CS156 (Introduction to AI), Spring 2021

Homework 9 submission

Roster Name: Neeval Kumar

Preferred Name (if different): Chosen Name

Student ID: 011877086

Email address: kumar.neeval@gmail.com

Any special notes or anything you would like to communicate to me about this homework submission goes in here.

References and sources

List all your references and sources here. This includes all sites/discussion boards/blogs/posts/etc. where you grabbed some code examples.

I only used tensorflows website for all the libraries.

https://www.tensorflow.org/tutorials

Solution

Load libraries and set random number generator seed

In [1]:
```
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers
import os
import matplotlib.pyplot as plt
from skimage import io
import numpy as np
from sklearn.metrics import plot_confusion_matrix
import seaborn as sns
from sklearn.metrics import accuracy_score
```

In [2]:
```
np.random.seed(42)
```

Load images into keras datatset

In [3]:
```
image_size = (180, 180)
batch_size = 32

print("Preprocessing Training set: ")
train_ds = tf.keras.preprocessing.image_dataset_from_directory(
    "flowers/training",
    labels='inferred',
    label_mode='categorical',
    validation_split=0.2,
    subset="training",
    seed=42,
    image_size=image_size,
    batch_size=batch_size,)

print("\nPreprocessing Validation set: ")
val_ds = tf.keras.preprocessing.image_dataset_from_directory(
    "flowers/training",
    validation_split=0.2,
    labels='inferred',
    label_mode='categorical',
    subset="validation",
    seed=42,
    image_size=image_size,
    batch_size=batch_size,
)

class_names = train_ds.class_names

print("\nClass labels are: \n", class_names)
```

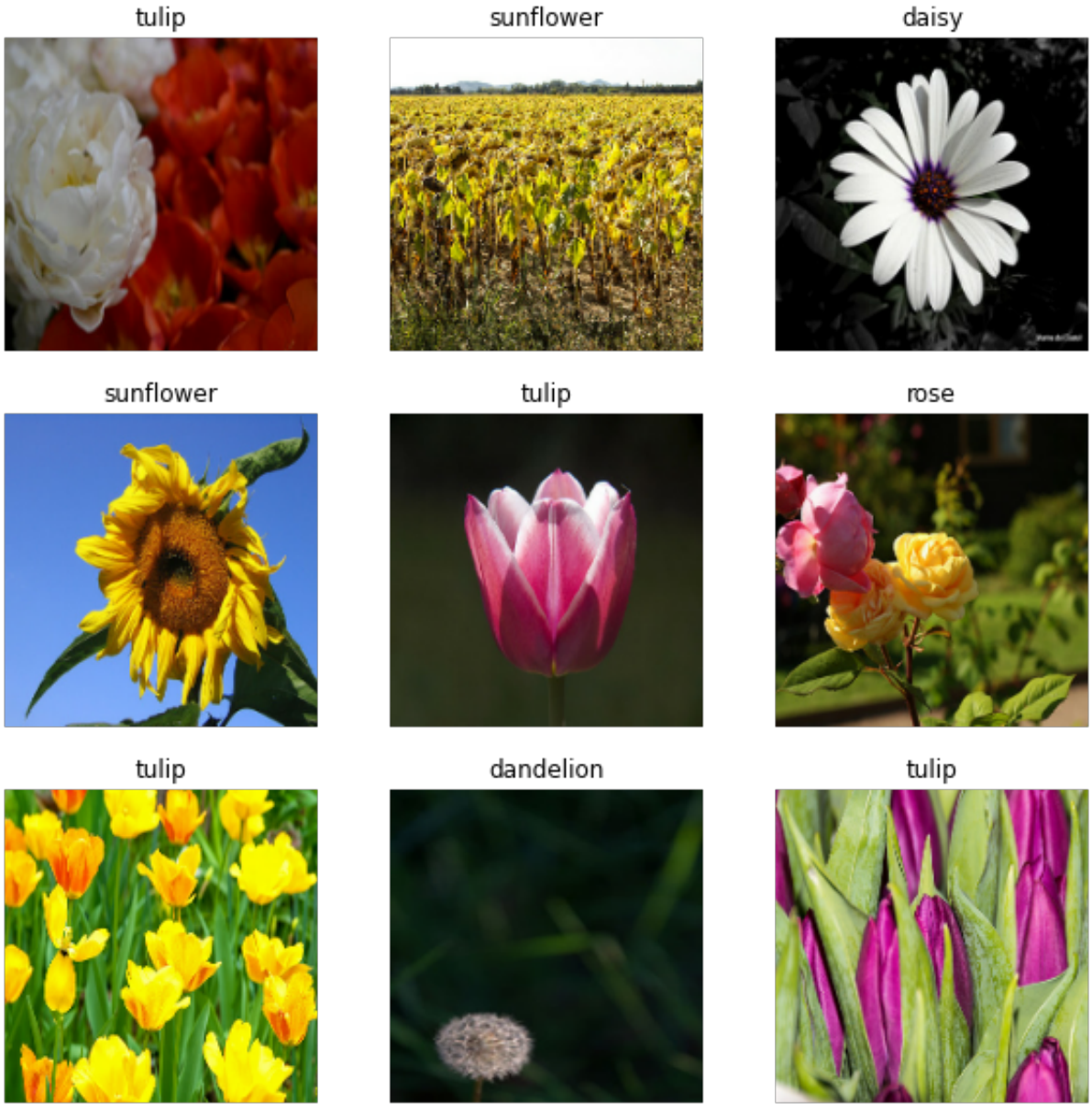
```
Preprocessing Training set:
Found 3456 files belonging to 5 classes.
Using 2765 files for training.

Preprocessing Validation set:
Found 3456 files belonging to 5 classes.
Using 691 files for validation.

Class labels are:
 ['daisy', 'dandelion', 'rose', 'sunflower', 'tulip']
```

