PROJECT

# Object Classification

A part of the Deep Learning Nanodegree Foundation Program

PROJECT REVIEW | CODE REVIEW | NOTES

## Meets Specifications

SHARE YOUR ACCOMPLISHMENT

You have obviously put a lot of time and effort into this project. Your architecture is quite unique, it great to see this level of experiment and creativity.

### 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.

Very creative coding using LabelBinarizer.

### Neural Network Layers

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

Well done. All placeholders are correctly instantiated.

✓ 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.

Well. As you mentioned, the logic is broken into functions but everything appears to be in place.

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

Well done in implementing `flatten(...)` using base tf functionality

**Note:** This functionality is also provided by the `tf.contrib` method `tf.contrib.layers.flatten(x_tensor, num_outputs)`.

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

Well done in implementing `fully_conn(...)` using base tf functionality.

**Note:** This functionality is also provided by the `tf.contrib` method `tf.contrib.layers.fully_connected(x_tensor, num_outputs, activation_fn=tf.nn.relu)`.

✓ 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.

Well done. A novel and innovative architecture. It is obvious that you have experimented with a range of convolutional network paradigms in developing this solution.

### 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%.

Well done, your testing accuracy of 64% was well above the specification minimum testing accuracy of 50%.

My opinion is that your network has a few more percent of improvement, however given the nature of your network it is difficult to identify where adjustment should be made.

DOWNLOAD PROJECT

RETURN TO PATH