San José State University
Computer Science Department
CS156, Introduction to Artificial Intelligence, Spring 2021

Homework #9

Objective:

This homework's objective is to implement an image classier for flower images.

Details:

For this assignment you will be using flower images for 5 different types of flowers: daisy, dandelion, rose, sunflower, and tulip. This dataset is publicly available from kaggle dataset repository and you can find more information about this dataset here: https://www.kaggle.com/alxmamaev/flowers-recognition

Use input file homework9_input_data.zip for this assignment. This is a compressed file that contains both training and test flower images in subdirectories named by the class names. Decompress this file on your local system where your notebooks can access these image files. Leave the subdirectory structure intact as you will need it for loading images and classes.

For this assignment you can utilize a lot of code snippets from the *CNN.Dog_vs_cat_images.ipynb* Jupyter notebook demonstrated in class.

Use *image_dataset_from_directory()* tensorflow function to load both test and training images, similar to the way shown in *CNN.Dog_vs_cat_images.ipynb* Jupyter notebook. For the test images you do not need to break them into batches, so use *batch_size=1* for the test images. For the training vs. validation images specify the proportions to be 80/20 and load them from the training subdirectory. For all three datasets use *labels='inferred'* and *label_mode='categorical'* parameters. This means that the labels will be inferred from the subdirectory names and will be encoded as a categorical vector, one-hot encoding.

Use data augmentation for your model. Use the same model architecture as the one defined for the dog vs. cat image classification by the *make_model()* function. Make sure to change the number of output classes to the appropriate number for this classification problem. Use *accuracy* as the metric and use *categorical_crossentropy* loss. This is the appropriate loss to use for the type of class encoding that was loaded for the training and validation data.

Train your model on the test/validation data. Keep in mind that the training might take a while and may use most of your computer resources. Allocate enough time to complete this homework. Somewhere around 20 epochs should be enough to reach a sufficiently good predictor. You are welcome to increase the number of epochs if you are not satisfied with the performance after 20 epochs.

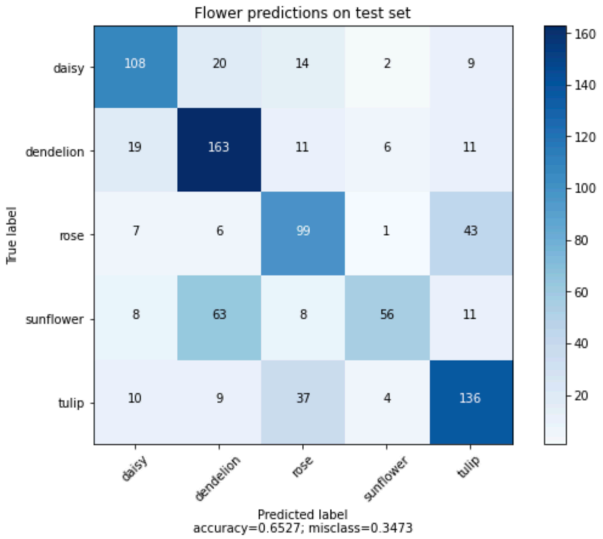
Evaluate your predictor on the set of test images, compose a confusion matrix for your predictions and plot the output. You are welcome to utilize any technique to do this evaluation and plotting. Here are a few tips. Most likely you will have to write your own code that iterates over loaded images and obtains predictions vs. true labels for all the test images. You can iterate over the images as follows (for a test set named *test_ds*):

```
for x, y in test_ds:
    ...
```

I suggest that you save the true labels and predicted labels into a list as you iterate over the test images. To compute the confusion matrix you can use the *confusion_matrix()* function provided in tensorflow:

```
cm = tf.math.confusion_matrix(labels=true_labels, predictions=predicted_labels).numpy()
```

There are many ways to plot a matrix in python and you can utilize any of the available ways to do so. However, you do not need to re-invent the wheel for this assignment and can simply utilize previously developed examples by others available under open source license. For example, here is a nice implementation of plotting for a confusion matrix: https://www.kaggle.com/grfiv4/plot-a-confusion-matrix
Your output should look something similar to this:



Finally, output 3 images that were misclassified along with the true and predicted labels for those images. Your output should look something like this, for 3 different images:

You can utilize any code examples shown in the following Jupyter notebooks:

- CNN.MNIST.ipynb
- CNN.Dog_vs_cat_images.ipynb

Submission:

Email your assignment submission to me at Yulia.Newton@sjsu.edu and the grader (Akshay Kajale) at akshay.kajale@sjsu.edu. Make sure to email this submission by 11:59pm on the due date listed in Canvas. Your sent email is the proof of submission. The subject of the email should say “CS156 Assignment 9”. In the body of the email list your name as it appears on the class roster and your student ID. Attach to this email both the pdf of your Jupyter notebook, which contains the solution for this homework assignment, as well as the notebook itself (the notebook file with .ipynb extension). Make sure to submit both files, otherwise the submission will not be considered complete.

Grading:

I will return the grades as fast as we can grade this homework. Normally it should not take more than a few weeks.

A total of 10 points are possible for this homework assignment.