Plot some images with their true labels

In [4]:
```
plt.figure(figsize=(10, 10))
for images, labels in train_ds.take(1):
    for i in range(9):
        ax = plt.subplot(3, 3, i + 1)
        plt.imshow(images[i].numpy().astype("uint8"))
        arr = labels[i].numpy()
        index_of = np.where(arr==1.)
        index_of = int(index_of[0])
        plt.title(class_names[index_of])
        plt.axis("off")
```



Augment Images and then plot augments

In [5]:
```
data_augmentation = keras.Sequential(
    [
        layers.experimental.preprocessing.RandomFlip("horizontal"),
        layers.experimental.preprocessing.RandomRotation(0.1),
    ]
)
```

In [6]:
```
plt.figure(figsize=(10, 10))
for images, _ in train_ds.take(1):
    for i in range(9):
        augmented_images = data_augmentation(images)
        ax = plt.subplot(3, 3, i + 1)
        plt.imshow(augmented_images[0].numpy().astype("uint8"))
        plt.axis("off")
```

CNN that makes a model

In [7]:
```
def make_model(input_shape, num_classes):
    inputs = keras.Input(shape=input_shape)
    # Image augmentation block
    x = data_augmentation(inputs)

    # Entry block
    x = layers.experimental.preprocessing.Rescaling(1.0 / 255)(x)
    x = layers.Conv2D(32, 3, strides=2, padding="same")(x)
    x = layers.BatchNormalization()(x)
    x = layers.Activation("relu")(x)

    x = layers.Conv2D(64, 3, padding="same")(x)
    x = layers.BatchNormalization()(x)
    x = layers.Activation("relu")(x)

    previous_block_activation = x  # Set aside residual

    for size in [128, 256, 512, 728]:
        x = layers.Activation("relu")(x)
        x = layers.SeparableConv2D(size, 3, padding="same")(x)
        x = layers.BatchNormalization()(x)

        x = layers.Activation("relu")(x)
        x = layers.SeparableConv2D(size, 3, padding="same")(x)
        x = layers.BatchNormalization()(x)

        x = layers.MaxPooling2D(3, strides=2, padding="same")(x)

        # Project residual
        residual = layers.Conv2D(size, 1, strides=2, padding="same")(
            previous_block_activation
        )
        x = layers.add([x, residual])  # Add back residual
        previous_block_activation = x  # Set aside next residual

    x = layers.SeparableConv2D(1024, 3, padding="same")(x)
    x = layers.BatchNormalization()(x)
    x = layers.Activation("relu")(x)

    x = layers.GlobalAveragePooling2D()(x)
    if num_classes == 2:
        activation = "sigmoid"
        units = 1
    else:
        activation = "softmax"
        units = num_classes

    x = layers.Dropout(0.5)(x)
    outputs = layers.Dense(units, activation=activation)(x)
    return keras.Model(inputs, outputs)
```

