UNIVERSITY OF VICTORIA

Department of Electrical and Computer Engineering

ECE 503 Optimization for Machine Learning

PROJECT REPORT

Title: Performance Analysis Using Different Cost Functions, Optimization Algorithms, and Features of Input Data.

Names: Huan-Rui Zhang

1. Introduction

The motivation of this project is to understand the pros and cons of different cost functions, optimization algorithms, and feature of input data. The dataset used in this project is the MNIST database. It consists of 10 classes of the hand-written digit [1]. Each one has a corresponding label between 0 to 9. The difference between this project and the lab 4 is that I want to understand the performance of optimization algorithms in different cost functions and in different feature of input data. Below lists the cost functions, optimization algorithms, and features of input data used in this project.

Cost functions:

- Softmax cost functions without regularization term
  - $E(\widehat{w_1}, \widehat{w_2}, \dots, \widehat{w_{10}}) = \frac{1}{P}\sum_{p=1}^{P} log\left(\sum_{j=1}^{K} e^{\widehat{w_j}^T \widehat{x_p}}\right) - \sum_{j=1}^{K}\left(\widehat{w_j}^T \bar{x}_j\right)$
- Softmax cost functions with regularization term
  - $E(\widehat{w_1}, \widehat{w_2}, \dots, \widehat{w_{10}}) = \frac{1}{P}\sum_{p=1}^{P} log\left(\sum_{j=1}^{K} e^{\widehat{w_j}^T \widehat{x_p}}\right) - \sum_{j=1}^{K}\left(\widehat{w_j}^T \bar{x}_j\right) + \frac{\mu}{2}\sum_{\boldsymbol{j=1}}^{\boldsymbol{K}}\left\|\widehat{\boldsymbol{w_j}}\right\|_{\boldsymbol{2}}^{\boldsymbol{2}}$

Optimization algorithms:

- Gradient Descent
- BFGS algorithm
- Conjugate Gradient

Input Data:

- Original data

- HOG feature of data

2. Problem Formulation

As the computer do not have the ability to look the hand-written digit in paper. We first formulate the hand-written digit into a machine learning problem, we need to scan the hand-written number into an image. An image is consisted of an array of intensity in different intensity of each pixel. In this project, the image only needs to be grayscale is enough. The range of a grayscale image can vary by applications, we should normalize, therefore, the grayscale image into the range between 0 to 1 by dividing the original grayscale image by its maximum value. As aforementioned, an image is an array of digits. In machine learning field, reshaping the input data from a two-dimensional array into one-dimensional vector is widely used. The label of a training data is important in supervised learning. After finished the preparation of training data, we will label each training data by giving different label number. Until now, we have the training data and the corresponding labels. The preparation of testing data is the same as training data, but the difference is that testing dataset can only be used in testing phase rather than training phase. The testing data is used to evaluate the performance of the trained model. If the model is overfitted to the training dataset, the accuracy of testing dataset will be low even the training loss value is very small in training phase. In that case, we should modify the model to mitigate the overfitting problem.

3. Solution Methods

The training dataset and testing dataset in this project are the same as in lab 4 [1] but will be applied to deal with the two different cost functions, three different optimization algorithms, and two different input data formats. Each image in the MNIST dataset is in 28*28 pixels and in grayscale. The training dataset has 1600 samples for each class of digit, while the testing dataset has 1000 samples for each class of digit.

To utilize the HOG feature for training and testing phase, I reused the HOG feature data generated in lab 4, which were reduced to 24*24 amount of data.

As the dataset has ten classes of label, I need to use the dataset to train ten different model for classifying the input data. The predicted class of the input data will be the class has the largest value among the ten models. The precise formula is listed below, which is referred from lab 4 either.

$j^* = arg\left(\max_{1\le j\le K} \widehat{w_j}^{*T} \hat{x}\right)$, where $\hat{x} = \begin{bmatrix} x \\ 1 \end{bmatrix}$ and $\widehat{w_j}*$ are the optimum model obtained by minimizing the cost function listed in section 1.

4. Computer Simulations and Numerical Results

To compare the performance of the obtained models, four metrics are considered: classification accuracy, iteration counts, step error ε, and training time. The classification accuracy is calculated from the confusion matrix. The iteration counts are the iteration number that the algorithm runs until step error ε less than 1e-6 or stopped by me due to hard to decrease in the step error. The training time is the CPU time for running the training phase.

