EECS 504 Foundations of Computer Vision: HW5
Term: Winter 2018
Instructor: Jason J. Corso, EECS, University of Michigan
Due Date: 3/23 23:59 Eastern Time

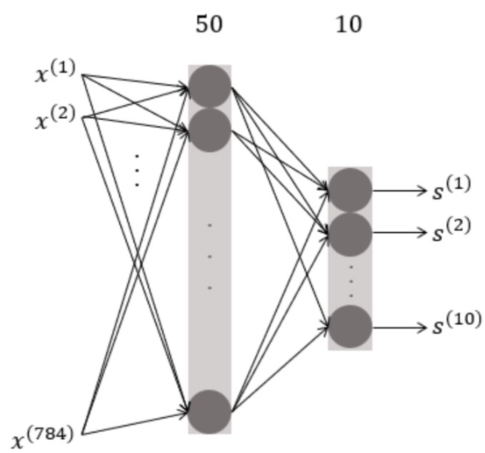
Constraints: This assignment may be discussed with other students in the course but must be written independently. Over-the-shoulder Matlab coding is strictly prohibited. **Web/Google-searching for background material is permitted. However, everything you need to solve these problems is presented in the course notes and background materials, which have been provided already.**

Goals: Deepen the understanding of neural networks.

Data: You need to download `hw5.zip` to complete this assignment. All paths in the assignment assume the data is off the local directory. Besides, you also need to download MNIST test and training set as you did in HW3.

Problem 1 (16): Simple fully-connected NN (2-layer)

We discussed neural networks in the class. In this problem, we will implement a very simple fully-connected neural network. Here is an example of 2-layer neural network.

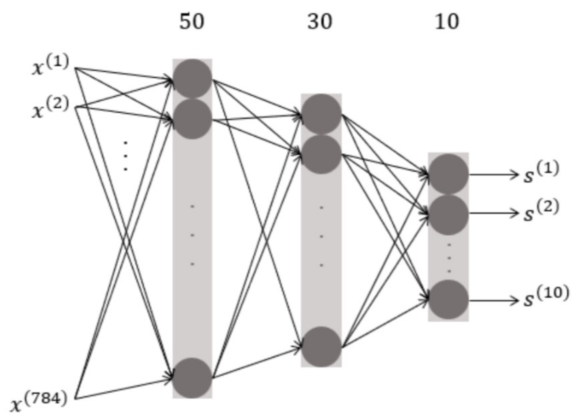


We will apply this model to the MNIST dataset. As you all know, each image of MNIST is of 28 x 28, and it contains a single hand-written digit. If we vectorize it, it will be a 784 vector. We will use the notation x to represent this input vector. The hidden layer consists of 50 neurons, and the output layer has 10 neurons. The final output is a score vector, of which the elements represent the probability of each digit. That is, $s^{(1)}$ represents the probability of x being digit 0, $s^{(2)}$ is for 1, ... and $s^{(10)}$ is for 9. If we feed the model with training samples and corresponding labels, the model can be used to predict which digit the given image contains from 0 to 9. Now, go to `p1.m`.

1. (6) First of all, implement the sigmoid function in `sigmoid.m`. Then, fill in the blanks in `p1.m` following the comments. Submit your `p1.m` to the Canvas.
2. (7) Run `p1.m` and it will display the loss and accuracy. Attach your plots and report the test accuracy in your writeup.
3. (3) How many "learnable" parameters (all the weights and bias) are there in this network? Write down the answer and derivation in your writeup.

Problem 2 (16): 3-layer NN

Let's add one more hidden layer to the network in problem 1. Our network will look like this:

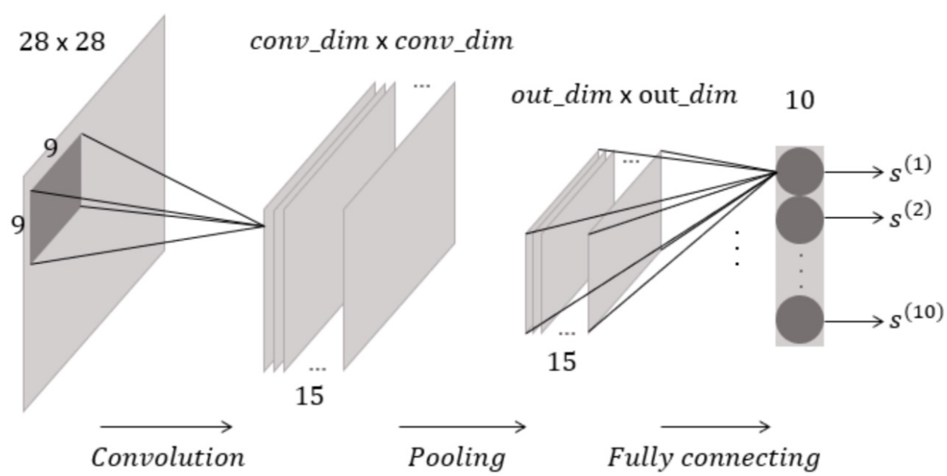


Now, let's repeat the tasks.

1. (6) Fill in the blanks in `p2.m` following the comments. Note that we use Xaxier's scaling factor. Submit your `p2.m` to the Canvas.
2. (7) Run the code. It will also give you two plots showing the loss and the accuracy. Attach your plots and report the test accuracy in your writeup.
3. (3) How many "learnable" parameters (all the weights and bias) are there in this network? Write down the answer and derivation in your writeup.

Problem 3 (20): Simple CNN

In this problem, we will implement CNN with only a single convolution layer and a fully connected layer. The architecture is show below.



The detailed information of this network is in `p3.m`.

1. (6) Fill in the blanks in `p3.m` following the comments. Submit your `p3.m` to the Canvas.
2. (7) Run the code and you may find it very slow. It is recommended to debug your code with less epochs (or less training data) first. Attach the output plots and report the test accuracy in your writeup (with given number of epochs and training data).
3. (3) How many "learnable" parameters (all the weights and bias) are there in this network? Write down the answer and derivation in your writeup.
4. (4) It is very likely that the performance you get in this problem is worse than the result of last two problems. But in many applications, we prefer to use CNN. Describe two advantages of CNN over fully-connected NN.

Submission Process: Submit a single pdf with your answers to these problems, including all plots and discussion. Submit the pdf to Gradescope. The entry code of this course is **9GEG8X**.

For coding assignments, include your code verbatim in your writeup. Put original program files in separate folders corresponding to problems and name your folders as `hw5p1`, `hw5p2`, etc. Pack all the folders into one zip file named as `hw5_<unique_name>.zip` and upload it to Canvas. The problem description will clarify whether you need to attach your code verbatim and/or turn in the original files for that problem, or both. **Code should be well commented for grading**.

Grading and Evaluation: The credit for each problem in this set is given in parentheses at the stated question (sub-question fraction of points is also given at the sub-questions). Partial credit will be given for both paper and Matlab questions. For Matlab questions, if the code does not run, then limited or no credit will be given.