In [8]:
```
train_ds = train_ds.prefetch(buffer_size=32)
val_ds = val_ds.prefetch(buffer_size=32)
model = make_model(input_shape=image_size + (3,), num_classes=5)
#keras.utils.plot_model(model, show_shapes=True)
model.summary()
```

```
Model: "model"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
input_1 (InputLayer)         [(None, 180, 180, 3)]     0         
_________________________________________________________________
sequential (Sequential)      (None, 180, 180, 3)       0         
_________________________________________________________________
rescaling (Rescaling)        (None, 180, 180, 3)       0         
_________________________________________________________________
conv2d (Conv2D)              (None, 90, 90, 32)        896       
_________________________________________________________________
batch_normalization (BatchNorma (None, 90, 90, 32)        128       
_________________________________________________________________
activation (Activation)      (None, 90, 90, 32)        0         
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 90, 90, 64)        18496     
_________________________________________________________________
batch_normalization_1 (BatchNor (None, 90, 90, 64)        256       
_________________________________________________________________
activation_1 (Activation)    (None, 90, 90, 64)        0         
_________________________________________________________________
activation_2 (Activation)    (None, 90, 90, 64)        0         
_________________________________________________________________
separable_conv2d (SeparableConv (None, 90, 90, 128)       8896      
_________________________________________________________________
batch_normalization_2 (BatchNor (None, 90, 90, 128)       512       
_________________________________________________________________
```

```
__________________________
activation_3 (Activation)       (None, 90, 90, 128)  0           batch_normali
zation_2[0][0]
__________________________________________________________________________________
__________________________
separable_conv2d_1 (SeparableCo (None, 90, 90, 128)  17664       activation_3[
0][0]
__________________________________________________________________________________
__________________________
batch_normalization_3 (BatchNor (None, 90, 90, 128)  512         separable_con
v2d_1[0][0]
__________________________________________________________________________________
__________________________
max_pooling2d (MaxPooling2D)    (None, 45, 45, 128)  0           batch_normali
zation_3[0][0]
__________________________________________________________________________________
__________________________
conv2d_2 (Conv2D)               (None, 45, 45, 128)  8320        activation_1[
0][0]
__________________________________________________________________________________
__________________________
add (Add)                       (None, 45, 45, 128)  0           max_pooling2d
[0][0]
                                                                 conv2d_2[0][0
]
__________________________________________________________________________________
__________________________
activation_4 (Activation)       (None, 45, 45, 128)  0           add[0][0]
__________________________________________________________________________________
__________________________
separable_conv2d_2 (SeparableCo (None, 45, 45, 256)  34176       activation_4[
0][0]
__________________________________________________________________________________
__________________________
batch_normalization_4 (BatchNor (None, 45, 45, 256)  1024        separable_con
v2d_2[0][0]
__________________________________________________________________________________
__________________________
activation_5 (Activation)       (None, 45, 45, 256)  0           batch_normali
zation_4[0][0]
__________________________________________________________________________________
__________________________
separable_conv2d_3 (SeparableCo (None, 45, 45, 256)  68096       activation_5[
0][0]
__________________________________________________________________________________
__________________________
batch_normalization_5 (BatchNor (None, 45, 45, 256)  1024        separable_con
v2d_3[0][0]
__________________________________________________________________________________
__________________________
max_pooling2d_1 (MaxPooling2D)  (None, 23, 23, 256)  0           batch_normali
zation_5[0][0]
__________________________________________________________________________________
__________________________
conv2d_3 (Conv2D)               (None, 23, 23, 256)  33024       add[0][0]
__________________________________________________________________________________
______________________
```

```
add_1 (Add)                     (None, 23, 23, 256)  0           max_pooling2d
_1[0][0]
                                                                 conv2d_3[0][0
]