The comparison of classification accuracy is listed in Table 1. By adding the regularization term, the classification accuracy can be improved in most of the case. Moreover, the classification accuracy can be further increased with the benefit of using HOG feature as its input data.

The underlined classification accuracies of gradient descent algorithm with $\mu=0$ in original data sets, gradient descent algorithm with $\mu=0$ in HOG data sets, and conjugate gradient algorithm with $\mu=0$ in HOG data sets are not aligned to my expectation as they should lower than the case with $\mu=0.002$. To clarify about these three numerical results, I ran these three case several times and got the same results. I think, however, the value is still in a reasonable range.

Table 1. Classification Accuracy

Classification Accuracy	Original data sets		HOG of the data sets	
	$\mu=0$	$\mu=0.002$	$\mu=0$	$\mu=0.002$
Gradient Descent	<u>91.94%</u>	91.77%	<u>98.18%</u>	97.74%
BFGS	90.07%	91.77%	97.71%	97.74%
Conjugate Gradient	91.77%	91.77%	<u>98.14%</u>	97.74%

Table 2 shows the iterations counts for each algorithm. The BFGS algorithms have the lowest iteration counts for each setting. The iteration counts for gradient descent algorithms or conjugate gradient descent algorithms are at least 15 times of BFGS has.

Table 2. Iteration Counts

Iteration counts	Original data sets		HOG of the data sets	
	μ =0	μ =0.002	μ =0	μ =0.002
Gradient Descent	5873	3834	4222	3658
BFGS	240	224	237	196
Conjugate Gradient	8011	3811	6364	3656

The step errors of each algorithm are summarized in table 3. All the three algorithms can converge to less than 1e-6 step error in the case with HOG data set μ=0.002. Most of the algorithms in either original data sets or HOG data sets are very hard to reduce its step error in the case μ=0; this is good evidence to show the advantage in optimization problems by adding a suitable regularization term.

Table 3. Step Errors

step errors	Original data sets		HOG of the data sets	
	μ =0	μ =0.002	μ =0	μ =0.002
Gradient Descent	0.001964886 (very hard to decrease)	0.000000998	0.002032955 (very hard to decrease)	0.000000998
BFGS	0.334247124 (very hard to decrease)	0.000000651	0.000006148421 81678988	0.000000528
Conjugate Gradient	0.001964886 (very hard to decrease)	0.000000997	0.001545947961 93322 (very hard to decrease)	0.000000999

The training time is measured the CPU time in the training phase, as listed in table 4. The BFGS algorithm have the shortest training time. The training times of gradient descent algorithm and conjugate gradient are almost 10 times of the BFGS's training time. Turning to the training time of HOG data set with BFGS algorithm, the training time in $\mu=0$ is less than that in $\mu=0.002$; this is also a weird numerical result. In my imagination, I think the regularization term helps the optimization performance, then the training time should be reduced. I think maybe the addition calculation of the regularization term in the optimization increase the training time.

Table 4. Training Time

Training times (Unit: Seconds)	Original data sets		HOG of the data sets	
	$\mu=0$	$\mu=0.002$	$\mu=0$	$\mu=0.002$
Gradient Descent	25223	16322	21110	13717
BFGS	1340	1795	1015	1228
Conjugate Gradient	29761	20310	21464	14414

The step errors of the training phase for $\mu=0$ in original data set and HOG data set are illustrated in figure 1. The step error is strictly decreasing by using the HOG data set. The step error of conjugate gradient algorithm reduced to the lowest step errors among others after 8^{th} iteration. The step error of BFGS algorithms have relatively big fluctuations compared to others.

