Scholars Journal of Physics, Mathematics and Statistics **ISSN 2393-8056 (Print)**
Sch. J. Phys. Math. Stat. 2017; 4(4):178-184 **ISSN 2393-8064 (Online)**
©Scholars Academic and Scientific Publishers (SAS Publishers)
(An International Publisher for Academic and Scientific Resources)

Adaptive Learning Rate SGD Algorithm for SVM

Shuxia Lu*, Zhao Jin
Key Lab. of Machine Learning and Computational Intelligence, College of Mathematics and Information Science, Hebei University, Baoding City , Hebei Province, 071002, China

***Corresponding author**
Shuxia Lu

Article History
Received: 14.10.2017
Accepted: 25.10.2017
Published: 30.10.2017

DOI:
10.21276/sjpms.2017.4.4.5

Abstract: Stochastic gradient descent (SGD) is a simple and effective algorithm for solving the optimization problem of support vector machine, where each iteration operates on a single training example. The run-time of SGD does not depend directly on the size of the training set, the resulting algorithm is especially suited for learning from large datasets. However, the problem of stochastic gradient descent algorithm is that it is difficult to choose the proper learning rate. A learning rate is too small, which leads to slow convergence, while a learning rate that is too large can hinder convergence and cause fluctuate. In order to improve the efficiency and classification ability of SVM based on stochastic gradient descent algorithm, three algorithms of adaptive learning rate SGD are used to solve support vector machine, which are Adagrad, Adadelta and Adam. The experimental results show that the algorithm based on Adagrad, Adadelta and Adam for solving the linear support vector machine has faster convergence speed and higher testing precision.
Keywords: Stochastic gradient descent, Large-scale learning, Support vector machines, Adagrad, Adadelta, Adam

INTRODUCTION

SGD is a simple and effective method, many works focus on designing variants of SGD that can reduce the variance and improve the complexity. Some popular methods include the Pegasos method [1], the stochastic gradient descent with Barzilai-Borwein update step for SVM [2], Budgeted Stochastic Gradient Descent for Large-Scale SVM Training [3], Bi-level stochastic gradient for large-scale support vector machine [4], and the stochastic variance reduced gradient method [5].

These methods are proven to converge linearly on strong convex problems. Pegasos performed stochastic gradient descent on the primal objective with a carefully chosen step size, which improves and guarantees convergence. Some recent works that discuss the improved approaches for SGD [6-11], such as quasi-Newton stochastic gradient descent, accelerated proximal stochastic dual coordinate ascent, stochastic dual coordinate ascent methods, scalability of stochastic gradient descent based on smart sampling techniques, and beyond the regret barrier algorithms for stochastic strongly convex optimization [12]. Presented an ensemble of support vector machine for text-independent speaker recognition.

In this paper, we focus on the problem of choosing the learning rate for SGD. The problem of stochastic gradient descent algorithm is that it is difficult to choose the proper learning rate. A learning rate is too small, which leads to slow convergence, while a learning rate that is too large can hinder convergence and cause fluctuate. In order to improve the efficiency and classification ability of SVM based on stochastic gradient descent algorithm, three algorithms of adaptive learning rate SGD are used to solve support vector machine, which are Adagrad, Adadelta and Adam. The experimental results show that the algorithm based on Adagrad, Adadelta and Adam for solving the linear support vector machine has faster convergence speed and higher testing precision.

STOCHASTIC GRADIENT DESCENT FOR SVM

In order to deal with the large-scale data classification problems, we describe the algorithms of stochastic gradient descent for SVM.

