Optimizing Top-k Multiclass SVM via Semismooth Newton Algorithm

San Zhang, Si Li, *Fellow, IEEE*

***Abstract*—Top-k performance has recently received increasing attention in large data categories. Advances like top-k multiclass SVM have consistently improved the top-k accuracy. However, the key ingredient in the state-of-the-art optimization scheme based upon Stochastic Dual Coordinate Ascent (SDCA) relies on the sorting method which yields $O(d \log d)$ complexity. In this paper, we leverage the semismoothness of the problem and propose an optimized top-k multiclass SVM algorithm, which employs semismooth Newton algorithm for the key building block to improve the training speed. Our method enjoys a local superlinear convergence rate in theory. In practice, experimental results confirm the validity. Our algorithm is 4 times faster than the existing method in large synthetic problems; Moreover, on real-world datasets it also shows significant improvement in training time.**

***Index Terms*—Multiclass SVM, top-k error, SDCA optimization, root finding, semismooth Newton method.**

I. Introduction

MULTICLASS classification is a fundamental problem in pattern recognition and has been attracting much attention in the machine learning community [1]–[5]. The major challenges posed by large-scale dataset for training multiclass classifier lies not only in the data size, but also in the number of data categories [6]–[8]. For example, there are 1000 object categories for the image classification task in ImageNet visual recognition challenge [8]. When the object classes increase, an important issue, i.e. the overlapping categories, emerges. Many real-world classification tasks involve large numbers of overlapping categories [9], which lead to class ambiguity. Thus, it is customary to report top-k accuracy for large-scale object recognition problems [10]–[13], where the top-k accuracy is the fraction of test data for which the counted correct label is among the top k predicted labels by the model. However, all these reported top-k error rates are based on the top-1 error. Recently, Lapin et al. generalize Crammer and Singer's Multiclass Support Vector Machine (MSVM) [14] to top-k MSVM which leads to improvements in top-k performance [15], [16]. Since the direct extension of MSVM to nonconvex top-k zero-one loss will encounter a computationally intractable problem; it minimizes the surrogate function, i.e., so-called top-k hinge loss which is a tight convex upper bound of the top-k zero-one loss. Furthermore, a highly efficient SDCA procedure [17] is proposed to solve the optimization problem.

S. Zhang and S. Li are with the Department of Automation, Hefei University, Anhui Province, 230031, China (e-mails: {sanzhang, sili}@mails.hfut.edu.cn).

II. Top-k Multiclass Support Vector Machine

We first review the well-known MSVM proposed by Crammer and Singer [14]. Given a set of n instance-label pairs $(\boldsymbol{x}_i, y_i),\ i = 1, \cdots, n$, where $\boldsymbol{x}_i \in \mathbb{R}^p$ and the associated label y_i is an integer from the set $\mathcal{Y} = \{1, \cdots, Y\}$. Let the weight vector $\mathbf{w}_j$ be the j-th column of parameter matrix $\mathbf{W}_{p\times Y}$. Crammer and Singer's MSVM solves the following problem

$$\min_{\mathbf{W}} \ \frac{\lambda}{2}\sum_{j\in\mathcal{Y}} \|\mathbf{w}_j\|^2 + \frac{1}{n}\sum_{i=1}^{n} \ell(\mathbf{W}; \boldsymbol{x}_i, y_i), \tag{1}$$

where λ is referred to as the positive regularization parameter and $\ell(\cdot)$ is called the loss function of example $(\boldsymbol{x}_i, y_i)$. The loss function is defined as

$$\ell(\mathbf{W}; \boldsymbol{x}_i, y_i) := \max_{j\in\mathcal{Y}} \left\{\mathbb{I}(j \neq y_i) + \langle \mathbf{w}_j - \mathbf{w}_{y_i}, \boldsymbol{x}_i\rangle\right\},$$

where $\mathbb{I}(\cdot)$ is the indicator function which takes a value of one if its argument is true. Then the multiclass decision function has the form

$$\operatorname*{argmax}_{j\in\mathcal{Y}} \langle \mathbf{w}_j, \boldsymbol{x}\rangle.$$

Let $\boldsymbol{e}_j$ be the j-th unit vector in $\mathbb{R}^Y$ and $\mathbf{1}$ with ones in all elements. For every i, let $\boldsymbol{c}_i = \mathbf{1} - \boldsymbol{e}_{y_i}$ and $\boldsymbol{b}_i = \mathbf{W}^\top \boldsymbol{x}_i - \left(\mathbf{W}^\top \boldsymbol{x}_i\right)_{y_i}$. To lighten the notation we denote the j-th largest component of $\boldsymbol{b}_i$ by $b_{[j]}$, i.e., $b_{[1]} \geq b_{[2]} \geq \cdots \geq b_{[Y]}$.