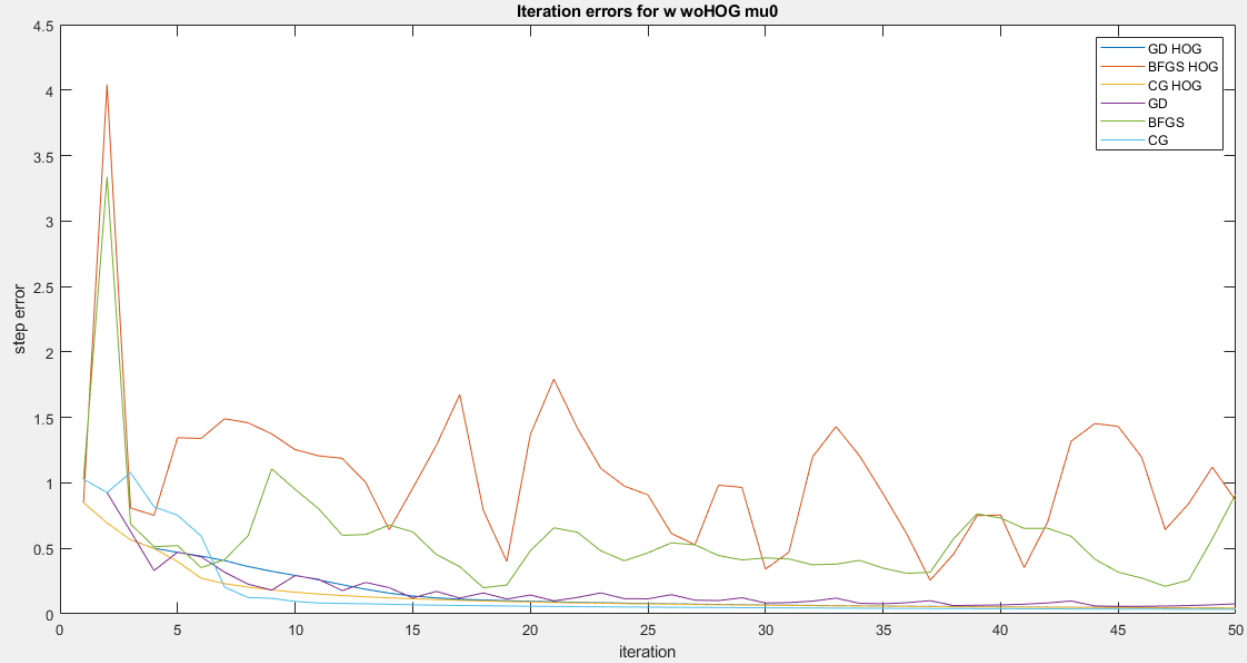

Figure 1. illustration for the step errors of the training phase for μ =0 in original data set and HOG data set.

The fluctuation of step error is mitigated by adding the regularization term with μ=0.002, as shown in figure 2. This may because the cost function with a suitable regularization term is rounder, making the algorithm more efficient to find the best solution in each iteration. The step error of BFGS algorithm have been improved with μ=0.002.

