Lab Course: distributed data analytics
Exercise Sheet 10

Nghia Duong-Trung, Mohsan Jameel
Information Systems and Machine Learning Lab
University of Hildesheim
Submission deadline: **June 28^{th}, Wednesday 23:59PM** (on LearnWeb, course code: 3117)

Instructions

Please following these instructions for solving and submitting the exercise sheet.

1. You should submit a zip or a tar file containing two things a) python scripts and b) a pdf document.
2. In the pdf document you will explain your approach (i.e. how you solved a given problem), and present your results in form of graphs and tables.
3. The submission should be made before the deadline, only through learnweb.
4. If you are M.Sc. Data Analytics summer 2017 intake student, you should submit to "First term students" link on LearnWeb.
5. If you are not M.Sc. Data Analytics student, you can submit to anyone of the links above.

Setting up TensorFlow

All the solution's code in this exercise sheet has been tested in several environment settings:

1. Local system
   (a) Windows 10, Anaconda3, Python 3.5
   (b) Nvidia Geforce 940MX. You can install the CPU-only version of TensorFlow if your system does not have a Nvidia graphics card. The solution's code is also tested on the CPU-only version of TensorFlow.
   (c) Tensorflow 1.1.0
2. ISMLL's Xeon-phi server
   (a) Ubuntu 16.04.2 LTS, Anaconda3, Python 3.5
   (b) Nvidia Tesla K80
   (c) Tensorflow 1.1.0

Remind yourself by TensorFlow tutorials in the exercise sheet 9. The dataset used in this exercise sheet is the Olivetti faces dataset, e.g. see the exercise sheet 9.

Exercise 1: Neural Networks on the Olivetti faces dataset (5 points)

First of all, let's look at some neural networks tutorials to understand how it works:

1. Deep Learning Tutorials `http://deeplearning.net/tutorial/`
2. Introduction to Neural Networks
   `https://pythonprogramming.net/neural-networks-machine-learning-tutorial/`
3. Multi-Layer Neural Network
   `http://ufldl.stanford.edu/tutorial/supervised/MultiLayerNeuralNetworks/`

Along with your solution, you might have to follow the proposed procedure.

1. Load the Olivetti faces dataset, randomly split it into training set 90% and test set 10%.
2. Define a learning model using cross entropy cost function. Explain how you come up with the learning model. In this exercise, your model should contain [Input Layer - Hidden Layer - Output Layer]. You are free to design the number of nodes in the hidden layer.
3. Train the model on the training set and make prediction on the test set.
4. Report and plot accuracy on both training set and test set.
5. Create a Tensorboard that presents basic information such as scalars, graphs, distributions, histograms and Images. You might check your Tensorboard at `localhost:6006`.

Exercise 2: Deep Neural Networks on the Olivetti faces dataset (5 points)

Basically, a deep neural network is a neural network with more than 2 hidden layers. In this exercise, you are going to extend the previous neural network model by adding the 2nd hidden layer.

Along with your solution, you might have to follow the proposed procedure.

1. Load the Olivetti faces dataset, randomly split it into training set 90% and test set 10%.
2. Define a learning model using cross entropy cost function. Explain how you come up with the learning model. In this exercise, your model should contain [Input Layer - Hidden Layer 1 - Hidden Layer 2 - Output Layer]. You are free to design the number of nodes in the two hidden layers.
3. Train the model on the training set and make prediction on the test set.
4. Report and plot accuracy on both training set and test set.
5. Create a Tensorboard that presents basic information such as scalars, graphs, distributions, histograms and Images. You might check your Tensorboard at `localhost:6006`.

Exercise 3: Convolutional Neural Networks on the Olivetti faces dataset (10 points)

A good tutorial on convolutional neural networks can be found in
`http://cs231n.github.io/convolutional-networks/`.

Along with your solution, you might have to follow the proposed procedure.

1. Load the Olivetti faces dataset, randomly split it into training set 90% and test set 10%.
2. Define a learning model using cross entropy cost function. Explain how you come up with the learning model. In this exercise, your model should contain [Input Layer - Convolutional Layer - Max Pooling Layer - Fully Connected Layer - Output Layer]. You are free to design the number of channels in the convolutional layer and the number of nodes in the fully connected layer.
3. Train the model on the training set and make prediction on the test set.
4. Report and plot accuracy on both training set and test set.
5. Create a Tensorboard that presents basic information such as scalars, graphs, distributions, histograms and Images. You might check your Tensorboard at `localhost:6006`.

Leaderboard (5 Bonus points)

Due to the effect of random sampling when splitting dataset into training set and test set. Let's run each model 5 times and take the average accuracy on training set and test set in the end. The average accuracy score on the test set will be the performance of your model.

Now you can compare the logistic regression, neural networks, deep neural networks and convolutional neural networks models on the Olivetti faces dataset. Let's make a small report over all implemented models which contains: the average accuracy on the training set; the average accuracy on the test set; the average training time; how many epochs; and the best learning rate, parameters and hyperparameters that you use. Make sure that your results can be reproduced.

The top 5 students who get the best average accuracy on the test set will get 2 points in this section.