UDACITY

Logout

‹ Return to "AI Programming with Python Nanodegree" in the classroom

Use a Pre-trained Image Classifier to Identify Dog Breeds

REVIEW | CODE REVIEW | HISTORY

Meets Specifications

Awesome work on this project! You've demonstrated a solid understanding of using ML classifiers within Python. Onward to the next project!

Timing Code

Student calls the time functions before the start of main code and after the main logic has been finished.
Great work making use of the time function here. This is a really useful tool to both manipulate the user experience and to check on the performance of your projects, especially as they scale in size and complexity.

Command Line arguments

adds command line argument for '--dir' uses default ='pet_images/'
Excellent work adding the --dir command line argument! This allows the user to change the working directory as and when required, and doesn't limit them to using just one specified directory.
adds command line argument for '--arch' default='vgg'
Same goes for the --arch CLI. You've demonstrated good knowledge of this!
adds command line argument for '--dogfile' default='dognames.txt'
Nice job with the --dogfile CLI

Pet Image Labels

Makes sure files starting with '.' are ignored. Checks for '.' using a conditional statement.
Great work ignoring certain file types!
Dictionary key and label are in the correct format and retrieves 40 key-value pairs. e.g:- {'Poodle_07956.jpg': ['poodle'], 'fox_squirrel_01.jpg': ['fox squirrel'] ... }
Brilliant job building the dog label dictionary! This was a tricky part of the project, and proves you have the skills to manipulate data (filenames) to produce a given format, no matter how much the filenames themselves differ. Well done!
'in_arg.dir' is passed as an argument inside check_images.py while calling the get_pet_labels function.
Nice - you've passed in the arguments retrieved from the user (via arg parsing, the defaults are used if the user doesn't specify anything) and passed them correctly to the get_pet_labels function.

Classifying Images

Appends images_dir to each value before making the function call. **classifier(images_dir+users_key, model)**
Great work here in passing the image directory (the argument obtained using the arg_parser as specified by the user, of by the default argument), the key (filename) and the model architecture to the classifier function. Just to recap this bit - this function then makes calls to the pre-trained image classifier neural network which has been trained on millions of images to learn how to predict what the images you pass to it are.
Convert the output to lower case and strip any whitespaces
Formatting looks good!
Appends 1 to correct label, and 0 to falsely classified values

Classifying Labels as Dogs

Check the displayed output and see if all matches are appropriately displayed.
Good stuff - all matches between the true labels (i.e. adjusted filenames) and the AI classifier labels are correctly categorised.
Check the displayed output and see if all non matches are appropriately displayed
All the displayed outputs match up and are appropriately displayed, good job.

Results

All three models score as expected.
Brilliant - all model outputs score as expected. Well done.

DOWNLOAD PROJECT

Rate this review

RETURN TO PATH

Rate this review