CSCI 5521: Machine Learning Fundamentals (Fall 2022)[1]

Homework 3 (Friday, Nov. 4)

Due on Gradescope at 11:59 PM, Friday, Nov. 18

Instructions:

- This homework has 3 questions, 100 points, on 3 pages.
- **For each questions, especially programming, please provide the detailed explanation and corresponding answer to the questions to avoid point reduction. Only submitting the source code without comprehensive pdf file is not enough to get all the points.**

1. (**40 points**) In this programming exercise, you will implement the forward and backward propagation for the multi-layer perceptron (MLP). You will use the handwritten digit dataset that is used for training various artificial intelligence systems. We help you to split the training/validation/test data to the corresponding files. For each file, the first 64 columns are the features, and the last columns are the corresponding labels. You need to train your MLP with optdigits_train.txt, tune the hyper-parameters with optdigits_valid.txt and test the prediction performance after select the best hyper-parameters with optdigits_test.txt. Before you start to train a MLP, you need to normalize all the 64 columns of the feature, *e.g.*,$X_{norm,i} = \frac{X_i - \mu_{train,i}}{\sigma_{train,i}}$, $i = 1, 2, \cdots, 64$. Note that $\mu_{train,i}$ and $\sigma_{train,i}$ represent the mean and the standard deviation of i-th feature in the training set. When you normalize the validation set and the test set, you need to use $\mu_{train,i}$ and $\sigma_{train,i}$.

   (a) (**20 points**) Implement a 1-layer MLP for classifying the provided data with 10 digits. You may need to refer to the slide of Chapter 10 for the MLP algorithm. In this case, you can use the sigmoid function as the activation function ($sigmoid(x) = \frac{1}{1+\exp(-x)}$) for this hidden layer. For the output, you can use the softmax function as the output layer activation function ($softmax(x_i) = \frac{\exp(x_i)}{\sum_j \exp(x_j)}$). The objective function is the cross-entropy loss

$$E(\boldsymbol{r}^t, \boldsymbol{y}^t) = -\sum_{i=1}^{K} \boldsymbol{r}_i^t \log \boldsymbol{y}_i^t \tag{1}$$

   More specifically, assume that we have a size of N training dataset $(\boldsymbol{x}^t, \boldsymbol{r}^t)$, $t = 1, \cdots, N$, where $\boldsymbol{r}^t \in \mathbb{R}^{K\times 1}$ is a one-hot vector with a totally K different category (for handwritten digit dataset, $K = 10$), and $\mathbb{R}$ a real number set. It means that if the feature vector $\boldsymbol{x}^t \in \mathbb{R}^{D\times 1}$ belongs to class k, $\boldsymbol{r}_k^t = 1$, and the remaining element of $\boldsymbol{r}^t$ is equal to 0, where D is the dimension of the feature vector (for this assignment, $D = 64$).

   In this sub-problem, we would like to tune the hyper-parameter in the hidden layer with $H = 5, 10, 15, 20, 25$ hidden units for this 1-layer MLP. **You need to report the validation accuracy v.s. the number of hidden units $H = 5, 10, 15, 20, 25$. Determine how many hidden units we should use to obtain the best performance. At last, report the accuracy on the test set with the selected best number of hidden units.**

   Hint: With $y_i = softmax(x_i) = \frac{\exp(x_i)}{\sum_j \exp(x_j)}$, the derivative is $\frac{\partial y_i}{\partial x_j} = y_i(\delta_{ij} - y_j)$, where $\delta_{ij} = 1$ if $i = j$, and $\delta_{ij} = 0$ otherwise. With $y = sigmoid(x) = \frac{1}{1+\exp(-x)}$, the derivative is $\frac{\partial y}{\partial x} = y(1-y)$.

[1] Instructor: Kshitij Tayal (tayal@umn.edu). TA: Xianyu Chen, and Yoshitaka Inoue (csci5521@umn.edu).

(b) (**20 points**) In this sub-problem, we would like to tune another hyper-parameter called the type of activation functions of the hidden layer (*e.g.,*sigmoid function, Relu function, tanh function). For this problem, we set the number of Hidden units as 10. **You need to report the validation accuracy v.s. the type of activation functions. Determine which activation function we should use to obtain the best performance. At last, report the accuracy on the test set by the selected best activation function.**
Hint: With $y = tanh(x) = \frac{\exp(x)-\exp(-x)}{\exp(x)+\exp(-x)}$, the derivative is $\frac{\partial y}{\partial x} = 1-y^2$. With $y = ReLU(x) = \max\{0, x\}$, the derivative is $\frac{\partial y}{\partial x} = 1$ if $x > 0$, and $\frac{\partial y}{\partial x} = 0$ if $x < 0$.