Thus, the loss function can be rewritten as

$$\ell(\boldsymbol{b}_i) = \max\left\{0, (\boldsymbol{c}_i + \boldsymbol{b}_i)_{[1]}\right\}.$$

Recently, Lapin et al. extended the above loss function to the top-k hinge loss [15], [16],

$$\ell_k(\boldsymbol{b}_i) = \max\left\{0, \frac{1}{k}\sum_{j=1}^{k} (\boldsymbol{c}_i + \boldsymbol{b}_i)_{[j]}\right\}, \tag{2}$$

where $1 \leq k < Y$. We show that the top-k multiclass SVM can be cast as an unconstrained optimization problem

$$\min_{\mathbf{W}} \ \frac{\lambda}{2}\sum_{j\in\mathcal{Y}} \|\mathbf{w}_j\|^2 + \frac{1}{n}\sum_{i=1}^{n} \ell_k(\mathbf{W}; \boldsymbol{x}_i, y_i). \tag{3}$$

A. Dual Problem of Top-k MSVM

To solve the top-k MSVM problem (3) using the SDCA framework, one may first derive its dual form. Following the notation given in [18], let $\mathbf{X}_i \in \mathbb{R}^{pY\times Y}$ be the matrix whose j-th column is $\mathrm{vec}(\boldsymbol{x}_i(\boldsymbol{e}_j - \boldsymbol{e}_{y_i})^\top)$ and $\boldsymbol{w} = \mathrm{vec}(\mathbf{W})$. Then,

$$\boldsymbol{b}_i = \mathbf{X}_i^\top \boldsymbol{w}.$$

Hence we can reformulate the primal optimization problem of top-k MSVM as

$$\min_{\boldsymbol{w}\in\mathbb{R}^{pY}} \; P(\boldsymbol{w}) := \frac{\lambda}{2}\boldsymbol{w}^\top\boldsymbol{w} + \frac{1}{n}\sum_{i=1}^{n}\ell_k(\boldsymbol{w};\mathbf{X}_i,y_i). \quad (4)$$

We obtain its equivalent optimization problem

$$\begin{aligned} \min\ & \frac{1}{2}\|\boldsymbol{\alpha}_i\|_2^2 + \boldsymbol{a}_i^\top\boldsymbol{\alpha}_i + \frac{1}{2}\left(\mathbf{1}^\top\boldsymbol{\alpha}_i\right)^2 \\ s.t.\ & 0 \le -\boldsymbol{\alpha}_i \le \frac{1}{k}\sum -\boldsymbol{\alpha}_i \\ & \sum -\boldsymbol{\alpha}_i \le 1 \\ & \alpha_i^{y_i} = 0 \end{aligned} \quad (5)$$

where

$$\boldsymbol{a}_i = \frac{1}{\rho_i}\left(\boldsymbol{c}_i + \mathbf{X}_i^\top\hat{\boldsymbol{w}}\right),\ \rho_i = \frac{1}{n\lambda}\|\boldsymbol{x}_i\|^2.$$

Here calculating $\mathbf{X}_i^\top\hat{\boldsymbol{w}}$ still takes $O(pY^2)$ operations, which is too expensive. We reshape the vector $\hat{\boldsymbol{w}}$ into a p-by-Y matrix $\hat{\boldsymbol{W}}$. Thus the computation

$$\mathbf{X}_i^\top\hat{\boldsymbol{w}} = \hat{\boldsymbol{W}}^\top\boldsymbol{x}_i - (\hat{\boldsymbol{W}}^\top\boldsymbol{x}_i)_{y_i}$$

takes $O(pY)$ operations. In order to avoid the heavy notation, we drop the subscript of $\boldsymbol{a}_i$ and let $\boldsymbol{z} = -\boldsymbol{\alpha}_i^{\setminus y_i},\ s = \sum z_j$, the above optimization problem (5) can be rewritten as

$$\begin{aligned} \min_{\boldsymbol{z},s}\ & \frac{1}{2}\|\boldsymbol{z}-\boldsymbol{a}\|^2 + \frac{1}{2}s^2 \\ s.t.\ & s = \sum z_j \\ & s \le 1 \\ & 0 \le z_j \le s/k. \end{aligned} \quad (6)$$

Once the problem (6) is solved, a sufficient increase of the dual objective will be achieved. Whilst for the primal problem, the process will lead to the update

$$\boldsymbol{w} = \boldsymbol{w} + \frac{1}{n\lambda}\mathbf{X}_i\left(\boldsymbol{\alpha}_i - \boldsymbol{\alpha}_i^{\text{old}}\right). \quad (7)$$

A pseudo-code of the SDCA algorithm for the top-k MSVM is depicted as Algorithm 1. To have the first $\boldsymbol{w}$, we can initialize $\boldsymbol{\alpha}_i = \mathbf{0}$ and then $\boldsymbol{w} = \mathbf{0}$.