__________________________________________________________________________________
_____________________
activation_6 (Activation)       (None, 23, 23, 256)  0           add_1[0][0]
__________________________________________________________________________________
_____________________
separable_conv2d_4 (SeparableCo (None, 23, 23, 512)  133888      activation_6[
0][0]
__________________________________________________________________________________
_____________________
batch_normalization_6 (BatchNor (None, 23, 23, 512)  2048        separable_con
v2d_4[0][0]
__________________________________________________________________________________
_____________________
activation_7 (Activation)       (None, 23, 23, 512)  0           batch_normali
zation_6[0][0]
__________________________________________________________________________________
_____________________
separable_conv2d_5 (SeparableCo (None, 23, 23, 512)  267264      activation_7[
0][0]
__________________________________________________________________________________
_____________________
batch_normalization_7 (BatchNor (None, 23, 23, 512)  2048        separable_con
v2d_5[0][0]
__________________________________________________________________________________
_____________________
max_pooling2d_2 (MaxPooling2D)  (None, 12, 12, 512)  0           batch_normali
zation_7[0][0]
__________________________________________________________________________________
_____________________
conv2d_4 (Conv2D)               (None, 12, 12, 512)  131584      add_1[0][0]
__________________________________________________________________________________
_____________________
add_2 (Add)                     (None, 12, 12, 512)  0           max_pooling2d
_2[0][0]
                                                                 conv2d_4[0][0
]
__________________________________________________________________________________
_____________________
activation_8 (Activation)       (None, 12, 12, 512)  0           add_2[0][0]
__________________________________________________________________________________
_____________________
separable_conv2d_6 (SeparableCo (None, 12, 12, 728)  378072      activation_8[
0][0]
__________________________________________________________________________________
_____________________
batch_normalization_8 (BatchNor (None, 12, 12, 728)  2912        separable_con
v2d_6[0][0]
__________________________________________________________________________________
_____________________
activation_9 (Activation)       (None, 12, 12, 728)  0           batch_normali
zation_8[0][0]
__________________________________________________________________________________
_____________________
```

```
separable_conv2d_7 (SeparableCo (None, 12, 12, 728)  537264      activation_9[
0][0]
__________________________________________________________________________________
_____________________
batch_normalization_9 (BatchNor (None, 12, 12, 728)  2912        separable_con
v2d_7[0][0]
__________________________________________________________________________________
_____________________
max_pooling2d_3 (MaxPooling2D)  (None, 6, 6, 728)    0           batch_normali
zation_9[0][0]
__________________________________________________________________________________
_____________________
conv2d_5 (Conv2D)               (None, 6, 6, 728)    373464      add_2[0][0]
__________________________________________________________________________________
_____________________
add_3 (Add)                     (None, 6, 6, 728)    0           max_pooling2d
_3[0][0]
                                                                 conv2d_5[0][0
]
__________________________________________________________________________________
_____________________
separable_conv2d_8 (SeparableCo (None, 6, 6, 1024)   753048      add_3[0][0]
__________________________________________________________________________________
_____________________
batch_normalization_10 (BatchNo (None, 6, 6, 1024)   4096        separable_con
v2d_8[0][0]
__________________________________________________________________________________
_____________________
activation_10 (Activation)      (None, 6, 6, 1024)   0           batch_normali
zation_10[0][0]
__________________________________________________________________________________
_____________________
global_average_pooling2d (Globa (None, 1024)         0           activation_10
[0][0]
__________________________________________________________________________________
_____________________
dropout (Dropout)               (None, 1024)         0           global_averag
e_pooling2d[0][0]
__________________________________________________________________________________
_____________________
dense (Dense)                   (None, 5)            5125        dropout[0][0]
==================================================================================
====================
Total params: 2,786,749
Trainable params: 2,778,013
Non-trainable params: 8,736
__________________________________________________________________________________
_____________________
```

