

HOMework #2

ECBM E6040, Professor Aurel A. Lazar

Deadline: 11:59PM, Mar 7th, 2016

INSTRUCTIONS: This homework contains two parts - theoretical and programming. Submission for this homework will be via bitbucket repositories created for each student and should contain the following

1. A file called `hw2.writeup.pdf` that contains solutions to the theoretical questions
2. Put all figures and discussions, and document all parameters you used in the programming question in the IPython notebook file, `hw2b.ipynb`, which is already included in the homework 2 repository. All the discussions should also be included in the notebook file.

Please be advised that the programming part of this homework may take some time to finish, so *start early!*

Theoretical

You will need the definition of the PDF of a matrix normal distribution to complete this part. The following provides a PDF of a simplified special case that will be used in this assignment.

Matrix Normal Distribution: A $n \times n$ square matrix valued random variable \mathbf{A} is said to follow a matrix normal distribution (\mathcal{MN}) with parameters $(\mathbf{M}, \lambda^{-\frac{1}{2}}\mathbf{I}, \lambda^{-\frac{1}{2}}\mathbf{I})$ if it has the following probability density function

$$\mathbb{P}\left(\mathbf{A}; (\mathbf{M}, \lambda^{-\frac{1}{2}}\mathbf{I}, \lambda^{-\frac{1}{2}}\mathbf{I})\right) = \frac{1}{(2\pi)^{\frac{n^2}{2}}} \exp\left(-\frac{1}{2} \text{Tr}\left[\lambda(\mathbf{A} - \mathbf{M})^T(\mathbf{A} - \mathbf{M})\right]\right)$$

PROBLEM a

(40 points)

Given the observed data $(\mathbf{x}^1, \mathbf{y}^1), \dots, (\mathbf{x}^m, \mathbf{y}^m)$, where $\mathbf{x}^i \in \mathbb{R}^n$ and $\mathbf{y}^i \in \mathbb{R}^n$, $\forall i \in [1, m]$, we are interested in finding an $\mathbf{A} \in \mathbb{R}^{n \times n}$ such that *in some sense* (to be defined below) $\mathbf{y}^i \approx \mathbf{A}\mathbf{x}^i$.

For simplicity, we use the following notation

$$\mathbf{Y} = [\mathbf{y}^1 \quad \mathbf{y}^2 \quad \cdots \quad \mathbf{y}^m] \quad \mathbf{X} = [\mathbf{x}^1 \quad \mathbf{x}^2 \quad \cdots \quad \mathbf{x}^m]$$

and assume that $m > n$, and that $\mathbf{X}\mathbf{X}^T$ is invertible.

- (i) For the least squares loss function

$$\mathcal{L}_{ls} = \sum_{i=1}^m \left(\mathbf{y}^i - \mathbf{A}\mathbf{x}^i \right)^T \left(\mathbf{y}^i - \mathbf{A}\mathbf{x}^i \right)$$

find $\mathbf{A}_{ls} = \underset{\mathbf{A}}{\operatorname{argmin}} \mathcal{L}_{ls}$.

- (ii) For the least squares loss function with Frobenius norm regularization term

$$\mathcal{L}_r = \lambda \|\mathbf{A}\|_F^2 + \sum_{i=1}^m \left(\mathbf{y}^i - \mathbf{A}\mathbf{x}^i \right)^T \left(\mathbf{y}^i - \mathbf{A}\mathbf{x}^i \right)$$

find $\mathbf{A}_r = \underset{\mathbf{A}}{\operatorname{argmin}} \mathcal{L}_r$.

Note: $\|\mathbf{A}\|_F^2 = \operatorname{Tr}(\mathbf{A}^T \mathbf{A})$.

