DOI: 10.21276/sjpms.2016.3.4.7

Scholars Journal of Physics, Mathematics and Statistics **ISSN 2393-8056 (Print)**
Sch. J. Phys. Math. Stat. 2016; 3(4):161-165 **ISSN 2393-8064 (Online)**
©Scholars Academic and Scientific Publishers (SAS Publishers)
(An International Publisher for Academic and Scientific Resources)

Stochastic Gradient Descent with SVM for Imbalanced Data Classification

Lu Shuxia[1]*, Zhu Chenxu[2], Zhou Mi[1]
[1]Key Lab. of Machine Learning and Computational Intelligence, College of Mathematics and Information Science, Hebei University, Baoding City , Hebei Province, 071002, China
[2]College of Science, Northwest Agriculture & Forestry University, Yangling City , Shanxi Province, 712100, China

***Corresponding Author:**
Lu Shuxia
Email: cmclusx@126.com

Abstract: Stochastic Gradient Descent (SGD) is an attractive choice for SVM training. SGD leads to a result that the probability of choosing majority class is far greater than that of minority class for imbalanced classification problem. In order to deal with the large-scale imbalanced data classification problems, a method named stochastic gradient descent algorithm with SVM for imbalanced data classification is proposed. First, to deal with imbalanced data classification problems, we define the weight according to the size of positive and negative dataset. Then, a fast learning algorithm on large datasets called the weighted stochastic gradient descent algorithm with SVM is proposed, which helps to reduce the hyperplane offset to the minority class, thus solve the large-scale imbalanced data classification problems. Experimental results on real datasets show that the proposed method is effective.
Keywords: Stochastic gradient descent, Weight; Imbalanced data, Large-scale learning, Support vector machines

INTRODUCTION

Recent advances in large-scale learning resulted in many algorithms for training SVMs using large data. CVM [1, 2] and parallel SVMs are the successful methods to train SVM from large data. SGD is a recently popularized approach that can be used for online training of SVM, in which the solver is based on the SGD algorithm and have demonstrated its effectiveness for the classification of large datasets with fast convergence and small memory requirements. Pegasos [5] performed stochastic gradient descent on the primal objective with a carefully chosen step size, which improves and guarantees convergence. Krzysztof [3] proposed stochastic gradient descent with Barzilai-Borwein update step for SVM. Wang [4] gives budgeted stochastic gradient descent for large-scale SVM training. This is achieved by controlling the number of SVs through one of the several budget maintenance strategies. Nicolas [5] proposed a bi-level stochastic gradient for large-scale support vector machine with automatic selection of the hyperparameter. Recently there are many improved approaches for SGD [6-12], such as quasi-Newton stochastic gradient descent, accelerated proximal stochastic dual coordinate ascent, stochastic dual coordinate ascent methods, SGD based on smart sampling techniques.

In the real world, these training samples are not always balanced. In an imbalanced dataset, the majority class have a large percentage for all the samples, while the samples in minority class just occupy a small part of all the samples. Many researchers have worked to solve this problem so that the classification performance of the majority class and that of minority class are good at the same time. To solve this problem, two kinds of methods have been proposed: one is based on sampling method and the other one is based on sample weighting method [13-16]. Sampling method includes: under sampling method and oversampling method. The sample weighting approach to the imbalanced data classification problem is to apply the weights to the training data points.

In this paper, we focus on the large and imbalanced datasets effective classification problem, a stochastic gradient descent algorithm with SVM for imbalanced data classification is proposed. It consists of two stages. The first stage is to obtain the weight according to the size of positive and negative dataset. In the second stage, a fast learning algorithm on large datasets called the weighted stochastic gradient descent algorithm with SVM (WSGD) is proposed, which helps to reduce the hyperplane offset to the minority class, thus solve the large scale imbalanced data classification problems. Experiments on large classification datasets also demonstrated that the proposed method has comparable performance.