Algorithm 1 Stochastic Dual Coordinate Ascent Algorithm for Top-k MSVM

Require: $\boldsymbol{\alpha}$, λ, k, ϵ
1: $\boldsymbol{w} \leftarrow \sum_i \frac{1}{n\lambda}\mathbf{X}_i\boldsymbol{\alpha}_i$
2: **while** $\boldsymbol{\alpha}$ is not optimal **do**
3: Randomly permute the training examples
4: **for** $i = 1, \cdots, n$ **do**
5: $\boldsymbol{\alpha}_i^{old} \leftarrow \boldsymbol{\alpha}_i$
6: Update $\boldsymbol{\alpha}_i$ by solving sub-problem (6)
7: $\boldsymbol{w} \leftarrow \boldsymbol{w} + \frac{1}{n\lambda}\mathbf{X}_i\left(\boldsymbol{\alpha}_i - \boldsymbol{\alpha}_i^{old}\right)$
8: **end for**
9: **end while**
Ensure: $\boldsymbol{w}$, $\boldsymbol{\alpha}$

III. Experiments

In this section, we first demonstrate the performance of our semismooth Newton method on synthetic data. Then, we apply our algorithm to the top-k multiclass SVM problem to show the efficiency compared with the existing method in [15]. Our algorithms used to solve problem are implemented in C with a Matlab interface and run on 3.1GHz Intel Xeon (E5-2687W) Linux machine with 128G of memory. The compiler used is GCC 4.8.4. Both our code and libsdca package of [15] ensure the "-O3" optimization flag is set. The experiments are carried out in Matlab 2016a. All the implementation will be released publicly on website.

A. Efficiency of the Proposed Algorithm

To investigate the scalability in the problem dimension of our algorithm, two synthetic problems are randomly generated with d ranging between $50,000$ and $2,000,000$. In the first test problem, a_j is randomly chosen from the uniform distribution $U(15,25)$ as in [19], [20]. In the second test, following the setup of [15], [21], data entries are sampled from the normal distribution $N(0,1)$. In the third synthetic problem, a_j is chosen by independent draws from uniform distribution $U(-1,1)$. For pure comparison, we assume the problem without the constraint $s \le r$. Thus, the knapsack problem which corresponds to the $s = r$ case will not occur in these synthetic problems.

We first present numerical results to investigate the scalability of our proposed algorithm compared with the sorting-based method for different values of $k = 1, 5, 10$. Fig. 1(a), 1(b) and 1(c) correspond to the first, the second and the third test problems respectively. They tell us that the running times grow linearly with the problem size for both the sorting-based method and our proposed algorithm. However, our algorithm **??** is consistently much faster than the sorting-based method. When the problem size $d \ge 2 \times 10^6$, our proposed algorithm is 2.5 times faster in the first problem, and 4 times faster in both the second and the third problems respectively. In addition to the superlinear convergence, the semismooth Newton method accesses to accurate solutions in a few iterations. Our numerical results suggest that it usually takes 3$\sim$5 iterations to converge.

TABLE I: Datasets used in the experimental evaluation.

Dataset	Classes	Features	Training size	Testing size
FMD	10	2048	500	500
News20	20	15,478	15,935	3,993
Letter	26	16	15,000	5,000
INDoor67	67	4,096	5,360	1,340
Caltech101	101	784	4,100	4,100
Flowers	102	2,048	2,040	6,149
CUB	200	2,048	5,994	5,794
SUN397	397	4,096	19,850	19,850
ALOI	1,000	128	86,400	21,600
ImageNet	1,000	2,048	1,281,167	50,000

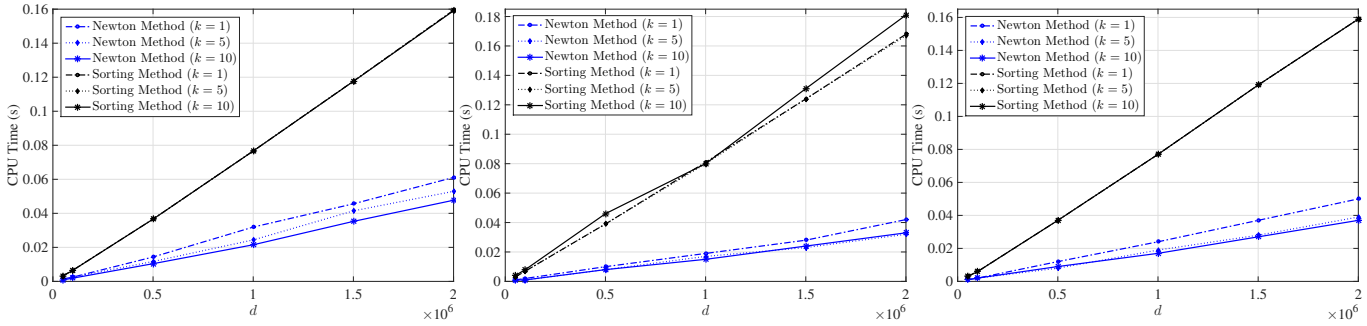

Fig. 1: Scaling of our algorithm compared with sorting method. Left: $a_j \sim U(10, 25)$. Middle: $a_j \sim N(0, 1)$. Right: $a_j \sim U(-1, 1)$.