- (iii) Assume that the errors $\epsilon_i = \mathbf{y}^i - \mathbf{A}\mathbf{x}^i$ are normally distributed with mean $\mathbf{0}$ under the ideal \mathbf{A} , i.e., $\epsilon_i \sim \mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{I})$.
Find \mathbf{A}_{ML} , the maximum likelihood estimate of \mathbf{A} .
- (iv) Under the same assumption of normal error distribution, consider a prior on \mathbf{A} of the form $\mathbf{A} \sim \mathcal{MN}(\mathbf{M}, \lambda^{-\frac{1}{2}}, \mathbf{I}, \lambda^{-\frac{1}{2}} \mathbf{I})$.
Find \mathbf{A}_{MAP} , the maximum a posteriori estimate of \mathbf{A} . What will be \mathbf{A}_{MAP} if we assume \mathbf{M} to be the zero matrix.
- (v) Comment on the relation between the expressions derived in (i) and (iii), and of those derived in (ii) and (iv).

Programming

For this part, you will experiment with different Multilayer Perceptron configurations, and empirically study various relationships among number of layers and number of parameters. You should start by going through the [Deep Learning Tutorials Project](#). In particular, the source code provided in the Homework 2 repository is excerpted from `logistic_sgd.py` and `mlp.py`.

You are asked to partially reproduce the phenomena shown in Figure 6.9 and Figure 6.10 of the textbook. The original work for these two figures implemented an advanced framework of deep network [2], which is beyond the material covered in the course till now. Instead of reimplementing the original work, you should simply use the multilayer perceptron described in the [tutorial](#).

You will be using the street view house numbers (SVHN) dataset [1]. The dataset is similar in flavor to MNIST, but contains substantially more labeled data, and comes from a significantly harder, real world problem (recognizing digits and numbers in natural scene images). You will use the **Format 2** of the SVHN dataset. Each sample in this format is a MNIST-like 32-by-32 RGB image centered around a single character. Many of the images do contain some distractors on the sides, which of course makes the problem interesting.

The task is to implement an MLP to classify the images of the SVHN dataset. The input to the MLP is a color image, and the output is a digit between 0 and 9.

A python routine called `load_data` is provided to you for downloading and preprocessing the dataset. You should use it, unless you have absolute reason not to. The first time you call `load_data`, it will take some time to download the dataset (about 180 MB). Please be careful NOT TO commit the dataset files into the repository.

Note that all the results, figures, and parameters should be placed inside the IPython notebook file `hw2b.ipynb`.

PROBLEM b (60 points)

1. First enable the construction of MLPs with multiple hidden layers. Implement your MLP in the skeleton `myMLP` class in `hw2b.py`.
2. Implement an MLP with 2 hidden layers. Compare the effect of two activation functions, *tanh* and *softmax*, on neurons in hidden layers with other parameters fixed. Note that the output neuron always uses *softmax*. Document your choice of parameters explicitly, and discuss your test accuracy results in both cases.
3. Experiment with the number of hidden layers. In particular, generate a plot similar to the one in Figure.1. Note that Figure.1 is similar in spirit to Figure 6.9 of the textbook. Each hidden layer should contain the same amount of

neurons. You might want to start with a network with 3 hidden layers, and experiment with parameters (e.g., activation function, learning rate, number of hidden neurons, *etc.*). After finding a set of parameters, run your MLP 8 times with the number of hidden layers varying from 1 to 8 (the total number of layers thus ranges from 3 to 10). Document your choice of parameters explicitly, and discuss your test accuracy results.

