PROJECT

Object Classification

A part of the Deep Learning Nanodegree Foundation Program

PROJECT REVIEW | CODE REVIEW | NOTES

Requires Changes

2 SPECIFICATIONS REQUIRE CHANGES

SHARE YOUR ACCOMPLISHMENT

Awesome

You did really great! Just small things to fix. For some reason your submission only achieves 38%. It can be an error in the system. Check the files if the result were better and submit it again, please. In the comments below I answered your questions.
About your comment of not being able to recreate the same performance as before. Are you saying it with the same code it is giving a worst performance? If yes, it can be a simple random factor. Your weights are initialized randomly, maybe you started with a great performance and it is not normal, maybe it was not even correct in reality.

Suggestion

Here there is a great blog post about Conv Nets: http://colah.github.io/posts/2014-07-Conv-Nets-Modular/
Great article about CNN: http://www.matthewzeiler.com/pubs/arxive2013/eccv2014.pdf
I recommend checking the Google Python Style Guide, there are great tips about how to improve coding, in general:
https://google.github.io/styleguide/pyguide.html

Required Files and Tests

✓ **The project submission contains the project notebook, called "dlnd_image_classification.ipynb".**

All the unit tests in project have passed.

Awesome

You have passed all the unit tests! Congratulations!

Preprocessing

Rate this review

The `normalize` function normalizes image data in the range of 0 to 1, inclusive.

Awesome

You did great in the preprocessing.

The `one_hot_encode` function encodes labels to one-hot encodings.

Awesome

Great use of the LabelBinarizer.

Neural Network Layers

The neural net inputs functions have all returned the correct TF Placeholder.

The `conv2d_maxpool` function applies convolution and max pooling to a layer.

The convolutional layer should use a nonlinear activation.

This function shouldn't use any of the tensorflow functions in the tf.contrib or tf.layers namespace.

Awesome

You did great here, just for information you could have done things here in two different orders:
1)convolution
2)activation
3)max pooling

or
1)convolution
2)max pooling
3)activation
Even if you are able to do both performing the activation function after max pooling is better: https://www.quora.com/In-Convolutional-Nets-CNN-isnt-the-maxpool-layer-and-ReLU-layer-redundant

✓ The `flatten` function flattens a tensor without affecting the batch size.

Awesome

Nicely done. I like to use the tf.contrib.layers.flatten, it is simpler to code.

✓ The `fully_conn` function creates a fully connected layer with a nonlinear activation.

Awesome

Well done here. Great that you used tf.contrib.layers.fully_connected.
Here there is some articles that could help you choose the activation function:
https://www.quora.com/What-is-special-about-rectifier-neural-units-used-in-NN-learning
https://www.quora.com/What-is-the-vanishing-gradient-problem

✓ The `output` function creates an output layer with a linear activation.

Neural Network Architecture

↻ The `conv_net` function creates a convolutional model and returns the logits. Dropout should be applied to alt least one layer.

Requirement

For some reason your submission only achieves 38%. It can be an error in the system but I will explain to you anyway.
Here it is missing dropout but the importance of dropout you can check in this two articles:
http://blog.kaggle.com/2015/01/02/cifar-10-competition-winners-interviews-with-dr-ben-graham-phil-culliton-zygmunt-zajac/
http://www.cs.toronto.edu/~rsalakhu/papers/srivastava14a.pdf
The main idea here is to show that you understand how to use. Usually to reach the same result with and without dropout you need to increase the number of neurons (with dropout) but in big data sets it is really good.
There are two good articles that you can see their architecture of the CNN to image classification:

http://cs231n.github.io/convolutional-networks/#architectures
https://arxiv.org/ftp/arxiv/papers/1512/1512.00242.pdf

Neural Network Training

✓ The `train_neural_network` function optimizes the neural network.

Awesome

You did great in all the `session.run` parts.

✓ The `print_stats` function prints loss and validation accuracy.

✓ The hyperparameters have been set to reasonable numbers.

Awesome

All the parameters are reasonable but your result is not good yet.
You may need to increase the number of epochs or the neurons on the neural net.

The neural network validation and test accuracy are similar. Their accuracies are greater than 50%.

Requirement

For some reason your submission only achieves 38%. It can be an error in the system. Check the files if the result were better and submit it again, please.

RESUBMIT PROJECT

DOWNLOAD PROJECT

Learn the best practices for revising and resubmitting your project.

RETURN TO PATH

Student FAQ