CS 260 Homework 1

Wenlong Xiong (204407085)

February 7, 2019

1 Introduction

In this assignment, I used PyTorch to implement a Logistic Regression and a Linear SVM model (without regularization), and trained and tested them on a subset of the MNIST dataset with just 0's and 1's to learn about binary classification. To optimize the model, I used stochastic gradient descent (with and without momentum) for different learning rates. The learning rates I tested were **learning rate** $= 10^{-k}, k = [2, 3, 4, 5, 6]$. Since the assignment specification did not explicitly ask to try different momentum values, I set it to **momentum = 0.9** and did not change it. I recorded the average training loss for each of my models over 10 epochs, as well as the final test accuracy.

2 Metrics

We define the average training loss in each epoch as the following:

$$\frac{1}{B}\sum_{b=1}^{B}\sum_{d=1}^{D_b}\frac{1}{D_b} loss(\text{labels}_{b,d}, \text{model}_b(\text{data}_{b,d}))$$

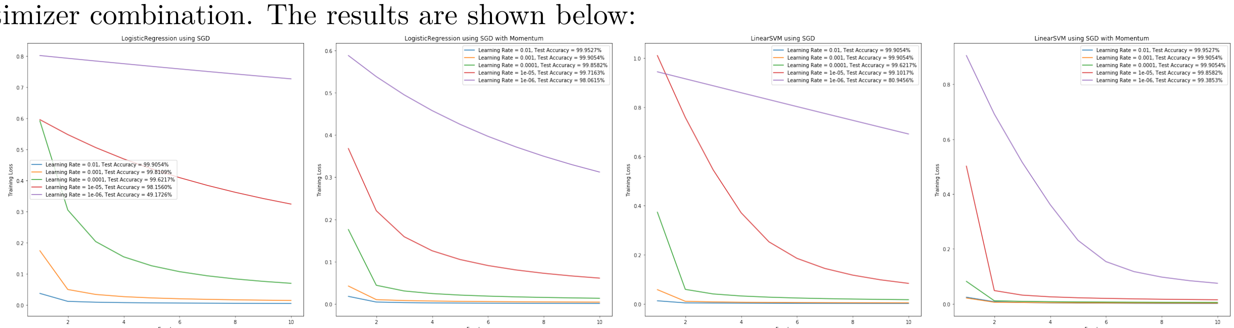
B is the number of batches, D_b is the number of data points in batch b, and $\text{model}_b(\text{data}_{b,d})$ is the prediction by the model after batch b on the current batch of data.

3 Results

To compare the classification accuracy of Logistic Regression vs Linear SVM, I ran these two models with the SGD optimizer (both with and without momentum=0.9), at a learning rate of 0.0001 for 10 epochs. The training loss is recorded below for the 4 combinations, along with the final model accuracy when applied to the test set. From this, we can see that in general, momentum helps a model converge more quickly, and reduce training loss while improving test accuracy (this is shown by a higher test accuracy and lower training loss when comparing the two optimizers within a single model). We also see that in general, LinearSVM has similar if not better performance given the same learning rate (this is shown by LinearSVM / SGD + Momentum having a higher test accuracy than Logistic Regression / SGD + Momentum). The model with the best test accuracy was Linear SVM and SGD + Momentum, for a learning rate of 0.0001 and momentum of 0.9. The **test accuracy was 99.9054 %** classified correctly.

	Logistic Regression		LinearSVM	
Epoch	SGD	SGD + Momentum	SGD	SGD + Momentum
1	0.5920	0.1762	0.3736	0.0818
2	0.3051	0.0443	0.0594	0.0110
3	0.2035	0.0309	0.0407	0.0089
4	0.1544	0.0248	0.0325	0.0077
5	0.1257	0.0211	0.0276	0.0068
6	0.1067	0.0187	0.0242	0.0062
7	0.0933	0.0169	0.0218	0.0058
8	0.0833	0.0156	0.0199	0.0055
9	0.0754	0.0145	0.0185	0.0052
10	0.0692	0.0136	0.0173	0.0050
Test Acc.	99.6217 %	99.8582 %	99.6217 %	99.9054 %

I also tried using different learning rates to see how the step size would affect model learning. I tested the learning rates $= 10^{-k}$, where $k = [2, 3, 4, 5, 6]$, for each model + optimizer combination. The results are shown below:



In general, we can see that with larger step sizes (larger learning rates), the training loss tends to converge much more quickly. This makes sense, because if we take larger steps in the direction of the gradient towards the absolute minimum of the loss function, we'll reach the minimum in fewer steps. If we take smaller steps, our models will take longer to converge. However, if our steps are too large, the gradient descent algorithm can overshoot the minimum and diverge (and if the steps are too small, the algorithm may reach a local minimum and never escape it). It also makes sense that optimizer with momentum performed better than pure stochastic gradient descent; momentum accelerates gradients in the direction of the moving average of the past gradients, which lets the model converge more quickly. Since the models with larger learning rates and momentum have more fully converged to the local minimum of the loss function by the end of 10 epochs, they have a higher test accuracy.