Consider a binary classification problem with examples $S = \{(\mathbf{x}_i, y_i),\ \ i = 1, \cdots, N\}$, where instance $\mathbf{x}_i \in R^d$ is a *d*-dimensional input vector and $y_i \in \{+1, -1\}$ is the label. Training an SVM classifier $f(\mathbf{x}) = \mathrm{sgn}(\mathbf{w}^T \mathbf{x})$ using *S*, where **w** is a vector of weights associated with each input, which is formulated as solving the following optimization problem

$$\min \quad p_t(\mathbf{w}) = \frac{\lambda}{2}\|\mathbf{w}\|^2 + l(\mathbf{w}; (\mathbf{x}_t, y_t)) \ , \tag{1}$$

where $l(\mathbf{w}; (\mathbf{x}_t, y_t)) = \max(0, 1 - y_t \mathbf{w}^T \mathbf{x}_t)$ is the *hinge loss* function and $\lambda \geq 0$ is a regularization parameter used to control model complexity.

SGD works iteratively. It starts with an initial guess of the model weight $\mathbf{w}_1$, and at *t*-th round it updates the current weight $\mathbf{w}_t$ as

$$\begin{aligned} \mathbf{w}_{t+1} &= \mathbf{w}_t - \eta_t \nabla_t p_t(\mathbf{w}_t) \\ &= (1 - \eta_t \lambda)\mathbf{w}_t + \eta_t \mathbf{1}[y_t \langle \mathbf{w}_t, \mathbf{x}_t \rangle] y_t \mathbf{x}_t \end{aligned} \tag{2}$$

where

$$\mathbf{1}[y_t \langle \mathbf{w}_t, \mathbf{x}_t \rangle < 1] = \begin{cases} 1, & if\ y_t \langle \mathbf{w}_t, \mathbf{x}_t \rangle < 1 \\ 0, & otherwise. \end{cases}$$

which is the indicator function which takes a value of one if its argument is true (**w** yields non-zero loss on the example (**x**, y)), and zero otherwise. We then update using a step size of $\eta_t = 1/(\lambda t)$. After a predetermined number *T* of iterations, we output the last iterate $\mathbf{w}_{t+1}$.

Then, the decision function for SVM with SGD is as follows

$$f_{t+1}(\mathbf{x}) = \mathrm{sgn}(\mathbf{w}_{t+1}^T \mathbf{x}) \tag{3}$$

ADAPTIVE LEARNING RATE SGD ALGORITHM FOR SVM

Stochastic gradient descent parameter update rule:

$$\theta = \theta - \eta \cdot \nabla_\theta J\left(\theta; x^{(i)}, y^{(i)}\right) \tag{4}$$

In the following, we present three adaptive learning rate SGD algorithms for SVM. It is especially suited for learning from large datasets.

Adam SVM

Adaptive Moment Estimation (Adam) [13] is a method that computes adaptive learning rates for each parameter. We use Adam method to optimize SVM.

$$\begin{aligned} m_t &= \beta_1 m_{t-1} + (1 - \beta_1)\nabla_\theta J(\theta) \\ v_t &= \beta_2 v_{t-1} + (1 - \beta_2)\nabla_\theta J(\theta)^2 \end{aligned} \tag{5}$$

where $\nabla_\theta J(\theta)^2 = \nabla_\theta J(\theta) \otimes \nabla_\theta J(\theta)$, $\otimes$ is an element-wise matrix-vector multiplication.

$$\begin{aligned} \hat{m}_t &= \frac{m_t}{1 - \beta_1^t} \\ \hat{v}_t &= \frac{v_t}{1 - \beta_2^t} \end{aligned} \tag{6}$$

The Adam update rule:

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t + \varepsilon}} \hat{m}_t \tag{7}$$

Algorithm 1：Adam SVM

1. Input： S, λ, T, ε, β_1, β_2, η
2. Initialize： $w_1 = \vec{0}$, $\hat{m}_1 = \vec{0}$, $\hat{v}_1 = \vec{0}$, ε=1e-8, β_1=0.9, β_2=0.999, η=0.001
3. for $t = 1, \cdots, T$
4. choose $i_t \in \{1, \ldots, |S|\}$ uniformly at random
5. if $y_{i_t} \langle w_t, x_{i_t} \rangle < 1$, then
6. $\nabla_{t+1} = \lambda w_t - \alpha_t y_{i_t} x_{i_t}$
7. else
8. $\nabla_{t+1} = \lambda w_t$
9. $m_{t+1} = \beta_1 m_t + (1 - \beta_1) \nabla_{t+1}$
10. $v_{t+1} = \beta_2 v_t + (1 - \beta_2) \nabla^2_{t+1}$
11. $\hat{m}_{t+1} = \dfrac{m_{t+1}}{1 - \beta_1^t}$ $\hat{v}_{t+1} = \dfrac{v_{t+1}}{1 - \beta_2^t}$,
12. $w_{t+1} = w_t - \dfrac{\eta}{\sqrt{\hat{v}_{t+1} + \varepsilon}} \hat{m}_{t+1}$
13. Output： w_{T+1}