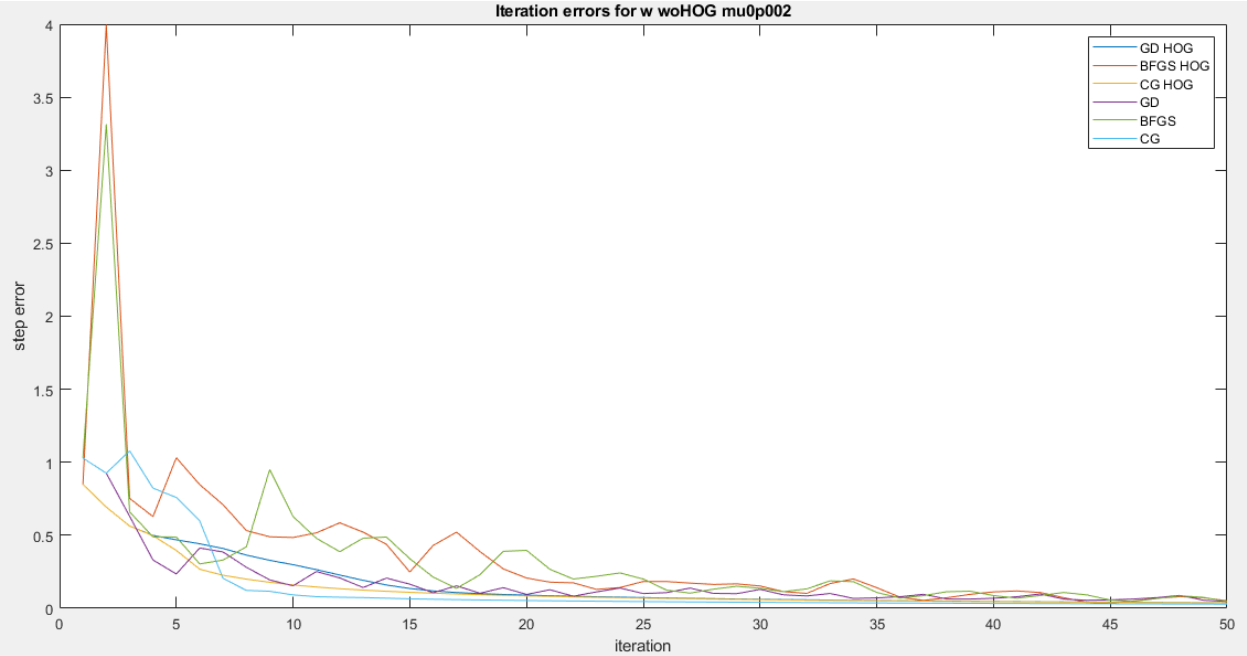

Figure 2. illustration for the step errors of the training phase for μ =0.002 in original data set and HOG data set.

The confusion matrices for each case in this project are shown in below tables. A conclusion of this project is given at the end of this report.

Confusion matrix for Gradient Descent with μ =0

959	0	11	1	1	8	11	3	5	10
0	1111	10	1	3	4	3	6	9	7
1	3	920	24	5	3	9	21	13	0
2	2	16	911	2	36	2	7	26	11
0	0	5	1	906	10	10	7	8	28

6	1	6	28	0	774	15	1	33	6
6	4	12	3	10	14	904	0	12	0
3	2	10	12	6	5	3	947	9	25
1	12	37	21	9	29	1	2	847	7
2	0	5	8	40	9	0	34	12	915

Confusion matrix for Gradient Descent with μ =0.002

961	0	11	2	1	10	9	3	7	12
0	1108	7	1	6	4	3	11	10	8
0	2	906	20	3	2	4	19	9	2
2	2	16	912	1	35	2	8	22	12
1	1	11	0	909	12	11	6	7	37
4	3	2	29	0	769	13	1	25	7
10	4	15	2	12	17	911	0	13	0
1	1	17	15	2	7	3	942	12	22
1	14	38	18	9	26	2	2	858	8

0	0	9	11	39	10	0	36	11	901

Confusion matrix for BFGS with μ =0

937	0	9	4	3	11	13	3	9	8
0	1101	17	0	5	4	2	8	10	6
3	10	900	23	11	7	16	17	13	1
3	2	26	887	2	42	1	14	32	9
2	0	6	2	895	11	9	6	10	35
13	1	6	42	0	748	17	1	42	6
14	4	15	2	6	15	896	1	18	0
5	2	13	11	9	4	2	940	12	36
2	13	35	25	10	37	1	0	813	18
1	2	5	14	41	13	1	38	15	890

Confusion matrix for BFGS with μ =0.002

961	0	11	2	1	10	9	3	7	12

0	1108	7	1	6	4	3	11	10	8
0	2	906	20	3	2	4	19	9	2
2	2	16	912	1	35	2	8	22	12
1	1	11	0	909	12	11	6	7	37
4	3	2	29	0	769	13	1	25	7
10	4	15	2	12	17	911	0	13	0
1	1	17	15	2	7	3	942	12	22
1	14	38	18	9	26	2	2	858	8
0	0	9	11	39	10	0	36	11	901

Confusion matrix for Conjugate Gradient with μ =0

953	0	11	2	1	8	12	3	5	10
0	1110	10	1	4	4	3	5	9	7
1	4	920	21	7	3	10	21	13	0
3	3	17	909	2	35	1	8	27	11
1	0	6	1	904	10	10	7	8	27

8	1	5	35	0	773	15	1	33	6
8	4	11	2	7	15	904	0	14	0
3	2	10	12	6	6	2	947	10	26
1	11	37	21	11	29	1	2	843	8
2	0	5	6	40	9	0	34	12	914