2. (**30 points**) In this programming exercise, you will utilize PyTorch to implement a neural network for a classification problem with the provided data in Problem 1. Currently, Pytorch has become the most popular deep-learning library. Here, you can refer PyTorch example for how to set up a whole classifier training pipeline in Link. We recommend to go over the provided example before answering this question. We expect that your best model can achieve at least 90% test accuracy.

   Here are some useful PyTorch materials you may use MNIST Example, PyTorch Tutorials (Learn the Basics, Introduce to PyTorch - YouTube, Learning PyTorch):

   (a) (**10 points**) For this problem, we would implement the stopping criteria. More specifically, the training procedure would terminate if we observe that the accuracy of the validation set has not been improved in 10 consecutive epochs. **Report the implementation detail and your explanation in your answer.**

   (b) (**5 points**) In this sub-problem, you will be asked to implement $L = 1, 2, 3, 4$ hidden layer for MLP. In this case, you can use the Relu activation function. **You need to report the validation accuracy v.s. the number of the hidden layer. Determine which number of the hidden layer we should use to obtain the best performance. At last, report the accuracy on the test set by the selected best number of the hidden layers.**

   (c) (**5 points**) In this sub-problem, we would like to tune another hyper-parameter called the type of activation functions of the hidden layer (*e.g.,*sigmoid function, Relu function, tanh function). For this problem, we set the number of Hidden units as 256 and 128 for the first and second hidden layers with the number of hidden layer $L = 2$. **You need to report the validation accuracy v.s. the type of activation functions. Determine which activation function we should use to obtain the best performance. At last, report the accuracy on the test set by the selected best activation function.**

   (d) (**10 points**) We want you to play with different hyperaparameters of the neural network. **Tell us about different hyperparamaters you tried, the accuracy you obtained, and what you learned from doing these experiments.** [For example, you can tune the learning rate, batch size, dropout, and/or l_1/l_2 regularization.] **You need to add the corresponding code in Q2.py for (d) part.** You can refer to the style for (b) and (c) parts.

3. (**30 points**) In this programming exercise, you will utilize PyTorch to implement a neural network for a classification problem with CIFAR-10, dataset consisting of 60000 32×32 color images in 10 classes, with 6000 images per class. We expect that your best model can achieve at least 70% test accuracy. For this problem, we do not provide the corresponding template. The reason is that we

hope you can write deep learning pipeline from scratch. You can refer to code style in Problem 2 and refer to Link

(a) (**15 points**) **Load the CIFAR-10 dataset into workspace.** For your information, you can also refer to Link. After loading and normalizing the data, you are asked to implement a 2d convolutional neural network. You can determine the network architecture and activation functions by yourself. Since the CIFAR-10 dataset only contains the train and test set. You can use the test set to select your best hyper-parameters. And then, **with these selected best hyper-parameters, you can report the prediction accuracy on the test set.**

(b) (**15 points**) **Document your hyperparameter search, different things you tried and your overall experience with convolution neural network.**

Submission

- Additional Instructions: You need to submit the code written in Python instead of any other programming language. Other programming languages are not accepted and would cause credit loss. We provide the template including the corresponding inputs and the outputs of the function, you need to fill the function and generate the expected results.
- For all of the programming, you can use the CPU for calculation. You can use the PyTorch $\geq$ 1.10.0.
- Things to submit:
  - `hw3_sol.pdf`: the document including all of your answers to the problem including the summary or figure of the programming problems. The PDF format is accepted. If you would like to use the hand-written document, you can scan it and make sure that the scanned copy is clearly readable. If it is difficult to read, credit would be lost. But you need to attach the figure or results generated from the code to the pdf instead of drawing a figure manually.
  - `MySimpleMLP.py`: a text file containing the python function for Problem 1. Use the skeleton file `MySimpleMLP.py` and then fill in the missing parts.
  - `MyTorchMLP.py`: a Python source file for Problem 2. Use this skeleton file `MyTorchMLP.py` and fill in the missing parts.
  - `Q2.py`: a Python source file for Problem 2. Fill in the missing parts for Problem 2(d).
  - `MyCifarNet.py`: a Python source file for Problem 3. It is an empty template, please complete it.
  - `Q3.py`: a Python source file for Problem 3. It is an empty template, please complete it.
- Submit: You need to submit the materials electronically to the Gradescope. **For This assignment, you need to submit the materials [Hw3-Written (for `hw3_sol.pdf`) and the HW3 programming (the compressed file such zip file including the Python codes)].** We will grade your code in vanilla Python.