IV. Concluding Remarks

In this paper, we leverage the semismoothness of the optimization problem and develop an optimized top-k multiclass SVM. While our proposed semismooth Newton method enjoys the local superlinear convergence rate, we also present an efficient algorithm to obtain the starting point, which works quite well in practice for the Newton iteration. Experimental results on both synthetic and real-world datasets show that our proposed method scales better with larger numbers of categories and offers faster convergence compared with the existing sorting-based algorithm. We note that there are many other semismooth scenarios, such as ReLU activation function in deep neural networks and hinge loss in the empirical risk minimization problem. It must be very appealing to exploit the semismooth structure and propose more efficient machine learning algorithms in future work.

Acknowledgements

The authors would like to thank the reviewers for their valuable suggestions on improving this paper. Thanks also goes to Wu Wang for the helpful email exchange.

References

[1] C. M. Bishop, Pattern Recognition and Machine Learning. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006.

[2] C.-W. Hsu and C.-J. Lin, "A comparison of methods for multiclass support vector machines," IEEE transactions on Neural Networks, vol. 13, no. 2, pp. 415–425, 2002.

[3] R. Rifkin and A. Klautau, "In defense of one-vs-all classification," Journal of machine learning research, vol. 5, no. Jan, pp. 101–141, 2004.

[4] G.-X. Yuan, C.-H. Ho, and C.-J. Lin, "Recent advances of large-scale linear classification," Proceedings of the IEEE, vol. 100, no. 9, pp. 2584–2603, 2012.

[5] A. Rocha and S. K. Goldenstein, "Multiclass from binary: Expanding one-versus-all, one-versus-one and ecoc-based approaches," IEEE Transactions on Neural Networks and Learning Systems, vol. 25, no. 2, pp. 289–302, 2014.

[6] J. Deng, A. Berg, K. Li, and L. Fei-Fei, "What does classifying more than 10,000 image categories tell us?" Computer Vision–ECCV 2010, pp. 71–84, 2010.

[7] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva, "Learning deep features for scene recognition using places database," in Advances in neural information processing systems, 2014, pp. 487–495.

[8] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein et al., "Imagenet large scale visual recognition challenge," International Journal of Computer Vision, vol. 115, no. 3, pp. 211–252, 2015.

[9] L. Cai and T. Hofmann, "Exploiting known taxonomies in learning overlapping concepts." in IJCAI, vol. 7, 2007, pp. 708–713.

[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in Advances in neural information processing systems, 2012, pp. 1097–1105.

[11] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," arXiv preprint arXiv:1409.1556, 2014.

[12] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 1–9.

[13] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.

[14] K. Crammer and Y. Singer, "On the algorithmic implementation of multiclass kernel-based vector machines," Journal of Machine Learning Research, vol. 2, pp. 265–292, 2001.

[15] M. Lapin, M. Hein, and B. Schiele, "Top-k multiclass svm," in Advances in Neural Information Processing Systems, 2015, pp. 325–333.

[16] ——, "Loss functions for top-k error: Analysis and insights," in 29th IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2016). Las Vegas, NV, USA: IEEE Computer Society, 2016, pp. 1468–1477.

[17] Y. Zhang and X. Lin, "Stochastic primal-dual coordinate method for regularized empirical risk minimization." in ICML, 2015, pp. 353–361.

[18] S. Shalev-Shwartz and T. Zhang, "Accelerated proximal stochastic dual coordinate ascent for regularized loss minimization," Mathematical Programming, vol. 155, no. 1, pp. 105–145, 2016.

[19] R. Cominetti, W. F. Mascarenhas, and P. J. Silva, "A newton's method for the continuous quadratic knapsack problem," Mathematical Programming Computation, vol. 6, no. 2, pp. 151–169, 2014.

[20] K. Kiwiel, "Variable fixing algorithms for the continuous quadratic knapsack problem," Journal of Optimization Theory and Applications, vol. 136, no. 3, pp. 445–458, 2008.

[21] P. Gong, K. Gai, and C. Zhang, "Efficient euclidean projections via piecewise root finding and its application in gradient projection," Neurocomputing, vol. 74, no. 17, pp. 2754–2766, 2011.