Introduction to Optimization
HW 5

Training Neural Networks with SGD and Adagrad Methods

1. Take the neural network (NN) and the data from HW3 and implement SGD and Adagrad methods for this NN.

- SGD: http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture3.pdf, slide 76 + http://cs231n.github.io/optimization-1/ .

- Adagrad: http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture7.pdf, slides 29-31 + http://cs231n.github.io/neural-networks-3/ .

- Use Xavier random weight initialization for training the network: (http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture6.pdf , slide 48 + http://cs231n.github.io/neural-networks-2/#init , "Calibrating the variances with 1/sqrt(n)")

2. Train the network.

- Find the optimal learning rate for the training (http://cs231n.github.io/neural-networks-3/#anneal, use exponential decay), i.e. optimize the following parameters: initial training rate, decay constant. Provide tables with the loss function values at epoch = 1000 for different parameters.

- Find the optimal minibatch size. Plot the loss function value at epoch = 1000 vs. the batch size.

- Study the influence of data reshuffling (after each epoch) on the training process.

- For this exercise, you can optimize the parameters independently, i.e. first find the optimal learning rate (choose a reasonable minibatch size) and then the optimal minibatch size.

- Consider both the training set and the test set (separately) in all your calculations.

3. Compare convergence of these methods with BFGS (plot the cost function value vs. epoch in semiology form).

- Compare both the training set and the test set (separately).

- In your comparison, consider both the loss and the run time.

Remark: The cost function should be divided by the number of samples to keep it independent of the data/batch size.