Confusion matrix for Conjugate Gradient with μ =0.002

961	0	11	2	1	10	9	3	7	12
0	1108	7	1	6	4	3	11	10	8
0	2	906	20	3	2	4	19	9	2
2	2	16	912	1	35	2	8	22	12
1	1	11	0	909	12	11	6	7	37
4	3	2	29	0	769	13	1	25	7
10	4	15	2	12	17	911	0	13	0
1	1	17	15	2	7	3	942	12	22
1	14	38	18	9	26	2	2	858	8

0	0	9	11	39	10	0	36	11	901

Confusion matrix for Gradient Descent with HOG and μ =0

973	0	4	0	0	2	2	1	3	5
0	1125	1	0	1	1	3	5	1	4
1	1	1007	1	1	1	0	12	0	1
0	5	7	998	0	3	0	3	3	6
0	0	0	0	967	0	2	3	2	9
2	0	0	5	0	881	6	0	0	2
1	0	3	0	4	1	942	0	2	0
1	1	5	2	2	0	0	997	2	5
2	3	5	2	2	2	2	2	957	6
0	0	0	2	5	1	1	5	4	971

Confusion matrix for Gradient Descent with HOG and μ =0.002

974	0	7	1	0	2	5	1	4	8

0	1124	1	0	2	1	3	6	1	4
1	1	1002	1	1	1	0	12	1	2
0	4	4	992	0	5	0	3	6	5
0	0	0	0	964	0	3	3	3	9
2	0	0	6	0	878	2	0	2	3
1	2	2	0	6	2	939	0	1	0
1	3	8	4	0	0	0	991	5	6
1	1	8	5	2	3	5	2	945	7
0	0	0	1	7	0	1	10	6	965

Confusion matrix for BFGS with HOG and μ =0

971	1	5	0	0	1	3	1	3	4
0	1119	2	1	1	0	4	3	0	1
1	2	1006	2	1	1	1	13	1	0
0	4	10	998	0	4	2	10	3	10
0	1	0	0	967	0	4	0	2	14

2	0	0	6	0	877	7	0	0	4
2	0	2	0	3	2	930	0	2	0
0	2	2	0	5	0	0	993	1	9
2	6	5	3	3	6	5	4	959	16
2	0	0	0	2	1	2	4	3	951

Confusion matrix for BFGS with HOG and μ =0.002

974	0	7	1	0	2	5	1	4	8
0	1124	1	0	2	1	3	6	1	4
1	1	1002	1	1	1	0	12	1	2
0	4	4	992	0	5	0	3	6	5
0	0	0	0	964	0	3	3	3	9
2	0	0	6	0	878	2	0	2	3
1	2	2	0	6	2	939	0	1	0
1	3	8	4	0	0	0	991	5	6
1	1	8	5	2	3	5	2	945	7

0	0	0	1	7	0	1	10	6	965

Confusion matrix for Conjugate Gradient with HOG and μ =0

973	0	3	1	0	2	2	1	3	5
0	1124	1	1	1	1	3	5	1	3
2	1	1008	1	1	1	0	12	0	1
0	5	7	996	0	3	0	3	3	6
0	0	0	0	966	0	2	3	2	10
1	0	0	5	0	880	6	0	0	2
1	0	3	0	4	1	942	0	2	0
1	1	5	2	2	0	0	998	2	6
2	4	5	2	2	3	2	2	957	6
0	0	0	2	6	1	1	4	4	970

Confusion matrix for Conjugate Gradient with HOG and μ =0.002

974	0	7	1	0	2	5	1	4	8

0	1124	1	0	2	1	3	6	1	4
1	1	1002	1	1	1	0	12	1	2
0	4	4	992	0	5	0	3	6	5
0	0	0	0	964	0	3	3	3	9
2	0	0	6	0	878	2	0	2	3
1	2	2	0	6	2	939	0	1	0
1	3	8	4	0	0	0	991	5	6
1	1	8	5	2	3	5	2	945	7
0	0	0	1	7	0	1	10	6	965

5. Conclusion

I analysed the pros and cons of different cost functions, optimization algorithms, and feature of input data in this project. This is a great chance to have a comprehensive comparison, even the total training time is over 46 hours. With the experience learnt in this project, I have more solid understanding about different cost functions, optimization algorithms, and feature of input data. I can trade-off between performance and accuracy by choosing the suitable scheme in the future.

6. References

[1] Wu-Sheng Lu, UVic ECE503 Sep. 2023 course lab manual published in May 2021.