A Technical Report about SAG,SAGA,SVRG

Tao Hu
Department of Computer Science
Peking University
No.5 Yiheyuan Road Haidian District, Beijing, P.R.China
taohu@pku.edu.cn

Notice: this is a course project for <Algorithms for Big Data Analysis> conducted by Zaiwen Wen at Peking University; and <Deep Learning> conducted by Zhihua Zhang at Peking University.

- **for <ALgorithms for Big Data Analysis>, I completed the SAG, SAGA, SVRG algorithm programming.**
- **for <Deep Learning>, I completed different gradient method ablation study.**

I just write it as a entirety for for completeness and continuity!

Abstract

The project report mainly includes the ablation study about the regularization term, stochastic gradient method, etc.

1 Overview

There exists many gradient method, Full Gradient(FG) is proposed first. as more and more data generate, a light-weight method named Stochastic Gradient(SG) appears. Based on the SG method, there emerges several Stochastic Method: Stochastic Gradient Method(SGD)[1], Momemtum[5], AdaDelta[9], RmsProp[8], SDCA[7]. Recently, NL Roux[6] proposed a Variation Method Stochastic Average Gradient (SAG) that realizes linear convergence. lots of variety based on Variance Method such as SVRG[3] SAGA[2], appears. In this project report I will compare several aspects about some of these methods.

The problem I experiment on is regularized logistic regression.

$$min_w \quad P(w) = \frac{1}{n}\sum_{i=1}^{n} ln(1 + exp(-w^T x_i y_i)) + \lambda L(w)$$

where w is weight, L(w) means regression item which can be Ridge Regression or Lasso Regression.

as we use gradient method, we have the gradient formula:

$$\nabla P(w) = \frac{1}{n}\sum_{i=1}^{n} \frac{exp(-y_i w^T x_i)}{1+exp(-y_i w^T x_i)}(-y_i x_i) + \lambda L^{'}(w)$$

if I use *l1*-regularized logistic regression, then $L^{'}(w) = sign(w)$. otherwise, I will choose *l2*-regularized logistic regression which $L^{'}(w) = 2w$.

the dataset we use is mnist[4], so the x_i I use above is a 785-by-1 vector, y_i is a scalar, w is 1-by-785 vector. the mnist data size is 28-28, adding a bias totally is 785. for each mnist image, I have normalized it to 1 for convenience of training.as the logistic regression is a binary classification problem, hence I just divide the mnist data into even and odd for later study.

the final binary digit classfier is:

Author Info: Taohu ,Peking University

$$y = sign(w^T x)$$

y=1 means odd, otherwise means even.

2 Experiment

2.1 How to obtain the optimal

In the following Experiment, I will use a parameter named " Objective minus Optimum" which we need the optimum, I will choose the optimal value by SGD with a exponential decay learning rate:

$$\mu_t = \eta_0 a^{\lfloor \frac{t}{step} \rfloor}$$

η_0, a are the grid search parameter,here we choose the step = 3, which means to decay the learning rate exponentially for every 3 epochs. 100 epochs were processed finally. the grid search result is in Table 1.

as the result shows, the minimal is 0.3707, we keep the same grid search parameter where $\eta_0 = 1, a = 0.9$ and increase the final epochs to 200, then further achieve minimal value 0.3638/0.9018, we choose the optimum 0.3638 in the following experiment.

Table 1: Optimum Grid Search Table

a | η_0	10	1	0.1	0.01	0.001
0.9	0.4200/0.9013	**0.3707/0.9018**	0.4173/0.8955	0.5830/0.8457	1.0758/0.7315
0.7	0.4118/0.9037	0.3826/0.9	0.4764/0.8705	0.5966/0.8097	1.8617/0.6378
0.5	0.4616/0.8997	0.3887/0.8975	0.4878/0.8615	0.8215/0.7896	3.113/0.5356

the element in every grid means "loss/test accuracy", the result is selected by the highest test accuracy in 100 epochs. best optimal achieves when a=0.9,$\eta_0 = 1$

2.2 Ablation Study

2.2.1 Regulation

There exists different regulation items including L1 norm, L2 norm. here I will compare different regression factor λ in Figure 1. different regulation type influence on test accuracy is also compared in Figure 2.

in those Figures, I choose 100 epochs to run all the methods.

