Project-1 of "Neural Network and Deep Learning"

Yanwei Fu

March 27, 2025

Abstract

(1) This is the first project of our course. The deadline is 8:00 AM, May 1, 2025. Please submit your report on eLearning. Contact: deeplearning.fudan@yandex.com for any assistant.

(2) The goal of your write-up is to document the experiments you've done and your main findings. So be sure to explain the results. Hand in a single PDF file of your report. Enclose a Github link to your codes in your submitted file. You should also provide a link to your dataset and your trained model weights in your report. You may upload the dataset and model into Google Drive or other Netdisk service platform. Also put the name and Student ID in your paper. Lack of code link or model weights link will lead to a penalization of scores.

(3) About the deadline and penalty. In general, you should submit the paper according to the deadline of each mini-project. The late submission is also acceptable; however, you will be penalized 10% of scores for each week's delay.

(4) We provide a Python codes using NumPy package for implementation as a general reference. You need to implement some functions and method **on your own.** See more information in the README of the codes provided. But you can also use any other program languages if you want, e.g., C/C++, R. You may use Mindspore to do some visualization or do some experiments if your implemented version runs too slow. **But you must implement your own version first.**

(5) Note that the goal of this project is to let the students do the practice, and write some basic components of a neural network or CNN. So do not invoke the deep learning functions/modules that can directly give the results of the following questions(e,g, the PyTorch package). The project is very easy, and you do not really need GPUs; just running it on CPU is good enough.

1 Neural Network

In this problem we will investigate handwritten digit classification. MNIST (Modified National Institute of Standards and Technology database) is a large database of handwritten digits commonly used for training various image processing systems. The database is also widely used for training and testing in the field of machine learning. It was created by "re-mixing" the samples from NIST's original datasets. The dataset contains 60,000 training images and 10,000 testing images. Each image is a 28x28 pixel grayscale image and is labeled with the correct digit(0-9) it represents. You need to implement one or more neural network to recognize the handwritten digit, and conduct experiment to test your model and conclude the ability of your model. After that, you may implement several modifications to your model and test whether the model acts better.

Do not worry about whether your mode's performance is better than others', the effort you've made to improve your model matters. Please refer to previous instruction of writing the report.

1.1 Hint

Below are additional features you could try to incorporate into your neural network to improve performance (the options are approximately in order of increasing difficulty). The modifications you make in trying to improve performance are up to you and you can even try things that are not on the question list in Sec. 1.2. But, let's stick with neural networks models and only use one neural network (no ensembles). Remember to write in your reports about what modifications you've made and the effectiveness of the modification.

1.2 Questions (The more you try, the more score you will get.)

1. Change the network structure: the vector *nHidden* specifies the number of hidden units in each layer.
2. Change the training procedure by modifying the sequence of step-sizes or using different step-sizes for different variables. That momentum uses the update

$$w^{t+1} = w^t - \alpha_t \nabla f\left(w^t\right) + \beta_t \left(w^t - w^{t-1}\right)$$

   where α_t is the learning rate (step size) and β_t is the momentum strength. A common value of β_t is a constant 0.9.
3. Try some regularization methods. (Say, l_2 regularization, dropout, early stopping or other methods). Does they improve the model performance on validation set or test set?
4. Implement the cross entropy loss. And incorporate a softmax (multinomial logistic) layer at the end of the network so that the 10 outputs can be interpreted as probabilities of each class. Recall that the softmax function is

$$p\left(y_i\right) = \frac{\exp\left(z_i\right)}{\sum_{j=1}^{J} \exp\left(z_j\right)}$$

   .
5. Implement the Conv2D operator on your own. And modify your Multi-Layer Perceptron into a CNN.
6. You can artificially create more training examples, by applying small transformations (translations, rotations, resizing, etc.) to the original images. But do not use other datasets.
7. Visualize your model weights(especially the convolution kernels) and see if you can read out something.