ONLINE MULTI-LABEL LEARNING WITH ACCELERATED NONSMOOTH STOCHASTIC GRADIENT DESCENT

Sunho Park[1] *and Seungjin Choi*[1,2]

[1] Department of Computer Science and Engineering, POSTECH, Korea
[2] Division of IT Convergence Engineering, POSTECH, Korea
{titan, seungjin}@postech.ac.kr

ABSTRACT

Multi-label learning refers to methods for learning a set of functions that assigns a set of relevant labels to each instance. One of popular approaches to multi-label learning is *label ranking*, where a set of ranking functions are learned to order all the labels such that relevant labels are ranked higher than irrelevant ones. Rank-SVM is a representative method for label ranking where ranking loss is minimized in the framework of max margin. However, the dual form in Rank-SVM involves a quadratic programming which is generally solved in cubic time in the size of training data. The primal form is appealing for the development of online learning but involves a nonsmooth convex loss function. In this paper we present a method for online multi-label learning where we minimize the primal form using the accelerated nonsmooth stochastic gradient descent which has been recently developed to extend Nesterov's smoothing method to the stochastic setting. Numerical experiments on several large-scale datasets demonstrate the computational efficiency and fast convergence of our proposed method, compared to existing methods including subgradient-based algorithms.

Index Terms— Label ranking, multi-label learning, Nesterov's method, nonsmooth minimization, stochastic gradient descent

1. INTRODUCTION

Multi-label learning seeks a classification function that predicts a set of relevant labels for an instance, whereas in multi-class problems a single label is assigned to an instance. Multi-label problems arise in various applications, including text mining, scene classification, image annotation, and bioinformatics, to name a few.

An easy but popular approach to multi-label learning is one-versus-all (OVA) strategy, which is also known as binary relevance (BR), where binary classifiers for each category label are independently learned. OVA or BR, however, does not take the dependency between labels into account. Class imbalance problem becomes severe especially when the number of class labels is large, since each binary classifier in OVA build a positive class by collecting data points belonging to one label of interest and construct a negative class by collecting all remaining data points in the rest of labels.

Another popular approach to multi-label learning is *label ranking* [5, 7, 10, 19], where a set of ranking functions are learned to order all the labels such that relevant labels are ranked higher than irrelevant ones. The ranking loss, which measures the number of label pairs ranked in a wrong order, is minimized in label ranking, hence pairwise relations between class labels is implicitly considered. Moreover, label ranking methods can alleviate the class imbalance problem since the minimization of the ranking loss is equivalent to the maximization of the area under ROC curve (AUC) which is known to be a predictive performance measure less sensitive to class imbalance [8, 12].

Rank-SVM [7] is a representative implementation for multi-label ranking where a set of ranking functions are learned by minimizing the ranking loss in the max margin framework. The dual formulation in Rank-SVM enables us to use arbitrary kernel functions, but it involves quadratic programming which requires quadratic memory space and cubic time complexity in the size of training data [7]. While an approximation optimization, followed from Franke and Wolfe [9], was proposed to to reduce the computational complexity, it is still prohibitive for large scale data.

In this paper we present a method for online multi-label learning where we minimize the primal form using the accelerated nonsmooth stochastic gradient descent [17] which has been recently developed to extend Nesterov's smoothing method to the stochastic setting. Since the model is iteratively updated by the gradient calculated from only a single data point, its computational requirement is considerably cheaper than batch methods. Numerical experiments on several large-scale datasets demonstrate the computational efficiency and fast convergence of our proposed method, compared to existing methods, including subgradient-based algorithms.

The rest of this paper is organized as follows. Section 2 presents related work and Section 3 provides a brief overview of the label ranking in the framework of regularized empirical ranking loss minimization. Section 4 presents our main contribution where we describe how the accelerated stochastic gradient descent method [17] is applied to the label ranking problem. Numerical experiments are provided in Section 5 and conclusions are drawn in Section 6.

2. RELATED WORK

In the last decade, various methods have been proposed to solve multi-label problems. As pointed out in [21], existing methods can be categorized into (1) first-order; (2) second-order; (3) higher-order approaches, in accordance with the order of correlation considered by classifiers. The first-order approach is usually simple and easy to implement because the dependence between labels is not considered at all. In the second-order approach, pairwise relations between labels are taken into account. The high-order approach assumes that each label is affected by all other labels's information. In most of cases, however, that kind of information is modeled in an indirect way, such as a chain of binary classifiers [18] or an extraction of the shared subspace among labels [11]. In this paper, we focus on the label ranking that belongs to the second-order approach.

