UDACITY

PROJECT

Object Classification

A part of the Deep Learning Nanodegree Foundation Program

PROJECT REVIEW

CODE REVIEW

NOTES

SHARE YOUR ACCOMPLISHMENT!

Meets Specifications

Hey! Excellent job!
Congratulations on passing the Object Classification project

You got great results with the techniques we learned so far, more will be coming and you will be able to pull more out of your network

SUGGESTION TO IMPROVE RESULTS

For Neural network or any Machine Learning problems, it is always desirable to have as much data as possible. And as your dataset is a bunch of images, we can augment it by slightly modifying the images. Basically, we can apply some rotation, cropping and changing the colors of the input images. You can try it to see if improves the results. I would suggest trying to generate that data using the simple library from here and see what your results will be with the same NN structure.

To don't get credit for explaining the reasons about `stddev` values let me point you to the discussion on Forums: CLICK
Also, fun fact, Andrej Karpathy just tweeted a few days ago about the importance of weight initialization: here. Sometimes even the best guys in the field face similar things, fun field to be in

Great progress so far, keep it up!
Kudos and happy learning

Required Files and Tests

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

iPython Notebook is present.

All the unit tests in project have passed.

Your code passed the unit tests. Great job!

Preprocessing

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

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

Another way you could do one-hot encoding is using **numpy**: `np.eye(10)[x]`

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.

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

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

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

Flawless implementation of all neural network layers!

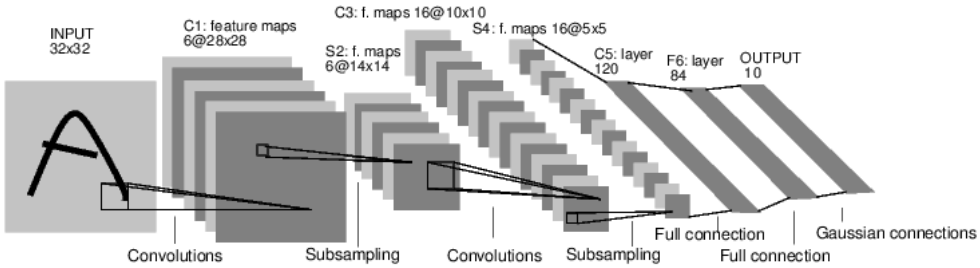
Neural Network Architecture

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

Great

OPTIONAL:

You might consider adding more of convolutional layers and have more of the fully connected ones, it will let your network be more flexible.
There is no rule of thumb in this field, all of the great Networks are products of countless hours of trying different combinations pretty much randomly. If you read papers there are no much explanation why one thing or another one worked, what it actually says is we took this we tried it and it worked, or it didn't work we changed this and it worked, in that manner.
You can try to implement some popular Neural Network architectures. And one of the good ones and simple ones to start with is popular LeNet, developed by Yan LeCun (director of AI Research at Facebook), and see how well it performs if you got more time try to implement AlexNet and so on.
Here is how LeNet looks like:



Do not afraid of taking ready network structures and use them for your problem, it is alright, and will save you a lot of time keeping away from reinventing the wheel.
Also, you can keep modify your own one, increase the number of Convolutional layers and fully connected ones, play with parameters, make the network a little bigger so it can learn more and see how it goes.

Neural Network Training

The `train_neural_network` function optimizes the neural network.

The `print_stats` function prints loss and validation accuracy.

The hyperparameters have been set to reasonable numbers.

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

DOWNLOAD PROJECT

RETURN TO PATH

Rate this review

Student FAQ