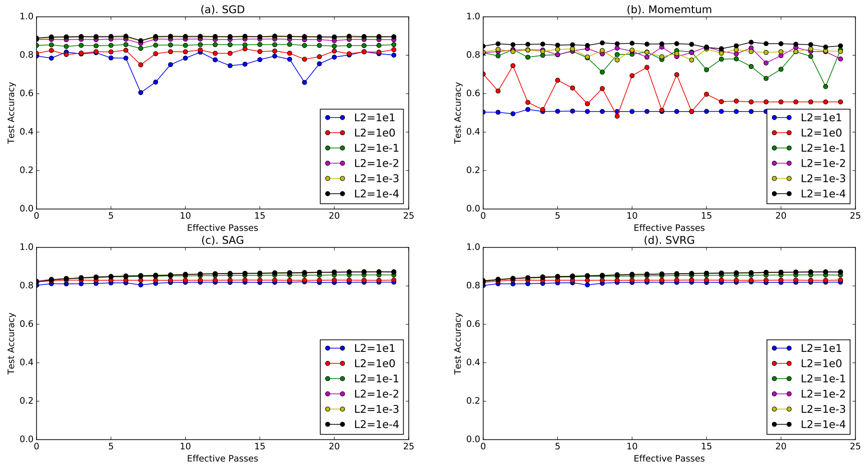

Figure 1: Regulation Value Experiment.all experiment use *l2*-regularized logistic regression as default. (a) Test Accuracy between different regulation λ with *SGD*. (b) Test Accuracy between different regulation λ with *SGD with Momentum*. (c) Test Accuracy between different regulation λ with *SAG*. (d) Test Accuracy between different regulation λ with *SVRG*.This figure is best viewed in colour.

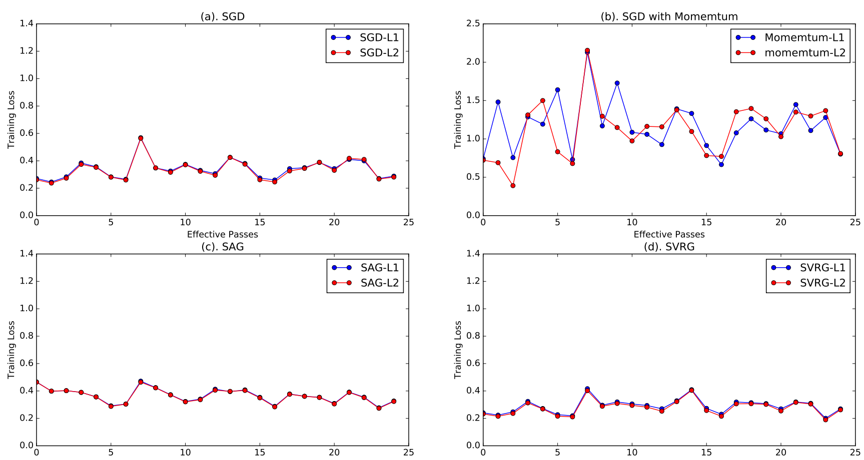

Figure 2: Regulation Type Experiment (a) Test Accuracy between *l1* and *l2* regularized logistic regression with *SGD*. (b) Test Accuracy between *l1* and *l2* regularized logistic regression *SGD with Momentum*. (c) Test Accuracy *l1* and *l2* regularized logistic regression with *SAG*. (d) Test Accuracy between *l1* and *l2* regularized logistic regression with *SVRG*.This figure is best viewed in colour.

2.2.2 Step Size Strategy

I mainly compared these step size strategy: fixed, exponential decay(step_size=3), backtracking. the training loss and test accuracy influenced by the step size strategy is displayed in Figure 3. 100 epochs are processed for every method.