Various methods for label ranking have been proposed in different frameworks. In [19], the existing boosting method is extended to solve the label ranking (which is called AdaBoost.MR). In [7], the

authors provide an extension of SVM for label ranking problems. In [10], the authors introduce the concept of constraint classification, where each instance is labeled to be consistent with a set of predefined constraints. In [5], the ranking among labels is encoded into a preference graph, and the ranking functions are learned by a boosting-based method with the corresponding preference graph. In [4], a set of perceptrons for each label are jointly learned in online learning setting to minimize a given ranking loss function. The method is called multi-class multi-label perceptron (MMP).

Stochastic gradient decent (SGD) approaches [2] are useful for large scale datasets, since they update model parameters only using the gradient information calculated from a single data point at each iteration. In addition there have been proposed several methods to solve the minimization of nonsmooth functions in the stochastic gradient descent manner. One simplest way is to use subgradient information instead of gradient, however, its convergence rate is considerably slow [20], i.e., it converges as $f(\theta_t) - \min_\theta f(\theta) \leq \mathcal{O}(\log t/t)$ after t iterations. To improve the convergence rate, the authors in [17] have proposed an accelerated stochastic gradient descent methods based on Nesterov's smooth method [15] which approximates the nonsmooth convex function by a strongly convex function with a Lipschitz continuous gradient. They have provided a proof that the convergence of the method is improved to $\mathcal{O}(1/t)$ [17].

3. LABEL RANKING FOR MULTI-LABEL LEARNING

In this section, we will briefly explain the label ranking in the framework of regularized ranking loss minimization. Suppose that we are given a set of N instance-label pairs, $\{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^N$, where $\boldsymbol{x}_i \in \mathcal{X} \subset \mathbb{R}^D$ are instances and $\boldsymbol{y}_i \in \mathcal{Y} = \{1, -1\}^K$ are label vectors with $|\mathcal{Y}| = 2^K$. Denote by $\boldsymbol{Y} \in \mathbb{R}^{K \times N}$ *label matrices*, where $Y_{k,i} = 1$ if the instance $\boldsymbol{x}_i$ is assigned label k and otherwise $Y_{k,i} = -1$. We also denote by $\mathcal{Y}_i = \{k | Y_{k,i} = 1\}$ a set of relevant labels of the instance $\boldsymbol{x}_i$, and by $\overline{\mathcal{Y}}_i$ the complementary set of $\mathcal{Y}_i$ in $\mathcal{Y}$.

The prediction task of multi-label learning can be implemented as a ranking system which produces the higher score value for all relevant labels of an instance than its irrelevant labels. Let $f_k : \mathbb{R}^D \to \mathbb{R}$, then our goal is to find a set of K number of functions, such that

$$f_k(\boldsymbol{x}_i) \geq f_l(\boldsymbol{x}_i), \text{ where } k \in \mathcal{Y}_i, l \in \overline{\mathcal{Y}}_i. \tag{1}$$

In the rest of the paper, we assume that each function f_k is linear, i.e., $f_k(\boldsymbol{x}) = \boldsymbol{m}_k^\top \boldsymbol{x} + b_k$, where $\boldsymbol{m}_k \in \mathbb{R}^D$ and $b_k \in \mathbb{R}$. For notational simplicity, we further assume that each $\boldsymbol{m}_k$ is augmented with the bias term b_k and a constant 1 is added into instance $\boldsymbol{x}$, i.e., $\boldsymbol{m}_k$ and $\boldsymbol{x}$ become a vector of $D+1$ length. With $\boldsymbol{M} = [\boldsymbol{m}_1, ..., \boldsymbol{m}_K] \in \mathbb{R}^{(D+1) \times K}$, the empirical pairwise ranking loss of the multi-label ranking functions $\boldsymbol{M}$ on the training data is defined as [19]

$$R_{\text{emp}}(\boldsymbol{M}) = \frac{1}{N} \sum_{i=1}^N R_i(\boldsymbol{M}), \tag{2}$$

where R_i is the number of pairs of labels that are ranked in the wrong order in the ith example, such that

$$R_i(\boldsymbol{M}) = \frac{1}{N_i} \sum_{(k,l) \in \mathcal{Y}_i \times \overline{\mathcal{Y}}_i} \mathbf{1}\Big(\boldsymbol{m}_k^\top \boldsymbol{x}_i \leq \boldsymbol{m}_l^\top \boldsymbol{x}_i\Big), \tag{3}$$

where $N_i = |\mathcal{Y}_i||\overline{\mathcal{Y}}_i|$ and $\mathbf{1}(\pi)$ is the *0-1 loss function* which equals 1 if the predicate π is true, otherwise 0.