WEIGHTED STOCHASTIC GRADIENT DESCENT FOR SVM

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

The weighted linear stochastic gradient descent for SVM (WLSGD)

Consider a binary classification problem with examples $S=\{(\mathbf{x}_i, y_i),\quad i=1,\cdots,N\}$, 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 $\mathbf{w}$ is a vector of weights associated with each input, which is formulated as solving the following optimization problem

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

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. s_t is the weight, which is set according to the size of positive and negative dataset. See the section of setting the weight for imbalanced problem in detail.

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

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta_t \nabla_t p_t(\mathbf{w}_t) = (1-\eta_t\lambda)\mathbf{w}_t + s_t\eta_t \mathbf{1}[\mathrm{y}_t \langle \mathbf{w}_t, \mathbf{x}_t \rangle < 1]\, \mathrm{y}_t\, \mathbf{x}_t \tag{2}$$

where

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

which is the indicator function which takes a value of one if its argument is true ($\mathbf{w}$ yields non-zero loss on the example ($\mathbf{x}$, 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 WLSGD is as follows

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

Setting the Weight for Imbalanced Problem

To deal with imbalanced dataset, we simply set the weight according to the size of positive and negative dataset. The data in the majority class have to receive lower weight than those in the minority class receives.

When the size of positive dataset is N_{pos} and that of negative dataset is N_{neg}, the weights are defined as

$$s_i = \begin{cases} 1/N_{pos} & \text{if } y_i = 1, \\ 1/N_{neg} & \text{otherwise.} \end{cases} \tag{4}$$

To maintain the weight ratio and make the convergence speed faster, we also use the following weighting formulation

$$s_i = \begin{cases} 1 & \text{if } y_i = 1,\ N_{pos} \geq N_{neg}, \\ N_{neg}/N_{pos} & \text{if } y_i = 1,\ N_{pos} < N_{neg}, \\ N_{pos}/N_{neg} & \text{if } y_i = -1,\ N_{pos} \geq N_{neg}, \\ 1 & \text{if } y_i = -1,\ N_{pos} < N_{neg}. \end{cases} \tag{5}$$

The weighted linear stochastic gradient descent for SVM (WLSGD) is given in algorithm 1.

Algorithm 1 WLSGD
Input: data *S*, regularization parameter λ, a predetermined number *T* of iterations ;
Initialize**:** $\mathbf{w}_1 = \mathbf{0}$;
Compute the weight s_t according to the formulation (5);
for $t = 1,\cdots,T$ do
choose $(\mathbf{x}_t, \mathrm{y}_t)$ uniformly at random;
$\mathbf{w}_{t+1} \leftarrow (1-\eta_t\lambda)\mathbf{w}_t$
if $\mathrm{y}_t \langle \mathbf{w}_t, \mathbf{x}_t \rangle < 1$ then
$\mathbf{w}_{t+1} \leftarrow \mathbf{w}_{t+1} + s_t\eta_t\, \mathrm{y}_t\, \mathbf{x}_t$; // compute $\mathbf{w}_{t+1}$ according to the formulation (2)

9. else
10. $\mathbf{w}_{t+1} \leftarrow \mathbf{w}_{t+1}$;
11. end if
12. end if
13. Output: $f_{t+1}(\mathbf{x}) = \mathrm{sgn}(\mathbf{w}_{t+1}{}^T\mathbf{x})$.

The weighted kernelized stochastic gradient descent for SVM (WKSGD)

SGD for SVM can be used to solve non-linear problems when combined with Mercer kernels. After introducing a nonlinear function ϕ that maps $\mathbf{x}$ from the input to the feature space and replacing $\mathbf{x}$ with $\phi(\mathbf{x})$, the optimization problem can be described as

$$\min \; p_t(\mathbf{w}) = \frac{\lambda}{2}\|\mathbf{w}\|^2 + s_t \cdot l\left(\mathbf{w};(\phi(\mathbf{x}_t), y_t)\right) \tag{6}$$

where $l\left(\mathbf{w};(\phi(\mathbf{x}_t), y_t)\right) = \max\left(0, 1 - y_t\mathbf{w}^T\phi(\mathbf{x}_t)\right)$ is the *hinge loss* function.