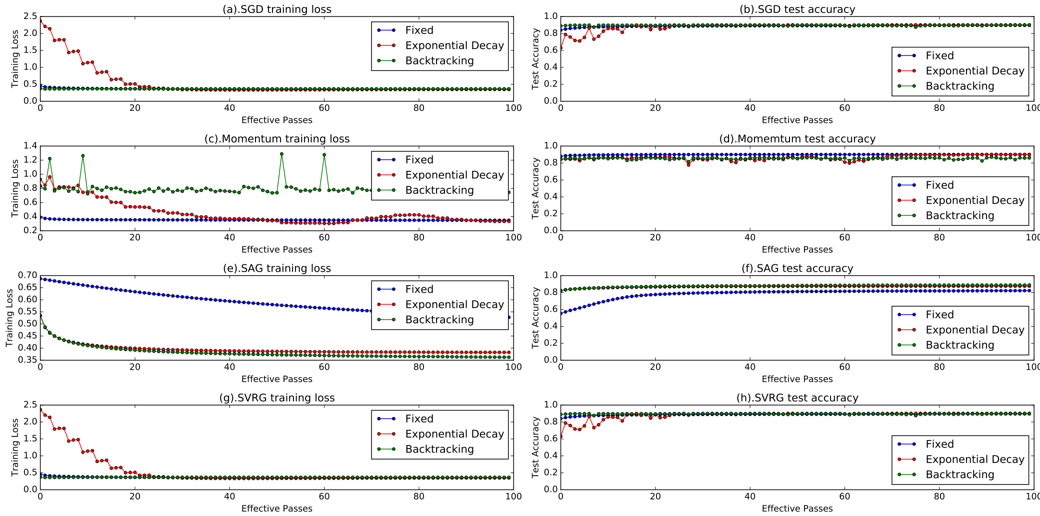

Figure 3: Step Size Strategy Experiment (a,b) training loss and test accuracy with three step size strategy with *SGD*. (c,d) training loss and test accuracy with three step size strategy *SGD with Momentum*. (e,f) training loss and test accuracy with three step size strategy with *SAG*. (g,h) training loss and test accuracy with three step size strategy with *SVRG*.This figure is best viewed in colour.

2.2.3 Different Method

Finally, different method including Stochastic Gradient Descend(SGD), SGD with Momentum, accelerated SGD with Momentum, SAG, SAGA, SVRG are compared in Figure 4. Training Loss, Validation Loss, Test Accuracy tendencies are listed.

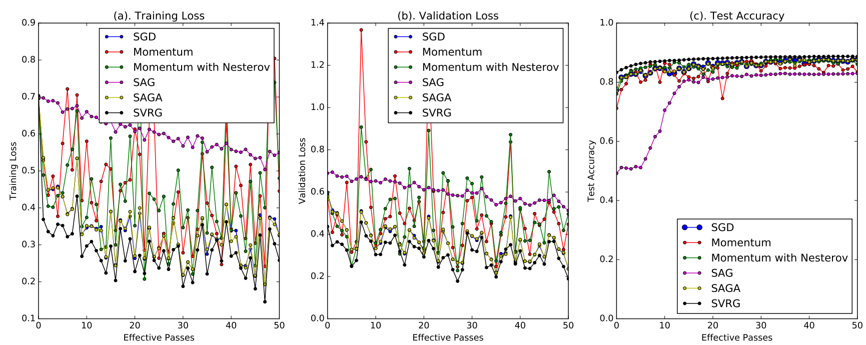

Figure 4: SG Method Experiment (a). training loss of different method. (b). validation loss of different method. (c). test accuracy of different method. This figure is best viewed in colour.

Time Consume Experiment is show in Figure **??**

2.2.4 Vectorization Programming

vectorization programming is very important in numerical calculation. it's a bad habit using "for loop" too frequently. here I will list two different writing style concerning the "non-vectorization" and "vectorization" programming.

vectorization is necessary especially in algorithm like SAG, SVRG, because the gradient computing is very frequently.

Algorithm 1 Non-vectorization

Require: initial value w,x,y

for i in 1:n **do**

calculate gradient:$\nabla P(w) = \frac{1}{n}\sum_{i=1}^{n}\frac{exp(-y_i w^T x_i)}{1+exp(-y_i w^T x_i)}(-y_i x_i) + \lambda L^{'}(w)$

end for

Output $\nabla(w)$

Algorithm 2 vectorization

Require: initial value w:d-1,x:n-d,y:n-1

$tmp = \frac{exp(-y\odot(xw))}{1+exp(-y\odot(xw)}$

$\nabla P(w) = x^T(tmp \odot (-y))$

Output $\nabla(w)$

3 Conclusion

Acknowledgments

This is only a technical report, I will appreciate it helps you.

References

References

[1] Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer, 2010.

[2] Aaron Defazio, Francis Bach, and Simon Lacoste-Julien. Saga: A fast incremental gradient method with support for non-strongly convex composite objectives. In *Advances in Neural Information Processing Systems*, pages 1646–1654, 2014.

[3] Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in Neural Information Processing Systems*, pages 315–323, 2013.

[4] Yann LeCun, Corinna Cortes, and Christopher JC Burges. Mnist handwritten digit database. *AT&T Labs [Online]. Available: http://yann. lecun. com/exdb/mnist*, 2, 2010.

[5] Ning Qian. On the momentum term in gradient descent learning algorithms. *Neural networks*, 12(1):145–151, 1999.

[6] Nicolas L Roux, Mark Schmidt, and Francis R Bach. A stochastic gradient method with an exponential convergence _rate for finite training sets. In *Advances in Neural Information Processing Systems*, pages 2663–2671, 2012.

[7] Shai Shalev-Shwartz and Tong Zhang. Stochastic dual coordinate ascent methods for regularized loss minimization. *Journal of Machine Learning Research*, 14(Feb):567–599, 2013.

[8] Tijmen Tieleman and Geoffrey Hinton. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4(2), 2012.

[9] Matthew D Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.