Due to the discrete nature of the 0-1 loss, a direct optimization of the ranking loss (2) is not an easy task. We instead consider the hinge loss function h, which is a continuous convex upper bound on the 0-1 loss in (2), as a surrogate function:

$$h(a) = [1 - a]_+, \tag{4}$$

where $[a]_+ = \max\{0, a\}$. Thus, the continuous convex upper bound on the empirical pairwise rank loss (2) is defined as

$$\bar{R}_{\text{emp}}(\boldsymbol{M}) = \frac{1}{N} \sum_{i=1}^N \bar{R}_i(\boldsymbol{M}), \tag{5}$$

where $\bar{R}_i(\boldsymbol{M}) = \frac{1}{N_i} \sum_{(k,l) \in \mathcal{Y}_i \times \overline{\mathcal{Y}}_i} \Big[1 - \boldsymbol{m}_k^\top \boldsymbol{x}_i + \boldsymbol{m}_l^\top \boldsymbol{x}_i\Big]_+$. With the appropriate regularizer (e.g., Frobenius norm) for the parameter matrix $\boldsymbol{M}$, the minimization of problem (5) can be expressed with introducing the slack variables $\xi_{ikl} \geq 0$:

$$\begin{aligned} \min_{\boldsymbol{M}, \boldsymbol{\xi}} \quad & \frac{\lambda}{2} \|\boldsymbol{M}\|_F^2 + \frac{1}{N} \sum_{i=1}^N \frac{1}{N_i} \sum_{(k,l) \in \mathcal{Y}_i \times \overline{\mathcal{Y}}_i} \xi_{ikl}, \\ \text{subject to} \quad & \boldsymbol{m}_k^\top \boldsymbol{x}_i - \boldsymbol{m}_l^\top \boldsymbol{x}_i \geq 1 - \xi_{ikl}, \quad (k,l) \in \mathcal{Y}_i \times \overline{\mathcal{Y}}_i, \end{aligned} \tag{6}$$

where $\lambda \geq 0$ is a regularization constant that trades off an empirical loss and the regularization, and $\|\boldsymbol{M}\|_F^2 = \sum_{k=1}^K \|\boldsymbol{m}_k\|^2$ is Frobenius norm. Note that we can arrive at the equivalent minimization problem to (6) in the max-margin framework [7]. We also have to note that each function $\bar{R}_i(\boldsymbol{M})$ in (5) is convex, but nonsmooth.

The minimization problem (5) or (6) can be solved in the dual form [6, 7] that allows us to use arbitrary kernel functions. However, solving a quadratic programming generally requires $\mathcal{O}(N^2)$ memory space and $\mathcal{O}(N^3)$ time complexity. Such a complexity is prohibitive for large-scale data sets. To circumvent this problem, the authors proposed the approximation optimization based on Franke and Wolfe method [9] (also known as conditional gradient [1]), but the time complexity of the method is still $\mathcal{O}(N^2 K)$ [6, 7].

4. ONLINE LEARNING WITH ACCELERATED NONSMOOTH STOCHASTIC GRADIENT DESCENT

In this section we present a method for online multi-label learning where the primal form problem (5) is minimized using the accelerated nonsmooth stochastic gradient descent optimization [17].

For the nonsmooth objective function as in our case, the classical update rule of SGD at iteration t is given by [3]

$$\boldsymbol{M}_{t+1} \quad = \quad \boldsymbol{M}_t - \eta_t \Big[\lambda \boldsymbol{M}_t + \partial \bar{R}_t(\boldsymbol{M}_t)\Big], \tag{7}$$

where $\bar{R}_t$ is a loss function in (5) associated with the example available at time t (we use a symbol t, instead of i, to represent the iteration), $\partial \bar{R}_t$ is a subgradient of $\bar{R}_t$ and $\eta_t = 1/[\lambda(t + \Omega)]$ is a step size at t where Ω is a large positive constant. The method is guaranteed to converge to optimal solution within some error range under certain constraints on the objective function and the step size [3]. However, the stochastic gradient descent method for nonsmooth functions yields considerably slow convergence rates [3]. To solve this problem, the authors in [17] propose a accelerated nonsmooth stochastic gradient descent (ANSGD) method based on Nesterov's smooth method [15]. In the rest of the section, we first present the update rule of ANSGD [17] for our problem (5) in abstract level, and then give its detail computations.