At *t*-th round it updates the current weight $\mathbf{w}_t$ as

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta_t\nabla_t p_t(\mathbf{w}_t) = (1-\eta_t\lambda)\mathbf{w}_t + s_t\eta_t\mathbf{1}[\mathrm{y}_t\langle\mathbf{w}_t, \phi(\mathbf{x}_t)\rangle < 1]\mathrm{y}_t\,\phi(\mathbf{x}_t)$$

It starts with an initial $\mathbf{w}_1 = \mathbf{0}$, update a step size $\eta_t = 1/(\lambda\,\mathrm{t})$, and for all *t* we can rewrite $\mathbf{w}_{t+1}$ as

$$\begin{aligned}\mathbf{w}_{t+1} &= \frac{1}{\lambda t}\sum_{i=1}^{t} s_{i_t}\mathbf{1}[\mathrm{y}_{i_t}\langle\mathbf{w}_t, \phi(\mathbf{x}_{i_t})\rangle < 1]\mathrm{y}_{i_t}\,\phi(\mathbf{x}_{i_t}) \\ &= \frac{1}{\lambda t}\sum_{j=1}^{N} s_j\alpha_{t+1}[\mathrm{j}]\mathrm{y}_j\,\phi(\mathbf{x}_j)\end{aligned} \tag{7}$$

For each *t*, let $\alpha_{t+1} \in R^N$ be the vector such that $\alpha_{t+1}[\mathrm{j}]$ counts how many times example *j* has been selected so far and we had a non-zero loss on it, namely,

$$\alpha_{t+1}[\mathrm{j}] = \left|\left\{t' \le t : i_{t'} = j \wedge \left(y_j\langle\mathbf{w}_{t'}, \phi(\mathbf{x}_j)\rangle < 1\right)\right\}\right| \tag{8}$$

Then, the decision function for WKSGD is as follows:

$$f_{t+1}(\mathbf{x}) = \mathrm{sgn}(\mathbf{w}_{t+1}{}^T\phi(\mathbf{x})) = \frac{1}{\lambda t}\sum_{j=1}^{N} s_j\alpha_{t+1}[\mathrm{j}]\mathrm{y}_j\,k(\mathbf{x}_j, \mathbf{x}) \tag{9}$$

Only one element of α is changed at each iteration. The algorithm does not refer to the implicit mapping $\phi(\cdot)$ and only use the kernel function. This WKSGD implementation is given in algorithm 2.

Algorithm 2 WKSGD

1. Input: data *S*, regularization parameter λ, a predetermined number *T* of iterations ;
2. Initialize: $\alpha_1 = \mathbf{0}$
3. Compute the weight s_t according to the formulation (5);
4. for $t = 1, \cdots, T$ do
5. choose $(\mathbf{x}_{i_t}, \mathrm{y}_{i_t})$ uniformly at random;
6. For all $j \ne i_t$, *set* $\alpha_{t+1}[\mathrm{j}] = \alpha_t[\mathrm{j}]$;
7. if $y_{i_t}\frac{1}{\lambda t}\sum_{j=1}^{N} s_j\alpha_t[\mathrm{j}]\mathrm{y}_j\,k(\mathbf{x}_j, \mathbf{x}_{i_t}) < 1$, then
8. $\alpha_{t+1}[\mathrm{i}_t] = \alpha_{t+1}[\mathrm{i}_t] + 1$; // count the selected times of example *j* with a non-zero loss on it.
9. else
10. $\alpha_{t+1}[\mathrm{i}_t] = \alpha_{t+1}[\mathrm{i}_t]$;
11. end if
12. end if
13. Output: $f_{t+1}(\mathbf{x}) = \mathrm{sgn}(\mathbf{w}_{t+1}{}^T\phi(\mathbf{x})) = \frac{1}{\lambda t}\sum_{j=1}^{N} s_j\alpha_{t+1}[\mathrm{j}]\mathrm{y}_j\,k(\mathbf{x}_j, \mathbf{x})$.

