Due Date: 10/08/2019 24/05/2019

CENG463 Homework III

In this homework, you are asked to implement a Fully-Connected Neural Network to solve MNIST Hand-written digit recognition problem. A sample code and data is provided you. The homework has 2 parts, one is the implementation, the other one is a report about your implementation.

The implementation should keep *main*, *train* and *test* function as they are (you can only modify *[hidden_layer]* part according to your own design). The whole design, number and size of hidden layers, activation functions, loss function.. etc., is up to you. As long as, the model learns (minimum of %20 accuracy is expected at least) you may change the design. However train, test and main functions and the skeleton that is provided you should stay still. Using any explicit library to shorten the operations on network is prohibited (you can only use numpy, mnist, matplotlib and other default python libraries). To download mnist library you can type ‘pip install python-mnist’. Also we want you to implement the code on Python3 and not use any absolute path in your code, you can use a relative path like “./mnist”.

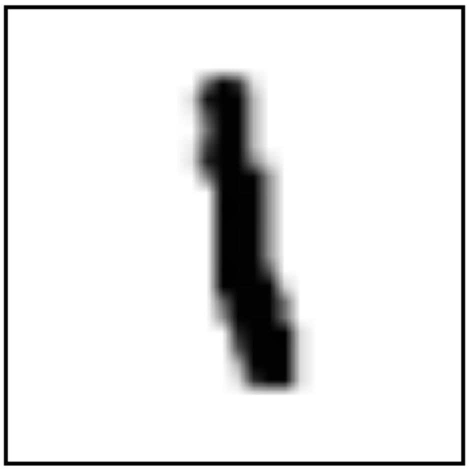
After the implementation we want you to carefully write a report about “What was the reason that your network learned?”, “How would you improve the accuracy more?”, “What was the trade-offs you face with?”, “And the reasoning of why did you design your network in that way” (Reasoning about things like: activation function, loss function, number and size of hidden layers, learning rate). Also the report should have a plot of loss and accuracy change through time and we want you to comment on the plots. (A detailed reasoning about the result and the shape of the graph). Please remember that report has a grading factor as much as implementation.

P.S. : Since you are almost an engineer, you are expected to write efficient , clean, well-commented codes.
P.P.S. : Please **DO NOT** submit whole python virtual environment with your implementation.

Guidelines

Preprocessing

An image and its pixel representation is shown below. All images are 28*28 in size and grey-scaled. Since giving input at 28*28 is not possible on Fully-connected networks, the data are flattened to 784 individual pixel value.



$$\simeq \begin{bmatrix}
0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&.6&.8&0&0&0&0&0&0\\
0&0&0&0&0&0&.7&1&0&0&0&0&0&0\\
0&0&0&0&0&0&.7&1&0&0&0&0&0&0\\
0&0&0&0&0&0&.5&1&.4&0&0&0&0&0\\
0&0&0&0&0&0&0&1&.4&0&0&0&0&0\\
0&0&0&0&0&0&0&1&.4&0&0&0&0&0\\
0&0&0&0&0&0&0&1&.7&0&0&0&0&0\\
0&0&0&0&0&0&0&1&1&0&0&0&0&0\\
0&0&0&0&0&0&0&.9&1&.1&0&0&0&0\\
0&0&0&0&0&0&0&.3&1&.1&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0
\end{bmatrix}$$

Also we use one-hot vectors in classification tasks. One -hot vectors look like this [0, 0, 0, 1, 0, 0] Which represents that the class 4 is predicted. Also for ground-truth labels, one-hot notation is used to determine which class was the true class for that data.

Additionally, generally training data is partitioned into training and validation splits to tune hyperparameters (learning rate, size and number of hidden layers, regularization parameters etc.). You tune hyperparameters with respect to the accuracy on validation data which is an unseen data for the network. Also you should not backpropagate through validation data, it is not here to learn but to test network's generalization.

Postprocessing

The predictions are in form on probability distribution, so to determine which class is chosen by the network, we simply select the index of the highest probability.
So [0.1, 0.04, 0.86] will be [0, 0, 1].

Softmax

Softmax is a function that normalizes a vector into a probability distribution ([11, 13, -1] → [0.1190, 0.8807, 7.32x10^-7]). It is widely used in classification tasks since normalizes into a probability distribution and higher probability of an element with respect to other elements in the list. The formula of softmax is:

Softmax function $\sigma(\mathbf{z})_j$

$$\sigma(\mathbf{z})_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}} \quad \text{for } j = 1, \cdots, K.$$