We consider the smooth approximation of the loss $\bar{R}_t$ in (5) via the convex conjugate [15, 22]. Letting g^* be the Fenchel dual of a function g, we would approximate $\bar{R}_t$ by a smooth function

$g^*_{\alpha_t} = (g+\alpha_t d)^*$, where d is a strongly convex function and $\alpha_t \geq 0$ is a decreasing sequence as t increases. Denoting $\mathrm{vec}(\boldsymbol{M}_t)$ by a vectorization operation forming a vector of length $(D+1)K$ by stacking the columns of $\boldsymbol{M}_t$, we assume that $\bar{R}_t$ can be expressed as

$$\begin{aligned}\bar{R}_t(\boldsymbol{M}_t) &= g^*(\boldsymbol{A}_t^\top \mathrm{vec}(\boldsymbol{M}_t)) \\ &= \max_{\boldsymbol{\beta}_t \in \mathcal{Q}} \left\langle \boldsymbol{\beta}_t, \boldsymbol{A}_t^\top \mathrm{vec}(\boldsymbol{M}_t) \right\rangle - g(\boldsymbol{\beta}_t), \end{aligned} \quad (8)$$

where $\boldsymbol{A}_t$ is a appropriate linear transformation determined by an example $(\boldsymbol{x}_t, \boldsymbol{y}_t)$ and $\mathcal{Q}$ is a domain (convex set) of g. Then inserting a non-negative strongly convex function d, e.g., $d(\boldsymbol{\beta}_t) = \frac{1}{2}\|\boldsymbol{\beta}_t\|^2$, we can construct $g^*_{\alpha_t}$ that is the smooth approximation of $\bar{R}_t$:

$$\begin{aligned}&\bar{R}_t(\boldsymbol{M}_t) \\ &\approx g^*_{\alpha_t}(\boldsymbol{A}_t^\top \mathrm{vec}(\boldsymbol{M}_t)) \\ &= \max_{\boldsymbol{\beta}_t \in \mathcal{Q}} \left\langle \boldsymbol{\beta}_t, \boldsymbol{A}_t^\top \mathrm{vec}(\boldsymbol{M}_t) \right\rangle - \left(g(\boldsymbol{\beta}_t) + \frac{\alpha_t}{2}\|\boldsymbol{\beta}_t\|^2 \right). \end{aligned} \quad (9)$$

As pointed out in [15, 22], $g^*_{\alpha_t}$ has several desirable properties: it well approximates the original function $\bar{R}_t$, i.e.,

$$|g^*_{\alpha_t}(\boldsymbol{A}_t^\top \mathrm{vec}(\boldsymbol{M}_t)) - \bar{R}_t(\boldsymbol{M}_t)| \leq \alpha_t C, \text{ for all } \boldsymbol{M}_t, \quad (10)$$

where $C = \max_{\boldsymbol{\beta}_t \in \mathcal{Q}} d(\boldsymbol{\beta}_t)$, and has a Lipschitz continuous gradient with constant at most $\frac{1}{\alpha_t}\|\boldsymbol{A}_t\|^2$, where

$$\|\boldsymbol{A}_t\| = \max_{\boldsymbol{u},\boldsymbol{v}:\|\boldsymbol{u}\|=\|\boldsymbol{v}\|=1} \boldsymbol{u}^\top \boldsymbol{A}_t \boldsymbol{v}.$$

The gradient of $\bar{R}_t$ with respect to $\boldsymbol{M}_t$ also can be approximately calculated using $g^*_{\alpha_t}$ and the definition of subgradient:

$$\begin{aligned}\mathrm{vec}\left(\nabla \bar{R}_t(\boldsymbol{M}_t)\right) &\approx \frac{\partial g^*_{\alpha_t}(\boldsymbol{A}^\top \mathrm{vec}(\boldsymbol{M}_t))}{\partial \mathrm{vec}(\boldsymbol{M}_t)} \\ &= \boldsymbol{A}_t \widehat{\boldsymbol{\beta}}_t, \end{aligned} \quad (11)$$

where $\widehat{\boldsymbol{\beta}}_t = \arg\max_{\boldsymbol{\beta}_t \in Q} g^*_{\alpha_t}(\boldsymbol{A}_t^\top \mathrm{vec}(\boldsymbol{M}_t))$.

