Brandeis University | COSI-165B | Deep Learning
Assignment 2 (100 points)
Due Date: Mar 20, Sunday, 23:59 PM (EST)

Instruction of homework submission

1. Submit your solutions in one pdf file named cosi165b- assignment -2-[yourname].
2. Submit your code files in one zip file named cosi165b-assignment-2-code-[yourname]. The code should include necessary comment.
3. Late policy: (late hours / 24) * 10%, that is one day late leads to 10% loss. If late hours > 7 * 24 (one week), it will be 100% loss.
4. There may be code of similar problems available on the internet (e.g., Github). It is okay to refer those code while you should finish your own code. If we find your code is the same as them, you will get 0 (100% loss) for this assignment.
5. Please obey academic integrity, no plagiarize will be allowed. Plagiarize will lead to serious penalties, e.g., failure on the assignment or failure in the course.
6. For any question, please contact TA (chunhuizhang@brandeis.edu) or instructor (chuxuzhang@brandeis.edu).

Problem A – Convolutional Neural Network (70 pts)

In this problem, you will build a convolutional neural network model for image classification (binary classification) using Pytorch. The data is the same with the first assignment. You will need to finish code in cnn_main.py and cnn_utils.py, and answer some questions.

Part I: Model (40 pts)

PA-I-1 (20 pts): Finish Net class for constructing CNN model in cnn_utils.py.
CNN model detail: 5 layers in total. The first 2 layers are convolution layers with kernel size = 5*5, stride = 1, output channel numbers = 6, 12. The filter size of max pooling layer after each convolution layer is 2*2. The last 3 layers are fully connected layers with output hidden number = 120, 64, 2, respectively. Use relu activation in each hidden layer and sigmoid activation in the last layer. Use default parameter initialization in Pytorch.

PA-I-2 (10 pts): Finish model_train function for CNN model training (using train data) with Adam optimization in cnn_main.py. Set batch size = 5.

PA-I-3 (10 pts): Finish model_test function for model testing (using test data) in cnn_main.py.

Part II: Performance (30 pts)

PA-II-1 (10 pts): Run your code and report accuracy over test data (Hint: accuracy is over 80%, result is usually not stable), show screenshot of your result.

PA-II-2 (10 pts): Set epoch number = 100, plot learning curves (test accuracy w.r.t. epoch number) of three optimization algorithms (i.e., SGD, Adagrad, and Adam) and compare them. You can choose different learning rates to make performances of different algorithms good.

PA-II-3 (10 pts): Use dropout strategy after the second convolution layer and the second fully connected layer (dropout ratio = 0.2). Show your revised code of Net class. Then redo PA-II-2.

Problem B – Residual Neural Network (30 pts)

In this problem, you will build a residual convolutional neural network (RCNN) to solve the same problem as problem A. You will need to finish code in rcnn_main.py and rcnn_utils.py, and answer some questions.

Part I: Model (20 pts)

PB-I-1 (20 pts): Finish Net class for constructing RCNN model in rcnn_utils.py. RCNN model detail: 7 layers in total. The first layer is convolution layer with kernel size = 5*5, stride=1, output channel number = 6. The second and third layers are convolution layers with one residual connection, kernel size = 1*1, stride = 1, output channel numbers = 6. The fourth layer is convolution layer with kernel size = 5*5, stride = 1, output channel number = 12. The filter size of max pooling layer after the first and fourth convolution layer is 2*2. The last 3 layers are fully connected layers with output hidden number = 120, 64, 2, respectively. Use relu activation in each hidden layer and sigmoid activation in the last layer. Use default parameter initialization in Pytorch.

Use model_train and model_test functions (same as Problem A) with Adam optimization in rcnn_main.py. Set batch size = 5.

Part II: Performance (10 pts)

PB-II-1 (5 pts): Run your code and report accuracy over test data (Hint: accuracy is over 85%, result is usually not stable), show screenshot of your result.

PB-II-2 (5 pts): Set epoch number = 100, plot learning curves (test accuracy w.r.t. epoch number) of three optimization algorithms (i.e., SGD, Adagrad, and Adam) and compare them. You can choose different learning rates to make performances of different algorithms good.