Adagrad SVM

Adagrad [14] is an algorithm for gradient-based optimization that does just this: It adapts the learning rate to the parameters, performing larger updates for infrequent and smaller updates for frequent parameters. For this reason, it is well-suited for dealing with sparse data.

The Adagrad update rule:

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{G_t + \varepsilon}} \otimes \nabla_\theta J(\theta_t) \tag{8}$$

Algorithm 2：Adagrad SVM

1. Input： S, λ, T, η
2. Initialize： $w_1 = \vec{0}$, $G_1 = \vec{0}$, η=0.01, ε=1e-8
3. for $t = 1, \cdots, T$
4. choose $i_t \in \{1, \ldots, |S|\}$ uniformly at random
5. if $y_{i_t} \langle w_t, x_{i_t} \rangle < 1$, then
6. $\nabla_{t+1} = \lambda w_t - \alpha_t y_{i_t} x_{i_t}$
7. else
8. $\nabla_{t+1} = \lambda w_t$
9. $w_{t+1} = w_t - \dfrac{\eta}{\sqrt{G_t + \varepsilon}} \otimes \nabla_{t+1}$
10. $G_{t+1} = G_t + \nabla_{t+1} \otimes \nabla_{t+1}$
11. Output： w_{T+1}

Adadelta SVM

Adadelta [15] is an extension of Adagrad that seeks to reduce its aggressive, monotonically decreasing learning rate. The parameter update takes the form:

$$\Delta\theta_{t+1} = -\frac{RMS[\Delta\theta]_{t-1}}{RMS[g]_t} \otimes \nabla_\theta J(\theta_t)$$
$$\theta_{t+1} = \theta_t + \Delta\theta_{t+1} \tag{9}$$

$RMS[\Delta\theta]_t$ and $RMS[g]_t$ update are thus:

$$E[\Delta\theta^2]_t = \gamma E[\Delta\theta^2]_{t-1} + (1-\gamma)\Delta\theta_t^2$$
$$E[\nabla^2]_t = \gamma E[\nabla^2]_{t-1} + (1-\gamma)\nabla_\theta J(\theta_t)^2 \tag{10}$$

Algorithm 3 : Adadelta SVM

1. Input : S, λ, T, ε, γ
2. Initializa : $w_1 = \vec{0}$, $E[\nabla^2]_1 = \vec{0}$, $E[\Delta w^2]_1 = \vec{0}$, ε=1e-8, γ=0.9
3. for $t = 1, \cdots, T$
4. choose $i_t \in \{1, \ldots, |S|\}$ uniformly at random
5. if $y_{i_t} \langle w_t, x_{i_t} \rangle < 1$, then
6. $\nabla_{t+1} = \lambda w_t - \alpha_t y_{i_t} x_{i_t}$
7. else
8. $\nabla_{t+1} = \lambda w_t$
9. $E[\nabla^2]_{t+1} = \gamma E[\nabla^2]_t + (1-\gamma)\nabla_{t+1}^2$
10. $\Delta w_{t+1} = -\frac{E[\Delta w^2]_t}{E[\nabla^2]_{t+1}} \otimes \nabla_{t+1}$
11. $w_{t+1} = w_t + \Delta w_{t+1}$
12. $E[\Delta w^2]_{t+1} = \gamma E[\Delta w^2]_t + (1-\gamma)\Delta w_{t+1}^2$
13. Output :

EXPERIMENTAL RESULTS

In this section, we perform some experiments that demonstrate the efficacy of our algorithm. The basic SGD algorithm is Pegasos [2]. To evaluate the classification accuracy and convergence rate of four methods, several datasets are used to illustrate in the linear kernel situations. Machine has four E5-2609 2.50GHz processors and 4GB RAM memory. The operating system is the CentOS-6.4.