In [17], the authors have proposed a stochastic gradient method for nonsmooth functions, where Nesterov's accelerated gradient method [14] is applied to the aforementioned smoothly approximated function. They have proven that the convergence rate of the method is improved to $\mathcal{O}(1/t)$ with the positive regularization constant ($\lambda > 0$) [17] . That algorithm is summarized in Table 1.

Algorithm 1: ANSGD [17]

Output: $\boldsymbol{M}_t$
Initialize $\boldsymbol{M}_0$ and $\boldsymbol{\Psi}_0$
for $t = 0, 1, 2, ...$ **do**
  $\alpha_t = \frac{2}{t+1}, \vartheta_t = \lambda(\alpha_t + 1/2\alpha_t - 1) + 1, \eta_t = \frac{\alpha_t}{\lambda+\vartheta_t}$
  $\boldsymbol{\Upsilon}_t \leftarrow \frac{(1-\alpha_t)(\lambda+\vartheta_t)\boldsymbol{M}_t + \alpha_t \vartheta_t \boldsymbol{\Psi}_t}{\lambda(1-\alpha_t)+\vartheta_t}$
  $\boldsymbol{M}_{t+1} \leftarrow \boldsymbol{\Upsilon}_t - \eta_t \left[\nabla \bar{R}_t(\boldsymbol{\Upsilon}_t) + \lambda \boldsymbol{\Upsilon}_t\right]$
  $\boldsymbol{\Psi}_{t+1} \leftarrow \frac{\vartheta_t \boldsymbol{\Psi}_t - \nabla \bar{R}_t(\boldsymbol{\Upsilon}_t)}{\lambda+\vartheta_t}$
end

In order to apply Algorithm 1 to our case, we first need to specify the function g. To this end, we consider the following equalities:

$$\begin{aligned}&\bar{R}_t(\boldsymbol{M}) \\ &= \frac{1}{N_t} \max_{b_{tkl} \in \{0,1\}} \sum_{(k,l) \in \mathcal{Y}_t \times \overline{\mathcal{Y}}_t} b_{tkl} \left[1 - (\boldsymbol{m}_k^\top \boldsymbol{x}_t - \boldsymbol{m}_l^\top \boldsymbol{x}_t)\right] \\ &= \frac{1}{N_t} \max_{\beta_{tkl} \in [0,1]} \sum_{(k,l) \in \mathcal{Y}_t \times \overline{\mathcal{Y}}_t} \beta_{tkl} \left[1 - (\boldsymbol{m}_k^\top \boldsymbol{x}_t - \boldsymbol{m}_l^\top \boldsymbol{x}_t)\right]. \end{aligned} \quad (12)$$

Thus based on (8) and (12), we can easily determine the form of g and $\boldsymbol{A}_t$. $\boldsymbol{A}_t$ is a $\mathbb{R}^{K(D+1)\times N_t}$ sparse matrix, the jth column of which (corresponding to the pair $(k,l) \in \mathcal{Y}_t \times \overline{\mathcal{Y}}_t$) is constructed by

$$[\boldsymbol{A}_t]_{:,j} = [\boldsymbol{0}, ..., \boldsymbol{0}, -\frac{1}{N_t}\boldsymbol{x}_t^\top, \boldsymbol{0}, ..., \boldsymbol{0}, \frac{1}{N_t}\boldsymbol{x}_t^\top, \boldsymbol{0}, ..., \boldsymbol{0}]^\top. \quad (13)$$

Here, only the kth and lth blocks of the jth column are occupied. In addition the function g is automatically given by

$$g(\boldsymbol{\beta}_t) = \begin{cases} -\frac{1}{N_t}\sum_{(k,l)\in\mathcal{Y}_t\times\overline{\mathcal{Y}}_t} \beta_{tkl} & \text{if } \boldsymbol{\beta}_t \in \mathcal{Q} \\ \infty & \text{otherwise} \end{cases}, \quad (14)$$

with the domain $\mathcal{Q} = [0\ 1]^{N_t}$.