EXPERIMENTAL RESULTS

In this section, we conduct the performance comparison of the four methods for real problems: MNIST, Ijcnnl, Shuttle, Letter, Usps, Adult. Most of the datasets are taken from the UCI machine learning repository [17]. Usps is taken from database [18]. The multi-classification dataset are artificially divided into binary classification dataset, which constitute the imbalanced dataset. The description of datasets is shown in Table 1.

The Gaussian function is taken as the kernel function $k(x_i, x_j) = \exp(-\|x_i - x_j\|^2 / \sigma^2)$. Set the kernel function width σ=1.5, a predetermined number of iterations $T = 10^6$. The regularization parameters are shown in Table 2.

Table 1: Introduction to datasets

datasets	#classes	Training size	Testing size（N_{pos}、N_{neg}，）	#features
MNIST	10	60000	10000（974、9026）	780
Ijcnnl	2	91701	49990（4853、45137）	22
Shuttle	7	43500	14500（2155、12345）	9
Letter	26	10000	10000（353、9647）	16
Usps	10	7291	2007（198、1809）	256
Adult	2	24974	12554（119、12435）	123

Table 2: Parameter setting

	LSGD	WLSGD	KSGD	WKSGD
λ	10^{-4}	10^{-4}	10^{-4}	10^{-4} (Mnist), 10^{-9} (Ijcnn), 10^{-8} (Shuttle), 10^{-7} (Letter), 10^{-15} (Usps), 10^{-4} (Adult)

Considering the imbalanced nature of the training datasets, the geometric mean accuracy is adopted to evaluate the performance of our algorithms,

$$g = \sqrt{a^+ \cdot a^-}$$

where

$$a^+ = \frac{\#\text{positive samples correctly classified}}{\#\text{ total positive samples classified}} \times 100\%,$$

$$a^- = \frac{\#\text{negative samples correctly classified}}{\#\text{ total negative samples classified}} \times 100\%.$$

Table 3: Experimental results of the testing geometric mean accuracy

Algorithms	LSGD	WLSGD	KSGD	WKSGD
MNIST	78.86	72.19	87.60	94.19
Ijcnnl	51.42	78.35	55.14	72.36
Shuttle	0	40.51	5.15	94.54
Letter	23.14	67.51	75.63	86.04
Usps	90.83	91.35	94.66	94.03
Adult	0	76.08	9.16	73.21

Twenty trials were conducted for the four algorithms and the average results are shown in Table 3 and Table 4. Table 3 shows the performance comparison of accuracy of the four methods in the real-world problems; the testing geometric mean accuracy of WLSGD and WKSGD is higher than LSGD and KSGD methods in most datasets. Table 4 shows the performance comparison of average training and testing time of the four methods in the real-world problems. As observed from the Table 4, WLSGD and WKSGD methods compare to LSGD and KSGD methods with almost same learning speed in most datasets.

Table 4: Experimental results of the average training and testing time

Algorithms	LSGD	WLSGD	KSGD	WKSGD
MNIST	597.53(s)	600.09(s)	777.34(s)	834.21(s)
	6.15(s)	5.94(s)	20.94(s)	17.82(s)
Ijcnnl	22.94(s)	21.21(s)	31.91(s)	46.89(s)
	1.11(s)	1.09(s)	0.20(s)	0.39(s)
Shuttle	11.75(s)	3.52(s)	5.03(s)	7.49(s)
	0.16(s)	0.16(s)	5.03(s)	4.18(s)
Letter	21.84(s)	20.81(s)	1.9(s)	3.77(s)
	0.21(s)	0.21(s)	0.11(s)	0.17(s)
Usps	277.26(s)	280.59(s)	58.74(s)	18.62(s)
	0.56(s)	0.61(s)	7.84(s)	2.88(s)
Adult	128.71(s)	97.10(s)	18.85(s)	65.80(s)
	1.39(s)	1.20(s)	0.45(s)	3.06(s)

