UDACITY

PROJECT

Object Classification

A part of the Deep Learning Nanodegree Foundation Program

PROJECT REVIEW

CODE REVIEW

NOTES

SHARE YOUR ACCOMPLISHMENT!

Requires Changes

3 SPECIFICATIONS REQUIRE CHANGES

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.

Nice job! I would just stick with the /255 though instead of the multiply, because it's probably more accurate by a tiny amount.

For more real-life applications, we would typically use sklearn's StandardScaler

George Hinton, a pioneer in neural networks, has a nice Coursera course on neural nets. This lecture explains why normalization helps.

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

Nice use of np.eye! You can also do it like this: `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.

We wanted a nonlinear activation here, so something like ReLU. Could partially explain some of your strange results.

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.

Nice model! On your last conv layer, you have a convkernel size of 1x1. I would use at least 2x2. Doing a 1x1 convolution is kind of useless. I would also increase the conv_num_outputs like 32, 64, 128, and not use dropout after the conv layers. Finally, I would structure the fully_conn sizes like 512, 256, 128. Usually we decrease fully_conn sizes with depth at the end of the network, and increase conv output sizes with depth at the beginning.

Here's more info on the architecture of conv nets. Usually we don't apply dropout to convolutional layers because they already have a lot of regularization built-in.

Every great net out there I see has the convolutional kernels increasing with network depth. For example, look up the VGG net designs. Here's a keras model, and here's a tensorflow model (the tf model code is quite confusing in my opinion). They start with a number (64) for number of kernels, and double it each time they descend a layer. They are using 3x3 kernels (I've always seen it recommended to use 3x3 or 5x5 kernels, although I've seen 2x2 kernels work in this project quite well). They don't use dropout until the dense layers, and then it's at 0.5. Just some food for thought on conv net design.

You can also use transfer learning, i.e. check out the 'extract features' section of this repo. This will pass the image through some pre-trained world-champion-level networks. You can then use the features vector and pass it through some fully-connected layers to get better results. The only issue here is that cifar-10 images are really small, and they would have to be up-scaled to 224x224 for the feature extraction to work.

Neural Network Training

The `train_neural_network` function optimizes the neural network.

The `print_stats` function prints loss and validation accuracy.

Right now you're printing training accuracy, and we want to see validation accuracy: Use the global variables `valid_features` and `valid_labels` to calculate 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%.

Strange that the training accuracy is going down. This isn't really a sign of overfitting, if anything, underfitting.

RESUBMIT

DOWNLOAD PROJECT

Best practices for your project resubmission

Ben shares 5 helpful tips to get you through revising and resubmitting your project.

Watch Video (3:01)

RETURN TO PATH

Rate this review

Student FAQ