We now explain the details for computing the gradient of $\bar{R}_t$ (11) based on an AUC maximization method in binary classification problems [22], where the classification function is learned to order all examples such that positive examples are ranked higher than negative examples. Since all β_{tkl} are decoupled as in (9), the analytical solution $\hat{\beta}_{tkl}$ can be given by solving each scalar quadratic problem:

$$\hat{\beta}_{tkl} = \mathrm{median}(1, z_{tl} - z_{tk}, 0), \quad (15)$$

where $z_{tk} = \frac{1}{\alpha_t N_t}\left(\boldsymbol{m}_k^\top \boldsymbol{x}_t - \frac{1}{2}\right)$ and $z_{tl} = \frac{1}{\alpha_t N_t}\left(\boldsymbol{m}_l^\top \boldsymbol{x}_t + \frac{1}{2}\right)$. Then, the gradient of $\bar{R}_i$ with respect to $\boldsymbol{m}_k$ can be simplified as

$$\frac{\partial \bar{R}_i(\mathrm{vec}(\boldsymbol{M}_t))}{\partial \boldsymbol{m}_k} = \frac{1}{N_t}\zeta_{tk}\boldsymbol{x}_t, \quad (16)$$

where

$$\zeta_{tk} = \begin{cases} -\sum_{l\in\overline{\mathcal{Y}}_t} \hat{\beta}_{tkl} & \text{if } Y_{k,t} = 1 \\ \sum_{l\in\mathcal{Y}_t} \hat{\beta}_{ilk} & \text{if } Y_{k,t} = -1 \end{cases}. \quad (17)$$

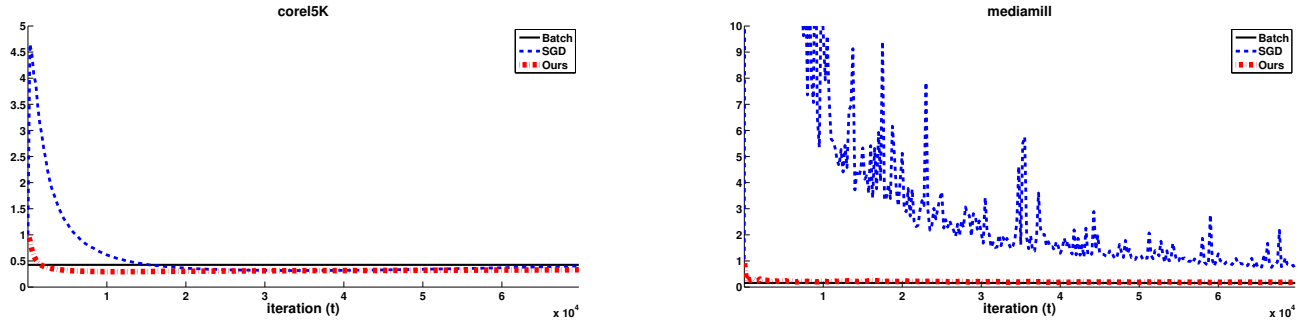
Given $\{\zeta_{tk}\}$, the gradient, $\nabla \bar{R}_t(\boldsymbol{M}_t)$, can be computed in $\mathcal{O}(K(D+1))$. Noth that compared to the update rule (7) of SGD, Algorithm 1 additionally involves the computation of $\{\zeta_{tk}\}$. However, applying the method in [22] into our case, ζ_{tk} can be efficiently calculated after sorting of two lists $\{z_{tk}\}_{k\in\mathcal{Y}_t}$ and $\{z_{tl}\}_{l\in\overline{\mathcal{Y}}_t}$. Its computational complexity is dominated by the complexity of sorting the two lists, and thus is roughly $\mathcal{O}(K \log K)$. For more details on the computations of $\{\zeta_{tk}\}$, one can refer to Section 4.2 in [22].

5. NUMERICAL EXPERIMENTS

We demonstrate the performance of our method on a few multi-label datasets from *mulan*[1], whose detail descriptions are summarized in Table 1. We compare our method with a batch style method for Rank-SVM and two online methods for the label ranking, including a classical SGD [3] to solve the problem (5) (see the update rule (7) of SGD) and MMP [4]. The batch style method for Rank-SVM solves the optimization problem (5) in primal form using the smooth approximation as in our method. Based on Neterov's smooth method in Section 4, the gradient of whole training dataset was calculated, and the limited memory BFGS [13, 16] was applied to find an optimal solution. Note that the original dual formulation for Rank-SVM [7] is prohibited for most datasets in Table 1 because it requires a quadric memory space in the size of training data.

Experiment settings for each method are as follows. For all experiments, we used a random 5-folds cross validation method where the dataset is randomly divided into 5-subsets, in which 1 subset was used for test data and the remaining subsets were used for training data. For the batch Rank-SVM, the regularization constant λ

[1] see http://mulan.sourceforge.net/datasets.html



(a) (b)

Fig. 1. The plot of the regularized hinge ranking loss of our method and SGD for test data, $\frac{\lambda}{2}\|\boldsymbol{M}_t\|_F^2 + \frac{1}{|\mathcal{T}_*|}\sum_{i\in\mathcal{T}_*}\bar{R}_i(\boldsymbol{M}_t)$, as t increases. We consider two datasets, (a) corel5K and (b) mediamill.