We tested the performance of four methods on three large datasets and four standard real datasets, three large datasets are derived from Pascal Large Scale Learning Challenge, four standard real datasets are downloaded from LIBSVM website. The Usps and Mnist datasets are used for the task of classifying digits 0, 1, 2, 3, 4 versus the rest of the classes. The original Letter dataset's labels represent 26 alphabets and we set the former 13 alphabets as positive class and the rest as negative class. We use the linear kernel and the regularization parameter λ in our experiments. The datasets characteristics and the parameters are given in Table 1.

Table-1: **Datasets and Parameters**

Dataset	#Training	#Testing	#Features
Alpha	400,000	100,000	500
Gamma	400,000	100,000	500
Delta	400,000	100,000	500
Mnist	60,000	10,000	780
Letter	15,000	5,000	16
Usps	7,291	2,007	256

Table-2: shows the testing accuracy of four methods for linear kernel on six datasets.

Table-2: **Comparisons of four methods**

Dataset	Pegasos	Adam	Adagrad	Adadelta
Alpha	72.68	76.69	77.65	77.15
Gamma	73.15	78.77	79.91	79.12
Delta	70.77	76.73	78.21	77.25
Mnist	87.03	87.46	87.68	85.30
Letter	73.51	73.46	70.71	73.66
Usps	83.83	84.13	83.43	83.94

Fig-1-6 shows the convergence rate four methods with the number of iteration growing.

