PROJECT

Object Classification

A part of the Deep Learning Nanodegree Foundation Program

PROJECT REVIEW | CODE REVIEW | NOTES

Requires Changes

SHARE YOUR ACCOMPLISHMENT

1 SPECIFICATION REQUIRES CHANGES

Great first submission on this project! 👍

You only need to make a small change to complete the project. Check out my comments below for the required change and some tips to further improve your projects and the prediction accuracy.

Required Files and Tests

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

✓ **All the unit tests in project have passed.**

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.**

Good to see you have used sklearn's LabelBinarizer - a pure Numpy solution would be:

```
    one_hot = np.eye(10)[x]
    return one_hot
```

Neural Network Layers

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

Good work! You have created the placeholders with the right shapes and data types and given them proper names.

✓ **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.

Great job! You correctly implemented a convolution, activation and max pooling layer in the right order.

Also well done on the weight initialization, as you might have noticed this can speed up the learning of your network considerably. Check out http://cs231n.github.io/neural-networks-2/#init for more tips on weight initialization.

Note that you can interchange the max pooling and nonlinear layer (as long as the nonlinear activation function is an increasing function - you can try to prove this mathematically). If the max pooling is applied before the nonlinear activation, the number of nonlinear activation function calls reduces, so this should give you a slight speed up!

✓ **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.**

Neural Network Architecture

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

The network meets the specifications, but please see the tips in the final comment of the review.

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%.

The accuracy is not greater than 50% but is stuck at 10%, which means the network is basically random guessing.

The problem of the network is the big number of filters (384) in the convolutional layer, reducing it to for example 32 will get at started learning. I got the following results running your notebook:

```
Epoch  1, CIFAR-10 Batch 1:  2.18672 ,  0.1908
Epoch  1, CIFAR-10 Batch 2:  2.04459 ,  0.1928
Epoch  1, CIFAR-10 Batch 3:  1.6819 ,  0.2678
Epoch  1, CIFAR-10 Batch 4:  1.7281 ,  0.3114
Epoch  1, CIFAR-10 Batch 5:  1.75792 ,  0.3202
Epoch  2, CIFAR-10 Batch 1:  1.85009 ,  0.348
Epoch  2, CIFAR-10 Batch 2:  1.66961 ,  0.3578
Epoch  2, CIFAR-10 Batch 3:  1.40312 ,  0.3606
Epoch  2, CIFAR-10 Batch 4:  1.58037 ,  0.3766
Epoch  2, CIFAR-10 Batch 5:  1.65605 ,  0.3798
```

In addition: convolutional networks with multiple convolutional layers with increasing depth (e.g. 32, 64, 128), a small convolutional filter (3x3) and stride (1x1), small max pooling size (2x2) and stride (2x2) and only a few or no fully connected layers usually work best. Be sure to check out http://cs231n.github.io/convolutional-networks/#architectures for a detailed discussion of and tips for building convolutional network architectures.

RESUBMIT PROJECT

DOWNLOAD PROJECT

Learn the best practices for revising and resubmitting your project.

RETURN TO PATH

Rate this review

Student FAQ