Table 1. Data description: LC means the Label Cardinality (the average number of 1's in a label vector).

	domain	# labels (K)	# examples (N)	# features (D)	LC
corel5K	images	374	5,000	499	3.522
rcv1v2	text	101	6,000	47,236	2.880
bibtex	text	159	7,395	1,836	2.402
tmc2007	text	22	28,596	49,060	2.158
mediamill	video	101	43,907	120	4.376
bookmark	text	208	87,856	2,150	2.028

was chosen among $\{10^{-6}, 10^{-5}, ..., 10^{0}\}$ by minimizing the ranking loss on the randomly selected validation set from the training data. This estimated regularization constant was used for SGD and our method. Furthermore, in SGD the constant Ω in (7) was set to 10^3. Note that, although MMP also aims to minimize the pairwise ranking loss, it optimizes the quite different form of objective function to our case. Followed by experimental results in [4], we set *isErr* function as a loss function to be minimized, which equals 1 if any pairs of labels are wrongly ordered and otherwise 0. In addition *uniform update* rule is used for a weighting method as in [4]. For all methods, each element of initial solution $\boldsymbol{M}_0$ was drawn from a Gaussian distribution with mean 0 and variance 0.01. Finally each online method is assumed to assess a randomly picked data point from training dataset at each iteration.

We first empirically compare the convergence rate of our method with SGD. Note that, we did not include the results of MMP since it minimizes the different objective function. In the rest of experiments, we compare the performance of all methods, including MMP, in terms of classification performance. In Fig 1, we plot $\frac{\lambda}{2}\|\boldsymbol{M}_t\|_F^2 + \frac{1}{|\mathcal{T}_*|}\sum_{i\in\mathcal{T}_*}\bar{R}_i(\boldsymbol{M}_t)$, where $\mathcal{T}_*$ is a set of test data. In both cases (for corel5K and mediamill datasets), we can confirm that our method converges significantly faster than SGD.

We then compare the performance of all methods in terms of the predictive performance. Note that, since the label ranking only gives the order of labels, we here consider the AUC value extracted from ROC curve for performance measure. To this end, we first computed ROC curve for K number of scores obtained from each test sample and then averaged their results across all test data. Then, the AUC value was calculated from this sample-averaged ROC curve. Note that by definition of AUC and ranking loss, we can easily see their equivalence relation, i.e., $1 -$ ranking loss $=$ AUC. For three online methods, SGD, MMP and our method, we set the maximum iteration to 7×10^4 and the final solution was averaged among the whole trajectory to remove undesirable fluctuation, i.e., $\widehat{\boldsymbol{M}} = \frac{1}{t}\sum_{j=1}^{t}\boldsymbol{M}_j$. From Table 2, which reports the average AUC values of each method, we notice that our method shows the comparative performance to the batch method, and outperforms other online methods for most cases. As a result, based on its fast convergence rate and reasonable classification performance, we expect that our method can be applied to real-world applications involving multi-label datsets which have large number of samples and labels.

Table 2. Performance comparison of the methods in terms of AUC, where the results are the average values and their standard deviation (the number in parentheses). The bold face represent the best AUC value among three online methods, SGD, MMP and our method.

dataset	Batch method	SGD	MMP	Our method
corel5K	0.895(0.002)	0.885(0.003)	0.843(0.007)	**0.887(0.003)**
rcv1v2	0.969(0.001)	0.963(0.001)	**0.964(0.002)**	**0.964(0.001)**
bibtex	0.946(0.001)	0.935(0.003)	0.935(0.002)	**0.945(0.001)**
tmc2007	0.958(0.001)	0.954(0.002)	0.956(0.001)	**0.957(0.002)**
mediamill	0.953(0.001)	0.945(0.001)	**0.948(0.001)**	0.946(0.001)
bookmark	0.922(0.001)	0.905(0.002)	0.906(0.002)	**0.913(0.001)**

6. CONCLUSIONS

We have presented an online multi-label learning method where the ranking loss function is minimized in the primal form using the accelerated nonsmooth stochastic gradient descent. Since the ranking functions are updated by the gradient calculated from a single data point at each iteration, the computational requirement is considerably cheaper than batch methods. Numerical experiments on several large-scale datasets demonstrated the computational efficiency and fast convergence of our proposed method, compared to existing methods including subgradient-based algorithms.