Train Model

In [9]:
```
epochs = 20

callbacks = [
    keras.callbacks.ModelCheckpoint("save_at_{epoch}.h5"),
]
model.compile(
    optimizer=keras.optimizers.Adam(1e-3),
    loss="categorical_crossentropy",
    metrics=["accuracy"],
)
model.fit(
    train_ds, epochs=epochs, callbacks=callbacks, validation_data=val_ds,
)
```

```
Epoch 1/20
87/87 [==============================] - 356s 4s/step - loss: 1.3189 - accurac
y: 0.4968 - val_loss: 1.7162 - val_accuracy: 0.2590
Epoch 2/20
87/87 [==============================] - 387s 4s/step - loss: 1.0320 - accurac
y: 0.6204 - val_loss: 2.2801 - val_accuracy: 0.2590
Epoch 3/20
87/87 [==============================] - 341s 4s/step - loss: 0.9433 - accurac
y: 0.6565 - val_loss: 3.1217 - val_accuracy: 0.2590
Epoch 4/20
87/87 [==============================] - 339s 4s/step - loss: 0.8775 - accurac
y: 0.6830 - val_loss: 3.2253 - val_accuracy: 0.2590
Epoch 5/20
87/87 [==============================] - 6634s 77s/step - loss: 0.7986 - accur
acy: 0.6966 - val_loss: 3.7676 - val_accuracy: 0.2590
Epoch 6/20
87/87 [==============================] - 4233s 49s/step - loss: 0.7009 - accur
acy: 0.7308 - val_loss: 2.0200 - val_accuracy: 0.3415
Epoch 7/20
87/87 [==============================] - 361s 4s/step - loss: 0.7344 - accurac
y: 0.7372 - val_loss: 1.0680 - val_accuracy: 0.6049
Epoch 8/20
87/87 [==============================] - 410s 5s/step - loss: 0.6332 - accurac
y: 0.7642 - val_loss: 0.9283 - val_accuracy: 0.6787
Epoch 9/20
87/87 [==============================] - 389s 4s/step - loss: 0.6122 - accurac
y: 0.7811 - val_loss: 0.7668 - val_accuracy: 0.7352
Epoch 10/20
87/87 [==============================] - 404s 5s/step - loss: 0.6197 - accurac
y: 0.7649 - val_loss: 1.9760 - val_accuracy: 0.4964
Epoch 11/20
87/87 [==============================] - 407s 5s/step - loss: 0.5665 - accurac
y: 0.7918 - val_loss: 0.7083 - val_accuracy: 0.7410
Epoch 12/20
87/87 [==============================] - 401s 5s/step - loss: 0.5752 - accurac
y: 0.7831 - val_loss: 0.6905 - val_accuracy: 0.7757
Epoch 13/20
87/87 [==============================] - 408s 5s/step - loss: 0.5498 - accurac
y: 0.7880 - val_loss: 0.9262 - val_accuracy: 0.7004
Epoch 14/20
87/87 [==============================] - 416s 5s/step - loss: 0.5377 - accurac
```

```
y: 0.7976 - val_loss: 0.7034 - val_accuracy: 0.7931
Epoch 15/20
87/87 [==============================] - 407s 5s/step - loss: 0.4841 - accurac
y: 0.8209 - val_loss: 0.5964 - val_accuracy: 0.8061
Epoch 16/20
87/87 [==============================] - 712s 8s/step - loss: 0.4397 - accurac
y: 0.8320 - val_loss: 1.1332 - val_accuracy: 0.7091
Epoch 17/20
87/87 [==============================] - 358s 4s/step - loss: 0.4976 - accurac
y: 0.8151 - val_loss: 0.6495 - val_accuracy: 0.7815
Epoch 18/20
87/87 [==============================] - 358s 4s/step - loss: 0.4395 - accurac
y: 0.8324 - val_loss: 0.6946 - val_accuracy: 0.7887
Epoch 19/20
87/87 [==============================] - 383s 4s/step - loss: 0.4417 - accurac
y: 0.8330 - val_loss: 0.5675 - val_accuracy: 0.8148
Epoch 20/20
87/87 [==============================] - 374s 4s/step - loss: 0.4256 - accurac
y: 0.8451 - val_loss: 0.6091 - val_accuracy: 0.8191
```

