Practicum AI

This exercise adapted from Baig et al. (2020) *The Deep Learning Workshop* from Packt Publishers.

Activity 3.02 (Teacher) - Page 150

Fruit Classification with Transfer Learning

In [1]:
```
import tensorflow as tf
```

In [2]:
```
file_url = 'https://github.com/PacktWorkshops/The-Deep-Learning-Workshop/raw/master/Chapter03/Datasets/Activity3.02/fruit
```

In [3]:
```
zip_dir = tf.keras.utils.get_file('fruits360.zip', origin = file_url, extract = True)
```

In [4]:
```
import pathlib
```

In [5]:
```
path = pathlib.Path(zip_dir).parent / 'fruits360_filtered'
```

In [6]:
```
print(path)
```

```
/home/danielmaxwell/.keras/datasets/fruits360_filtered
```

In [16]:
```
train_dir = path / 'Training'
validation_dir = path / 'Test'
```

In [22]:
```
total_train = 11398
total_val = 4752
```

In [64]:
```
from tensorflow.keras.preprocessing.image import ImageDataGenerator
```

In [65]:
```
train_image_generator = ImageDataGenerator(rescale = 1./255, rotation_range = 40, width_shift_range = 0.1,
                                           height_shift_range = 0.1, shear_range = 0.2, zoom_range = 0.2,
                                           horizontal_flip = True, fill_mode = 'nearest')
```

In [66]:
```
validation_image_generator = ImageDataGenerator(rescale = 1./255)
```

In [67]:
```
batch_size = 16
img_height = 100
img_width  = 100
channel    = 3
```

In [70]:
```
train_data_gen = train_image_generator.flow_from_directory(batch_size  = batch_size,
                                                           directory   = train_dir,
                                                           target_size = (img_height, img_width))
```

```
Found 11398 images belonging to 120 classes.
```

In [71]:
```
val_data_gen = validation_image_generator.flow_from_directory(batch_size  = batch_size,
                                                              directory   = validation_dir,
                                                              target_size = (img_height, img_width))
```

```
Found 4752 images belonging to 120 classes.
```

In [33]:
```
import numpy as np
import tensorflow as tf
from tensorflow.keras import layers
```

In [31]:
```
np.random.seed(8)
tf.random.set_seed(8)
```

In [35]:
```
from tensorflow.keras.applications import VGG16
```

In [76]:
```
base_model = VGG16(input_shape = (img_height, img_width, channel), weights = 'imagenet', include_top = False)
```

In [77]:
```
base_model.trainable =  False
```

In [78]:
```
base_model.summary()
```

```
Model: "vgg16"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_2 (InputLayer)         [(None, 100, 100, 3)]     0
_________________________________________________________________
block1_conv1 (Conv2D)        (None, 100, 100, 64)      1792
_________________________________________________________________
block1_conv2 (Conv2D)        (None, 100, 100, 64)      36928
_________________________________________________________________
block1_pool (MaxPooling2D)   (None, 50, 50, 64)        0
_________________________________________________________________
block2_conv1 (Conv2D)        (None, 50, 50, 128)       73856
_________________________________________________________________
block2_conv2 (Conv2D)        (None, 50, 50, 128)       147584
_________________________________________________________________
block2_pool (MaxPooling2D)   (None, 25, 25, 128)       0
_________________________________________________________________
block3_conv1 (Conv2D)        (None, 25, 25, 256)       295168
_________________________________________________________________
block3_conv2 (Conv2D)        (None, 25, 25, 256)       590080
_________________________________________________________________
block3_conv3 (Conv2D)        (None, 25, 25, 256)       590080
_________________________________________________________________
block3_pool (MaxPooling2D)   (None, 12, 12, 256)       0
_________________________________________________________________
block4_conv1 (Conv2D)        (None, 12, 12, 512)       1180160
_________________________________________________________________
block4_conv2 (Conv2D)        (None, 12, 12, 512)       2359808
_________________________________________________________________
block4_conv3 (Conv2D)        (None, 12, 12, 512)       2359808
_________________________________________________________________
block4_pool (MaxPooling2D)   (None, 6, 6, 512)         0
_________________________________________________________________
block5_conv1 (Conv2D)        (None, 6, 6, 512)         2359808
_________________________________________________________________
block5_conv2 (Conv2D)        (None, 6, 6, 512)         2359808
_________________________________________________________________
block5_conv3 (Conv2D)        (None, 6, 6, 512)         2359808
_________________________________________________________________
block5_pool (MaxPooling2D)   (None, 3, 3, 512)         0
=================================================================
Total params: 14,714,688
Trainable params: 0
```

```
Non-trainable params: 14,714,688
_________________________________________________________________
```

In [81]:
```
model = tf.keras.Sequential([
    base_model,
    layers.Flatten(),
    layers.Dense(1000, activation = 'relu'),
    layers.Dense(120, activation = 'softmax')
])
```

In [82]:
```
optimizer = tf.keras.optimizers.Adam(0.001)
```

In [83]:
```
model.compile(loss = 'categorical_crossentropy', optimizer = optimizer, metrics = ['accuracy'])
```

In [84]:
```
model.summary()
```

```
Model: "sequential_4"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
vgg16 (Model)                (None, 3, 3, 512)         14714688
_________________________________________________________________
flatten_4 (Flatten)          (None, 4608)              0
_________________________________________________________________
dense_8 (Dense)              (None, 1000)              4609000
_________________________________________________________________
dense_9 (Dense)              (None, 120)               120120
=================================================================
Total params: 19,443,808
Trainable params: 4,729,120
Non-trainable params: 14,714,688
_________________________________________________________________
```

In [85]:
```
model.fit_generator(
    train_data_gen,
    steps_per_epoch  = total_train // batch_size,
    epochs = 5,
    validation_data  = val_data_gen,
    validation_steps = total_val // batch_size
)
```

```
WARNING:tensorflow:sample_weight modes were coerced from
  ...
    to
  ['...']
WARNING:tensorflow:sample_weight modes were coerced from
  ...
    to
  ['...']
Train for 712 steps, validate for 297 steps
Epoch 1/5
712/712 [==============================] - 340s 478ms/step - loss: 2.0376 - accuracy: 0.5071 - val_loss: 1.2164 - val_acc
uracy: 0.6431
Epoch 2/5
712/712 [==============================] - 343s 482ms/step - loss: 0.6580 - accuracy: 0.8043 - val_loss: 0.6205 - val_acc
uracy: 0.8136
Epoch 3/5
712/712 [==============================] - 343s 481ms/step - loss: 0.4190 - accuracy: 0.8742 - val_loss: 0.5638 - val_acc
uracy: 0.8293
Epoch 4/5
712/712 [==============================] - 339s 477ms/step - loss: 0.3450 - accuracy: 0.8902 - val_loss: 0.4557 - val_acc
uracy: 0.8731
Epoch 5/5
712/712 [==============================] - 331s 465ms/step - loss: 0.2854 - accuracy: 0.9106 - val_loss: 0.3776 - val_acc
uracy: 0.8920
```

Out[85]: <tensorflow.python.keras.callbacks.History at 0x15af12a90>