_2"
_____
Layer (type)   Output Shape   Param #   Connected to
==
input_4 (InputLayer)   [(None, 299, 299, 3)] 0   []
block1_conv1 (Conv2D)   (None, 149, 149, 32)   864   ['input_4[0][0]']
block1_conv1_bn (BatchNorm   (None, 149, 149, 32)   128   ['block1_conv1[0][0]']
alization)
block1_conv1_act (Activati   (None, 149, 149, 32)   0   ['block1_conv1_bn[0][0]']
on)
block1_conv2 (Conv2D)   (None, 147, 147, 64)   18432   ['block1_conv1_act[0][0]']

. . . .

flatten_3 (Flatten)   (None, 204800)   0   ['block14_sepconv2_act[0][0]']

dense_3 (Dense)   (None, 15)   3072015 ['flatten_3[0][0]']

==
Total params: 23933495 (91.30 MB)
Trainable params: 3072015 (11.72 MB)
Non-trainable params: 20861480 (79.58 MB)
_____
```

```python
Xception1.compile(loss='categorical_crossentropy',optimizer='adam',metrics=['accuracy'])

early_stopping = keras.callbacks.EarlyStopping(monitor='val_accuracy', restore_best_weights=True)

hist4= Xception1.fit(train,validation_data=val,epochs=5,callbacks=[early_stopping])

Epoch 1/5
235/235 [==============================] - 403s 2s/step - loss: 0.5056 - accuracy: 0.9531 - val_loss: 0.1062 - val_accuracy: 0.9893
Epoch 2/5
235/235 [==============================] - 391s 2s/step - loss: 0.1700 - accuracy: 0.9866 - val_loss: 0.1137 - val_accuracy: 0.9910
Epoch 3/5
235/235 [==============================] - 372s 2s/step - loss: 0.1150 - accuracy: 0.9913 - val_loss: 0.1050 - val_accuracy: 0.9900
``` |