Neuronale Netze – Exercise – Multi-layer Perceptron

1.1 Introduction

These exercises are for you to understand the most important algorithm of the field: **Back Propagation**. It is not just "*doing the forward pass from the input layer to the output layer, calculating the error between those outputs and the labels, then propagating the error derivatives backward to the input layer to find how we update the weight*". It needs you to understand (not just memorize) the details of the algorithm. Otherwise you cannot perform well on the exam and it is also not going to help you in real scenarios. The statistics from the last two years showed that who got good exercise points tend to have the good overall grade. These exercises are actually the same as last year's, so if you want, you can ask last year students for an easy 4 points, but we're warning you, doing this yourself is the good way to cope with the exam.

This year, we require you to do **only one exercise** - *multi-layer perceptron*, and the exercise might give you maximum 4 exercise points which would be added to a maximum 60 points of the written exam. Please note that those exercise points are added to your final points if and only if you pass the written exam. The exercise points that you achieve will be kept over the next exams, so you do not need to do this year's exercise if you are okay with your last year's exercise points.

You can also do other exercises, such as *perceptron* or *logistic regression* (described in the appendix below), for your own good, but **we will not grade them**. Please note that we build a framework to ease your tasks, and the framework is being completed gradually: First you should do the *perceptron*, then the *logistic regression*, then the *multi-layer perceptron*. You can also learn from the sample codes for the first two exercises: *perceptron* and *logistic regression*. See the branches of the github code which corresponds to those sample codes (master - from the scratch, Ex1 - perceptron, Ex2 - logistic regression).

1.2 Goals

- Be familiar with `Numpy` and `Matplotlib`.
- Be familiar with the NN Praktikum framework design.
- Implement a Multi-layer Perceptron (doing classification or regression tasks)
- Use your Multi-layer Perceptron to classify MNIST handwritten digits.

1.3 Repository

- GitHub: `https://github.com/thanhleha-kit/NNPraktikum.git` (branch: Ex3)

1.4 Data

1.4.1 The real MNIST - handwritten digit recognition dataset

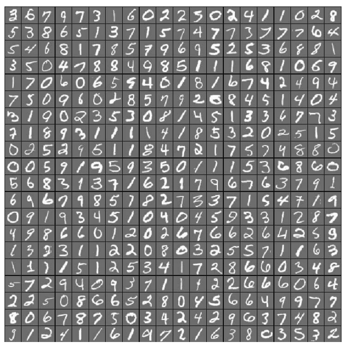

Figure 1: Example digits from MNIST dataset

- Grayscale images of handwritten digits (0–9), 28x28 pixels.
- 50000 images for training, 10000 for validation, 10000 for testing.
- You can use `data/mnist_seven.csv` for testing. The full data will be uploaded soon.

1.5 Your coding tasks

- Complete the code for Logistic Layer if you have not done so (or use the uploaded sample solution).
- Complete the code for a simple Multi-layer Perceptron (MLP) using the Logistic Layer.
- Run your MLP classifier implementation in `Run.py`.
- Vary the learning rate and number of epochs of your MLP classifier and report/draw.
- Bonus:
  - Implement some regularization methods

1.6 Files/Methods/Places should be changed

- `src/model/mlp.py`: Almost all methods, especially the `train()` and `classify()`.
- There should be methods for feed-forwarding and weight updating (or the full back propagation).
- `src/data/mnist_seven.py`: Read the `load()` method to see the changes. This will be used to load the real data for full MNIST task.
- `src/util/activation_function.py`: We need `softmax()` and `softmax_prime()` to be implemented.
- `src/util/loss_functions.py`: The Cross Entropy loss function should be implemented.
- `src/Run.py`: Change/Comment/Uncomment lines of code to run your MLP.

1.7 Notes

- Work in groups
- The framework is freely open. You can:
  - change the name of methods/variables,
  - add more methods/variables,
  - delete the methods that have _ at the beginning in their name.
- Ask questions/open discussions/send suggestions as soon as possible, will not have answers for questions/submissions after deadline.
- (but I will try my best checking your code and answering you all)
- Send me your github.

1.8 Submission

- Submit your code (link to github), the name and matriculation of all members in your group (or any question/suggestion) to thanh-le.ha@kit.edu
- For each group, one email from one person is enough.
- Deadline: `Friday 06.07.2018`.