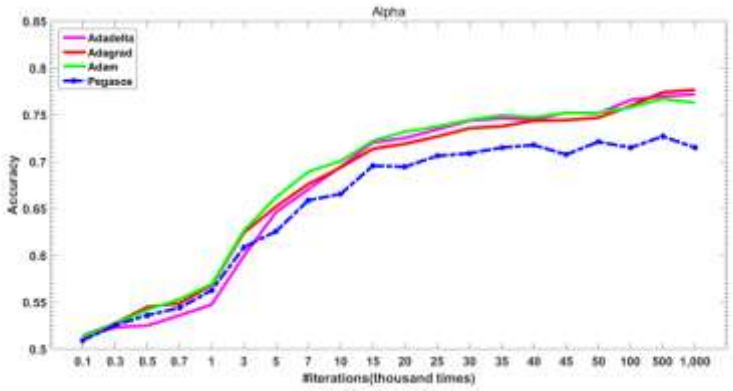

Fig-1: **Comparisons of four methods on Alpha dataset**

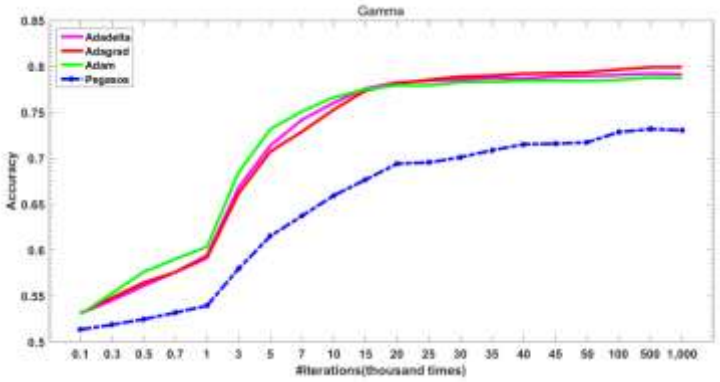

Fig-2: **Comparisons of four methods on Gamma dataset**

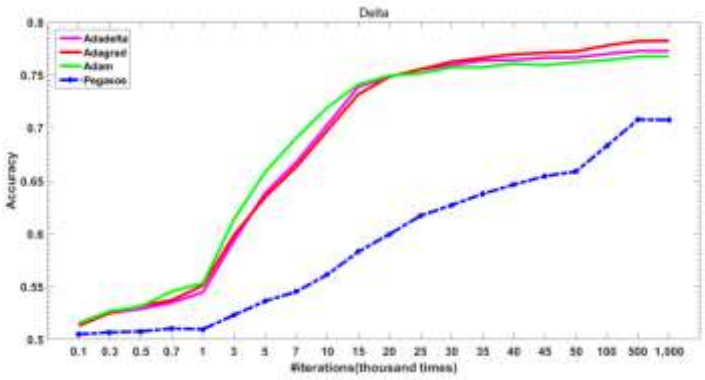

Fig-3: **Comparisons of four methods on Delta dataset**

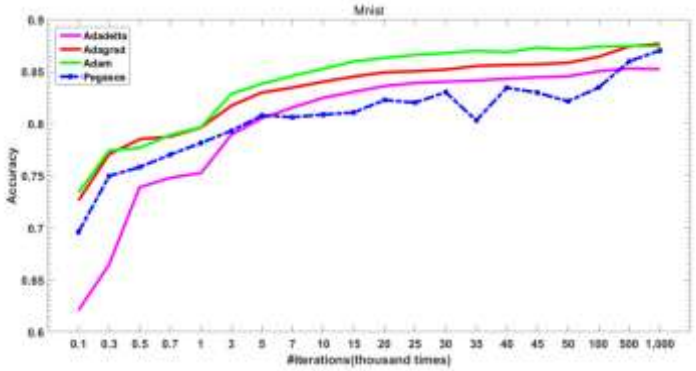

Fig-4: **Comparisons of four methods on Mnist dataset**

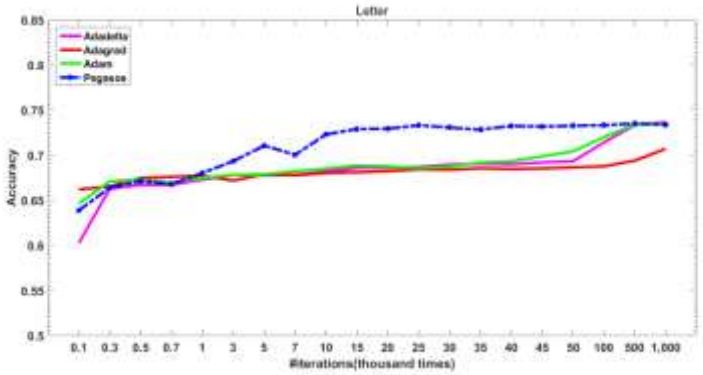

Fig-5: **Comparisons of four methods on Letter dataset**

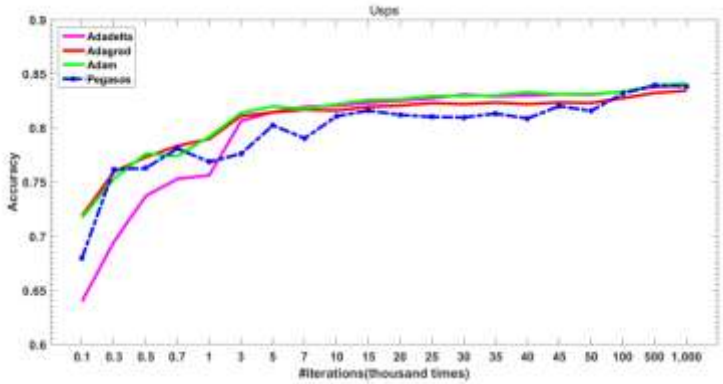

Fig-6: **Comparisons of four methods on Usps dataset**

Figures 1-4 shows that three methods (Adam, Adagrad, Adadelta for SVM) for linear kernel has a faster convergence rate than Pegasos on four datasets (Alpha, Gamma, Delta, Mnist). Figure 5 show that Pegasos has a faster convergence rate than three methods (Adam, Adagrad, Adadelta for SVM) on Letter dataset. Figure 6 show that four methods has almost same convergence rate on Usps dataset.

