

Model Development Phase Template

Date	12 July 2024
Name	Hrushikesh Patil
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)

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

## 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",)
            )
```

```
# 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

```
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)
```

```
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_kernels_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 (Conv olutio nal Neura l Netw ork)

```
model.summary()

Model: "sequential"
Layer (type) Output Shape Param #
-----
conv2d (Conv2D) (None, 224, 224, 32) 896
max_pooling2d (MaxPooling2D) (None, 112, 112, 32) 0
conv2d_1 (Conv2D) (None, 112, 112, 64) 18496
max_pooling2d_1 (MaxPooling2D) (None, 56, 56, 64) 0
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 800ms/step - loss: 0.2720 - accuracy: 0.9162 - val_loss: 0.2717 - val_accuracy: 0.9215
Epoch 2/10
234/234 [=====] - 189s 800ms/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
```

VGG1 6

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)	500080
block3_conv3 (Conv2D)	(None, 56, 56, 256)	500080
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)
=====

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

```
early_stopping = keras.callbacks.EarlyStopping(monitor='val_accuracy', restore_best_weights=True)  
hist=vgg16.fit(train_data,validation_data=validation_data,epochs=5,callbacks=[early_stopping])
```

```
Epoch 1/5  
215/235 [=====] - 248s 1s/step - loss: 0.8150 - accuracy: 0.9967 - val_loss: 0.8089 - val_accuracy: 0.9907  
Epoch 2/5  
215/235 [=====] - 218s 899ms/step - loss: 0.8122 - accuracy: 0.9969 - val_loss: 0.8087 - val_accuracy: 0.9780
```

ResNe t50

Model: "model"

Layer (Type)	Output Shape	Param #	Connected to
input_1 (InputLayer)	(None, 224, 224, 3)	0	[]
conv1_pad (ZeroPadding2D)	(None, 224, 224, 3)	0	["input_1[0][0]"]
conv1_conv (Conv2D)	(None, 112, 112, 64)	9472	["conv1_pad[0][0]"]
conv1_bn (BatchNormalizat ion)	(None, 112, 112, 64)	256	["conv1_conv[0][0]"]
conv1_relu (Activation)	(None, 112, 112, 64)	0	["conv1_bn[0][0]"]
pool1_pad (ZeroPadding2D)	(None, 112, 112, 64)	0	["conv1_relu[0][0]"]
pool1_pool (MaxPooling2D)	(None, 56, 56, 64)	0	["pool1_pad[0][0]"]

.....

conv1_block1_1bn (BatchNo rmalization)	(None, 7, 7, 2048)	2048	["conv1_block1_1_conv[0][0]"]
conv1_block1_add (Add)	(None, 7, 7, 2048)	0	["conv1_block1_1_bn[0][0]", "conv1_block1_1_conv[0][0]"]
conv1_block1_relu (Activati on)	(None, 7, 7, 2048)	0	["conv1_block1_add[0][0]"]
flatten_1 (Flatten)	(None, 49152)	0	["conv1_block1_relu[0][0]"]
dense_1 (Dense)	(None, 10)	501600	["flatten_1[0][0]"]

Total params: 25811807 (10.72 MB)
 Trainable params: 1385200 (5.32 MB)
 Non-trainable params: 21937712 (80.08 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
215/215 [=====] - 215s 850ms/step - loss: 3.4470 - accuracy: 0.2785 - val_loss: 2.2359 - val_accuracy: 0.3207
Epoch 2/5
215/215 [=====] - 205s 872ms/step - loss: 1.9393 - accuracy: 0.4280 - val_loss: 1.3628 - val_accuracy: 0.5039
Epoch 3/5
215/215 [=====] - 203s 865ms/step - loss: 1.7606 - accuracy: 0.4789 - val_loss: 1.1985 - val_accuracy: 0.6348
Epoch 4/5
215/215 [=====] - 213s 909ms/step - loss: 1.5399 - accuracy: 0.5379 - val_loss: 1.3706 - val_accuracy: 0.5927

```

After hyperparameter tuning:

```

Epoch 1/5
215/215 [=====] - 212s 862ms/step - loss: 0.8544 - accuracy: 0.9983 - val_loss: 0.8544 - val_accuracy: 0.9578
Epoch 2/5
215/215 [=====] - 196s 889ms/step - loss: 0.8210 - accuracy: 0.9971 - val_loss: 0.8160 - val_accuracy: 0.9980
Epoch 3/5
215/215 [=====] - 195s 838ms/step - loss: 0.8891 - accuracy: 0.9974 - val_loss: 0.8168 - val_accuracy: 0.9673
Epoch 4/5
215/215 [=====] - 193s 822ms/step - loss: 0.8688 - accuracy: 0.9974 - val_loss: 0.8219 - val_accuracy: 0.9943
Epoch 5/5
215/215 [=====] - 195s 838ms/step - loss: 0.8688 - accuracy: 0.9986 - val_loss: 0.8357 - val_accuracy: 0.9861

```


Inception

Model: "model_1"

Layer (type)	Output Shape	Param #	Connected to
input_1 (InputLayer)	(None, 299, 299, 3)	0	[]
conv2d_100 (Conv2D)	(None, 149, 149, 32)	864	["input_1[0][0]"]
batch_normalization_100 (Batch Normalization)	(None, 149, 149, 32)	96	["conv2d_100[0][0]"]
activation_100 (Activation)	(None, 149, 149, 32)	0	["batch_normalization_100[0][0]"]
.....			
flatten_2 (Flatten)	(None, 12288)	0	["activation_100[0][0]"]
dense_2 (Dense)	(None, 15)	186685	["flatten_2[0][0]"]

Total params: 2376879 (90.67 MB)
Trainable params: 186685 (7.50 MB)
Non-trainable params: 2180194 (83.17 MB)

Xception

Model: "model_1"

Layer (type)	Output Shape	Param #	Connected to
input_1 (InputLayer)	(None, 299, 299, 3)	0	[]
block1_conv1 (Conv2D)	(None, 147, 147, 32)	864	["input_1[0][0]"]
block1_conv1_bn (Batch Normalization)	(None, 147, 147, 32)	128	["block1_conv1[0][0]"]
block1_conv1_act (Activation)	(None, 147, 147, 32)	0	["block1_conv1_bn[0][0]"]
block1_conv2 (Conv2D)	(None, 147, 147, 64)	18624	["block1_conv1_act[0][0]"]
.....			
flatten_3 (Flatten)	(None, 20480)	0	["block1_conv2_act[0][0]"]
dense_3 (Dense)	(None, 15)	307205	["flatten_3[0][0]"]

Total params: 2393495 (91.30 MB)
Trainable params: 307205 (11.72 MB)
Non-trainable params: 2086290 (79.58 MB)

```
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: 0.9557 - val_loss: 0.8674 - val_accuracy: 0.9528
Epoch 2/5
235/235 [=====] - 136s 1s/step - loss: 0.2294 - accuracy: 0.9828 - val_loss: 0.2889 - val_accuracy: 0.9837
```

```
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 [=====] - 401s 2s/step - loss: 0.5856 - accuracy: 0.9521 - val_loss: 0.3862 - val_accuracy: 0.9803
Epoch 2/5
235/235 [=====] - 391s 2s/step - loss: 0.1288 - accuracy: 0.9888 - val_loss: 0.1117 - val_accuracy: 0.9918
Epoch 3/5
235/235 [=====] - 372s 2s/step - loss: 0.1158 - accuracy: 0.9913 - val_loss: 0.1058 - val_accuracy: 0.9988
```