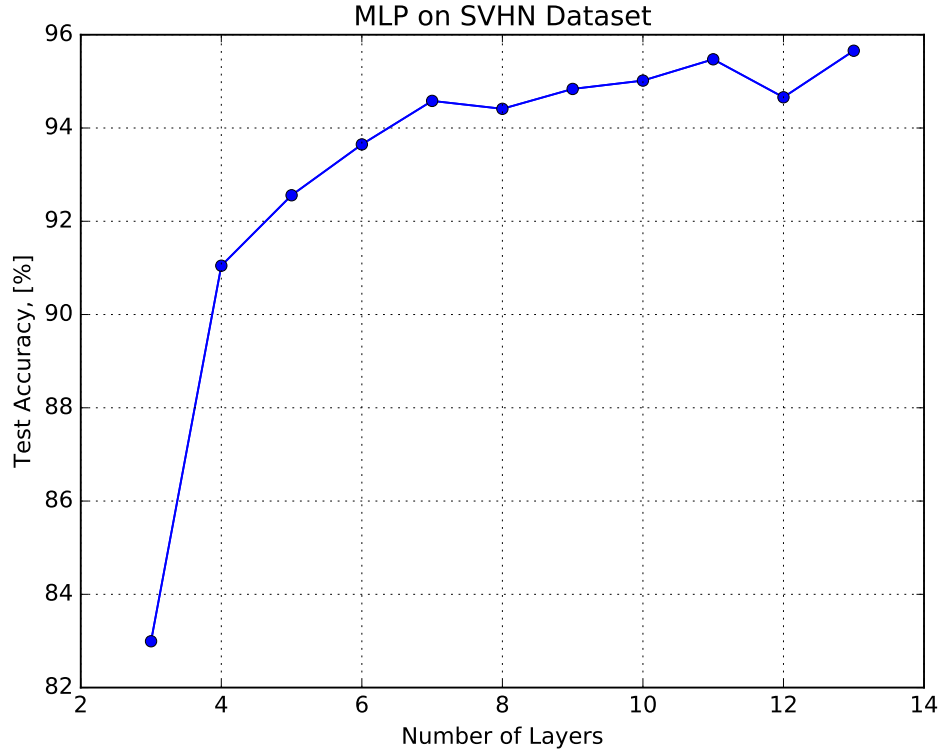


Figure 1:

- Experiment with the number of hidden layers, but fix the total number of neurons in all hidden layers. In particular, generate a plot similar to Figure.2. In Figure.2, the total amount of hidden neurons is fixed at 2.4K. You may chose another number. Each hidden layer should contain the same amount of neurons (that is, $\lfloor \frac{\text{total number}}{\text{number of layers}} \rfloor$ neurons in each layer). Run your MLP 8 times with number of hidden layers varying from 1 to 8. Document your choice of parameters and their number explicitly, and discuss your test accuracy results.

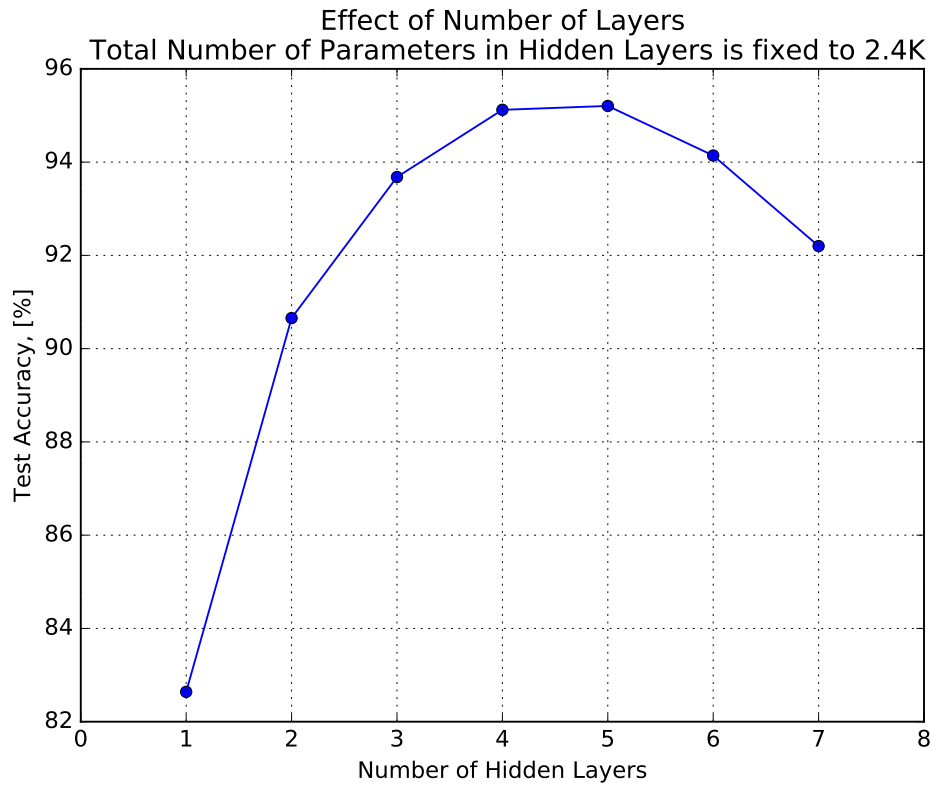


Figure 2:

- For a fixed number of hidden layers experiment with the number of neurons in hidden layers. In particular, generate a plot similar to the one in Figure.3. Note that Figure.3 is similar in spirit to Figure 6.10 in the textbook. Run your MLP with 1 hidden layer 5 times with 5 different numbers of hidden neurons. Repeat the above experiment with 2 hidden layers. Document your choice of parameters and their number explicitly, and discuss your test accuracy results.

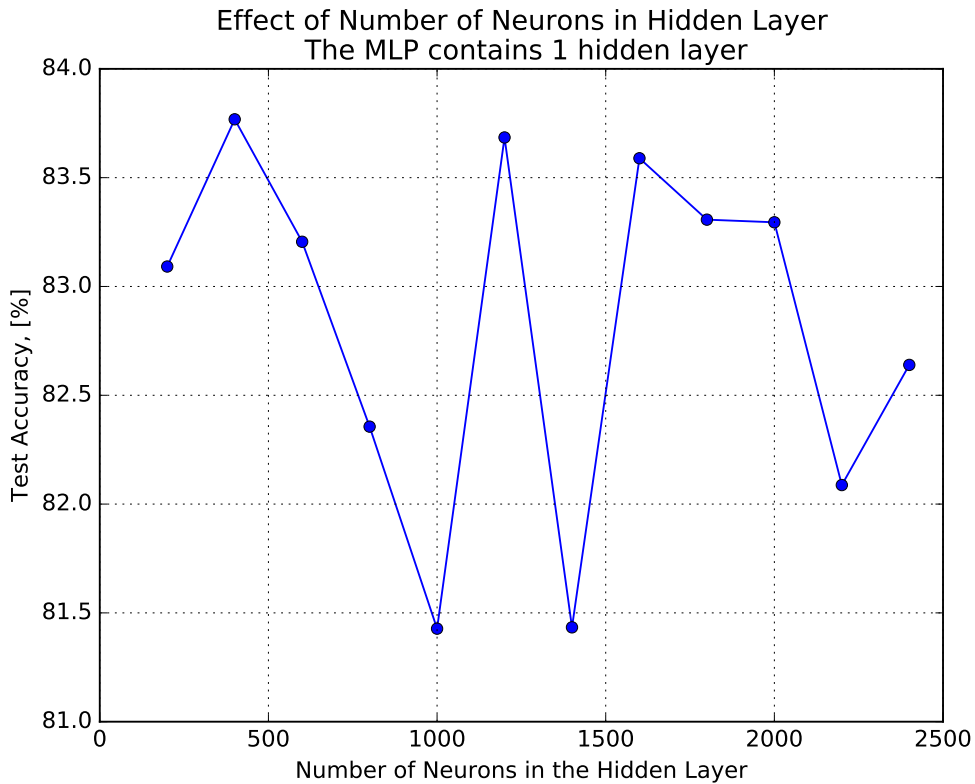


Figure 3:

NEED HELP:

If you have any questions you are advised to use Piazza forum which is accessible through courseworks.

GOOD LUCK!

References

- [1] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, Andrew Y. Ng, “Reading Digits in Natural Images with Unsupervised Feature Learning,” *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*.
- [2] Ian Goodfellow, Yaroslav Bulatov, Julian Ibarz, Sacha Arnoud, Vinay Shet, “Multi-digit Number Recognition from Street View Imagery using Deep Convolutional Neural Networks,” *ICLR 2014*.