CONCLUSION

We focus on the large and imbalanced datasets effective classification problem, the stochastic gradient descent algorithms with SVM for imbalanced data classification are proposed. It consists of two stages. The first stage is to obtain the weight according to the size of positive and negative dataset. In the second stage, a fast learning algorithm on large datasets called the weighted stochastic gradient descent algorithm with SVM is proposed, which helps to reduce the hyperplane offset to the minority class, thus solve the large scale imbalanced data classification problems. Experiments on real datasets show that the proposed method is effective.

ACKNOWLEDGEMENT

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

REFERENCES

1. Tsang IW, Kwok JT, Cheung PM. Core vector machines: fast SVM training on very large data sets. Journal of Machine Learning Research. 2005;6:363-392.
2. Singer SSY, Srebro N. Pegasos: Primal Estimated sub-Gradient Solver for SVM. Mathematical Programming. 2011;127(1):3-30.
3. Sopyla K, Drozda P. Stochastic Gradient Descent with Barzilai-Borwein update step for SVM. Information Sciences. 2015;316:218-233.
4. Wang Z, Crammer K, Vucetic S. Breaking the Curse of Kernelization: Budgeted Stochastic Gradient Descent for Large-Scale SVM Training. Journal of Machine Learning Research. 2013;13:3103-3131.
5. Couellan N, Wang W. Bi-level stochastic gradient for large scale support vector machine. Neurocomputing. 2015;153: 300-308.
6. Bordes A, Bottou L, Gallinari P. SGD-QN: careful quasi-Newton stochastic gradient descent. J. Mach. Learn. 2009;10:1737-1754.
7. Bordes A, Bottou L, Gallinari P. Sgdqn is less careful than expected. J. Mach. Learn. 2010;11:2229-2240.
8. Janakiraman VM, Nguyen XL, Dennis. Stochastic gradient based extreme learning machines for stable online learning of advanced combustion engines. Neurocomputing. 2016;177:304-316.
9. Shalev-Shwartz S, Zhang T. Accelerated proximal stochastic dual coordinate ascent for regularized loss minimization. Math. Program. 2016;155:105-145.
10. Shwartz S, Zhang. Stochastic dual coordinate ascent methods for regularized losss minimization. J. Mach. Learn. 2013;14:567-599.
11. Clemencon S, Bellet A, Jelassi O. Scalability of Stochastic Gradient Descent based on Smart Sampling Techniques. Procedia Computer Science. 2015;53:308–315.
12. Hazan E, Kale S. Beyond the Regret Minimization Barrier: Optimal Algorithms for Stochastic Strongly Convex Optimization. Journal of Machine Learning Research. 2014;15:2489-2512.
13. Li K, Kong X, Lu Z. Boosting weighted ELM for imbalanced learning. Neurocomputing. 2014;128:15-21.
14. He H, Garcia EA. Learning from imbalanced data. IEEE Trans. Knowl. Data Eng. 2009;21(9):1263–1284.
15. Sun YM, Wong KC, Kamel MS. Classification of imbalanced data: a review. International journal of pattern recognition and artificial intelligence. 2009;23(4):687-719.
16. Zong W, Huang GB, Chen Y. Weighted extreme learning machine for imbalance learning. Neurocomputing. 2013;101:229-242.
17. Frank A, Asuncion A. UCI machine learning repository, 2010. Available from http: //archive. ics. uci. Edu / ml.
18. Hull JJ. A database for handwritten text recognition research. IEEE Trans. Pattern Anal. Mach. Intell. 1994;16(5):550-554.