CONCLUSION

In this paper, we focus on the problem of choosing the learning rate for SGD. The problem of stochastic gradien t descent algorithm is that it is difficult to choose the proper learning rate. A learning rate is too small, which leads to slo w convergence, while a learning rate that is too large can hinder convergence and cause fluctuate. In order to improve the efficiency and classification ability of SVM based on stochastic gradient descent algorithm, three algorithms of adaptive learning rate SGD are used to solve support vector machine, which are Adagrad, Adadelta and Adam. The experimental r

esults show that the algorithm based on Adagrad, Adadelta and Adam for solving the linear support vector machine has f aster convergence speed and higher testing precision.

ACKNOWLEDGMENT

This research is supported by the natural science foundation of Hebei Province No. F2015201185.

REFERENCES

1. Shalev-Shwartz S, Singer Y, Srebro N, Cotter A. Pegasos: Primal estimated sub-gradient solver for svm. Mathematic al programming. 2011 Mar 1;127(1):3-30.
2. Krzysztof Sopyla, Pawel Drozda, Stochastic Gradient Descent with Barzilai-Borwein update step for SVM, Informat ion Sciences, 316, 2015, 218-233.
3. Zhuang Wang, Koby Crammer, Slobodan Vucetic, Breaking the Curse of Kernelization: Budgeted Stochastic Gradie nt Descent for Large-Scale SVM Training, Journal of Machine Learning Research, 13, 2013, 3103-3131.
4. Nicolas Couellan, Wenjuan Wang, Bi-level stochastic gradient for large scale support vector machine, Neurocomputi ng, 153, 2015, 300-308.
5. R. Johnson and T. Zhang, Accelerating Stochastic Gradient Descent using predictive variance reduction. In Advance s in Neural Information Processing Systems, 2013, 315-323.
6. Bordes A, Bottou L, Gallinari P. Sgd-qn: Careful quasi-newton stochastic gradient descent. Journal of Machine Lear ning Research. 2009;10(Jul):1737-1754.
7. Bordes A, Bottou L, Gallinari P, Chang J, Smith SA. Erratum: Sgdqn is less careful than expected. Journal of Machi ne Learning Research. 2010;11(Aug):2229-2240.
8. Shai Shalev-Shwartz, Tong Zhang, Accelerated proximal stochastic dual coordinate ascent for regularized loss mini mization, Math. Program. 155, 2016, 105-145.
9. Shalev-Shwartz S, Zhang T. Stochastic dual coordinate ascent methods for regularized loss minimization. Journal of Machine Learning Research. 2013;14(Feb):567-599.
10. Clémençon S, Bellet A, Jelassi O, Papa G. Scalability of stochastic gradient descent based on "smart" sampling tech niques. Procedia Computer Science. 2015 Jan 1;53:308-315.
11. Hazan E, Kale S. Beyond the regret minimization barrier: optimal algorithms for stochastic strongly-convex optimiz ation. Journal of Machine Learning Research. 2014 Jan 1;15(1):2489-2512.
12. Lei Z, Yang Y, Wu Z. Ensemble of support vector machine for text-independent speaker recognition. Int. J. Comput. Sci. Networks Secur. 2006 May;6(5):163-167.
13. Kingma D, Ba J. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980. 2014 Dec 22.
14. Duchi J, Hazan E, Singer Y. Adaptive subgradient methods for online learning and stochastic optimization. Journal o f Machine Learning Research. 2011;12(Jul):2121-59.
15. Zeiler MD. ADADELTA. An adaptive learning rate method. arXiv preprint arXiv:1212.5701. 2012 Dec 22.
16. Ma JT. Learning to detect malicious urls. University of California, San Diego; 2010.
17. Chih-Chung Chang and Chih-Jen Lin, LIBSVM. A library for support vector machines. 2017.