096260 - Deep Learning
Homework 1

Dmitry Gaidar – 314052903
Meghan Mergui - 342397478
Henrique Lispector – 990512212

Winter 2016
Technion

1. Model architecture description, training procedure.

a. Model Architecture

As model architecture, we chose a fully connected network (network 'n2' in the code).
In this model, we alternate between linear layers and the ReLU function. We then add a linear layer that will set the output size correctly. We are also using the logsoftmax layer at the end of our model, with the purpose of highlighting the largest values of input and suppressing values that are significantly lower than the maximum input.
With this model, we have **59,850 parameters**.

We tried two other networks:

- Another fully connected network, 'n1'.
  We first reshape the image into a vector without copy. Then, we alternate 3 linear layers and the 3 ReLU function. To prevent overfitting, we also used a dropout layer, with a probability of 0.5.
- A spatial convolutional network, 'n3'.

We understand that spatial convolution network is more suitable for this task but we failed to create it without compilation errors.

b. Training Procedure, and attempts.

We tried *data augmentation* (changing random pixels in a batch). However, it didn't improve the result and run time was significantly longer.
So, we decided to train the network without data augmentation but with *more epochs*.
We tried the SGD and Adam *Optimization algorithm*, but Adam Optimization didn't improve our model.
We also tried to add *a dropout layer*, we ran it with more epochs, but it didn't improve the model.

In order to check influence of different parameters, we have added the option to change parameters' values from command-line. Thus, we can easily change the optimization method (sgd, adam), choose the network we want to use (n1 or n2), the number of epochs, of batches, the learning rate and the momentum.

At the end we chose those parameters, and get an **accuracy of 98%.**

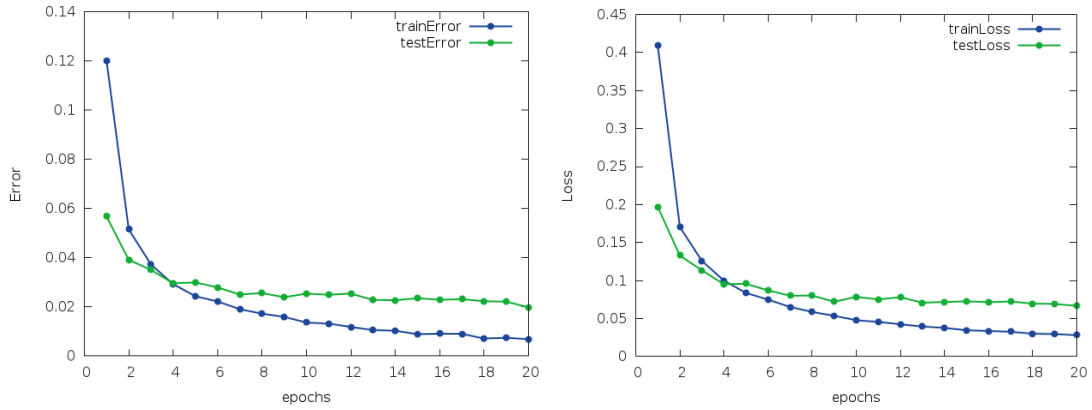
Batch size	Epochs	Momentum	Learning rate	Network	Optimization	**Acc.**
128	20	0.9	0.1	N2	SGD	**98%**

2. Results and graphs.

Batch size	Epochs	Accuracy
128	20	0.980168
128	10	0.975661
256	20	0.97516
256	10	0.969752
64	10	0.975461

In the end, the only parameters that affected significantly the learning accuracy was batch size and number of epochs. We concluded that there is a trade-off between number of batch size and accuracy of the model. If we choose a higher batch size, the compile time will be smaller, but the test error will be higher and vice-versa. Similarly, the higher the number of epochs, the more accurate the model gets, but, at the same time, compile time increases.
When we tried our model with 60 epochs and Dropout we got accuracy of 97.5.
Finally, a batch size of 128, 20 epochs and no Dropout produced the best results for us, and **we get an accuracy of 98.**

See the Error and Loss graphs below for model 'n2':



3. Load the model, test it on the test data.

In our main file, **classificationHW1.lua**, we can change the parameters 'save' and 'load'. The parameter 'save' allows us to save the model we are training when it is equal to 1. It saves it under **model.t7**.
As for the parameter 'load', it allows us to use a saved model to run it on a test set when it is equal to 1.
We also created a specific file, **functionHW1.lua** that does the same (loads the data, loads the model, and returns the test error and test loss).
You can try it on both!

4. Github link to access all the files of this homework.

https://github.com/hlispector/DeepLearning_HW1