Acknowledgments: This work was supported by National Research Foundation (NRF) of Korea (2012-0005032), NIPA ITRC Support Program (NIPA-2012-H0301-12-3002), POSTECH Rising Star Program, and NRF World Class University Program (R31-10100).

7. REFERENCES

[1] D. P. Bertsekas, *Nonlinear Programming*. Athena Scientific, 1999.

[2] L. Bottou, "Stochastic learning," in *Advanced Lectures on Machine Learning*, O. Bousquet and U. von Luxburg, Eds. Springer Verlag, 2004, pp. 146–168. [Online]. Available: http://leon.bottou.org/papers/bottou-mlss-2004

[3] S. Boyd and A. Mutapcic, "Subgradient methods," 2007, Lecture Notes for Winter 2006-07, Stanford University.

[4] K. Crammer and Y. Singer, "A family of additive online algortihtms for cagegory ranking," *Journal of Machine Learning Research*, vol. 3, pp. 1025–1058, 2003.

[5] O. Dekel, C. D. Manning, and Y. Singer, "Log-linear models for label ranking," in *Advances in Neural Information Processing Systems (NIPS)*, vol. 16. MIT Press, 2004.

[6] A. Elisseeff and J. Weston, "Kernel methods for multi-labelled classification and categorical regression problems," BIOwulf Technologies, Tech. Rep., 2001.

[7] ——, "A kernel method for multi-labeled classification," in *Advances in Neural Information Processing Systems (NIPS)*, vol. 14. MIT Press, 2002, pp. 681–687.

[8] T. Fawcett, "ROC graphs: Notes and practical considerations for researchers," HP Laboratories, Tech. Rep. HPL-2003-4, 2004.

[9] M. Frank and P. Wolfe, "An algorithm for quadratic programming," *Naval Research Logistics*, vol. 3, pp. 95–110, 1956.

[10] S. Har-Peled, D. Roth, and D. Zimak, "Constraint classification for multiclass classification and ranking," in *Advances in Neural Information Processing Systems (NIPS)*, vol. 14. MIT Press, 2002.

[11] S. Ji, L. Tang, S. Yu, and J. Ye, "Extracting shared subspace for multi-label classification," in *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, Las Vegas, Nevada, USA, 2008.

[12] C. X. Ling, J. Huang, and H. Zhang, "AUC: a better measure than accuracy in comparing learning algorithms," in *Lecture Notes in Artificial Intelligence*, vol. 2671. Springer, 2003, pp. 329–341.

[13] D. C. Liu and J. Nocedal, "On the limited memory BFGS method for large scale optimization," *Mathematical Programming*, vol. 45, no. 3, pp. 503–528, 1989.

[14] Y. Nesterov, *Introductory Lectures on Convex Optimization: A Basic Course*. Kluwer Academic Publishers, 2004.

[15] ——, "Smooth minimization of non-smooth functions," *Mathematical Programming, Series A*, vol. 103, pp. 127–152, 2005.

[16] J. Nocedal, "Updating quasi-Newton matrices with limited storage," *Mathematics of Computation*, vol. 35, no. 151, pp. 773–782, 1980.

[17] H. Ouyang and A. Gray, "Stochastic smoothing for nonsmooth minimizations: Accelerating SGD by exploiting structure," in *Proceedings of the International Conference on Machine Learning (ICML)*, Edinburgh, Scotland, UK, 2012.

[18] J. Read, B. Pfahringer, G. Holmes, and E. Frank, "Classifiere chains for multi-label classification," in *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML-PKDD)*, Bled, Slovenia, 2009.

[19] R. E. Schapire and Y. Singer, "BoosTexter: a boosting-based system for text categorization," *Machine Learning*, vol. 39, pp. 135–168, 2000.

[20] S. Shalev-Shwartz, Y. Singer, and N. Srebro, "Pegasos: primal estimated sub-gradient solver for SVM," in *Proceedings of the International Conference on Machine Learning (ICML)*, Covallis, OR, USA, 2007.

[21] M. L. Zhang and K. Zhang, "Multi-label learning by exploiting label dependency," in *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, Washington, DC, USA, 2010.

[22] X. Zhang, A. Saha, and S. V. N. Vishwanathan, "Smoothing multivariate performance measures," in *Proceedings of the Annual Conference on Uncertainty in Artificial Intelligence (UAI)*, Barcelona, Spain, 2011.