Out[9]: `<tensorflow.python.keras.callbacks.History at 0x7fa814a18af0>`

Load Test Dataset

In [10]:
```
test_ds = tf.keras.preprocessing.image_dataset_from_directory(
    "flowers/test",
    labels='inferred',
    label_mode='categorical',
    seed=42,
    image_size=image_size,
    batch_size=1,
)
```

```
Found 861 files belonging to 5 classes.
```

Predict images from test set onto model, create lists for true labels and predicted labels. Also store 3 misclassified for visualization

In [11]:
```
true_labels = []
predicted_labels = []
misclassified = []
count = 0
for images, label in test_ds:
    predictions = model.predict(images)
    best_prediction = np.argmax(predictions)
    true_label = np.argmax(label)
    predicted_labels.append(best_prediction)
    true_labels.append(true_label)

    if best_prediction != true_label and count != 3:
        misclass_arr = [label, images, best_prediction, true_label]
        misclassified.append(misclass_arr)
        count += 1

print("Finished test set onto model")
```

```
Finished test set onto model
```

Print Accuracy score of true labels and predicted labels, then plot confusion matrix

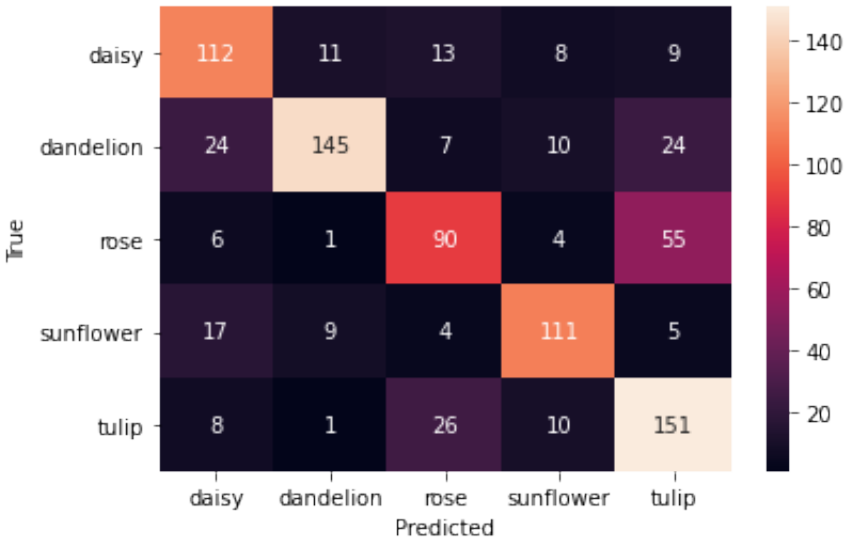
In [12]:
```
print("Accuracy score: ", accuracy_score(true_labels, predicted_labels))

cm = tf.math.confusion_matrix(labels=true_labels, predictions=predicted_label

sns.heatmap(
    cm, annot=True,fmt = 'd',
    xticklabels=class_names,
    yticklabels=class_names)
plt.xlabel("Predicted")
plt.ylabel("True")
```

```
Accuracy score:  0.7073170731707317
```

Out[12]:
```
Text(33.0, 0.5, 'True')
```



Display 3 misclassified Images

In [13]:
```
plt.figure(figsize=(5, 5))
for x in misclassified:
    fig, ax = plt.subplots(nrows=1, ncols=1)
    label = x[0]
    new_image = x[1]
    img_title =  str(class_names[x[3]]) + " predicted as " + str(class_names[
    plt.imshow(new_image[0].numpy().astype("uint8"))
    plt.title(img_title)
    plt.axis("off")
```

```
<Figure size 360x360 with 0 Axes>
```

sunflower predicted as dandelion

sunflower predicted as daisy

dandelion predicted as daisy

In [ ]: