Statistical Learning and Neural Networks

Neural networks Lab

Basics - Running your code and using TensorBoard

You will write your code in a .py source file. In order to run it you need to open a command prompt, move to the directory where your code is by using the cd command

```
cd your_directory
```

and type

```
python your_source_file.py
```

TensorBoard has to be run separately. Open a new command prompt, move to the directory where your code is and type

```
tensorboard --logdir=log_dir --host 127.0.0.1
```

Notice that log_dir is the subdirectory specified in your python source file where the data needed by TensorBoard is written. You only need to run this command once and TensorBoard will stay open and update its graphics when it detects a change in the files in the logdir. Notice that sometimes it can be slow to refresh. To see the graphical interface open the Chrome browser (NOTE: Internet Explorer will not work) and go the following address:

```
http://127.0.0.1:6006
```

MNIST dataset

The MNIST dataset is composed of images of handwritten digits. There are 60000 training images and 10000 testing images. The images are grayscale with size 28×28. Labels identifying the true digit are also provided. A function to import the dataset is already provided:

```
mnist = input_data.read_data_sets(FLAGS.data_dir, one_hot=True)
```

The mnist object allows you to request batches from the *train*, *validation* and *test* sets. The following instruction fetches a batch of size BATCH_SIZE from the *train* set. The images are placed in the tensor batch_xs, while the one-hot encoded labels are in the batch_ys tensor. One-hot encoding means that the vector has 10 entries (same as the number of classes=digits) and they are all 0, except one set to 1 in the position corresponding to the correct digit.

```
batch_xs, batch_ys = mnist.train.next_batch(BATCH_SIZE)
```

The sizes of the tensors are:

```
batch_xs: (BATCH_SIZE,784)
batch_ys: (BATCH_SIZE,10)
```

Exercise 1

Use pure Tensorflow (no Keras, no additional frameworks) to design a fully-connected neural network to classify MNIST images. The network must have:

- Three fully connected layers
- ReLU activation units

Therefore, this is the sequence of tensor sizes as they go through the network:

```
Input: (BATCH_SIZE,784)
First layer output: (BATCH_SIZE, NUM_HIDDEN_1)
Second layer output: (BATCH_SIZE, NUM_HIDDEN_2)
Third layer output: (BATCH_SIZE, 10)
```

The network should be trained with stochastic gradient descent and perform multinomial logistic regression using softmax cross-entropy as loss function.

Given an input image of 28×28 pixels, represented as a row vector 1×784 return the predicted 0...9 digit. You will have to:

- plot the loss function against the iteration number in TensorBoard
- measure the accuracy on the training set and plot it against the iteration number using TensorBoard
- measure the accuracy on the test set

Experiment with the parameters: learning rate in gradient descent, number of neurons in the layers, batch size and number of training iterations.

Exercise 2

Use Keras to design a convolutional neural network that classifies MNIST images. The network must have:

- Two 2D convolutional layers using strided convolution with stride equal to 2 in both spatial directions.
- A fully connected output layer
- ReLU activation units

Therefore, this is the sequence of tensor sizes as they go through the network:

```
Input: (BATCH_SIZE,28,28,1)
First layer output: (BATCH_SIZE, 14, 14, NUM_FILTERS_1)
Second layer output: (BATCH_SIZE, 7, 7, NUM_FILTERS_2)
Third layer output: (BATCH_SIZE, 10)
```

The network should be trained with stochastic gradient descent and perform multinomial logistic regression using softmax cross-entropy as loss function. You will have to:

- plot the loss function against the iteration number in TensorBoard
- measure the accuracy on the training set and plot it against the iteration number using TensorBoard
- measure the accuracy on the test set